CONTEXT-AWARE TRUST-BASED ACCESS CONTROL FOR UBIQUITOUS SYSTEMS

Malika Yaici, Faiza Ainennas and Nassima Zidi

Computer Department, University of Bejaia, Bejaia, Algeria

ABSTRACT

The ubiquitous computing and context-aware applications experience at the present time a very important development. This has led organizations to open more of their information systems, making them available anywhere, at any time and integrating the dimension of mobile users. This cannot be done without taking into account thoughtfully the access security: a pervasive information system must henceforth be able to take into account the contextual features to ensure a robust access control. In this paper, access control and a few existing mechanisms have been exposed. It is intended to show the importance of taking into account context during a request for access. In this regard, our proposal incorporates the concept of trust to establish a trust relationship according to three contextual constraints (location, social situation and time) in order to decide to grant or deny the access request of a user to a service.

KEYWORDS

Pervasive systems, Access Control, RBAC, Context-awareness, Trust management

1. INTRODUCTION

Ubiquitous computing, which is declined under different terms, corresponds to this technical (r)evolution conceived about fifteen years ago by Weiser [1]. In contrast to traditional computing, the novelty lies in the ability of mobility and integration of systems in the physical environment, and this spontaneously and at multiple scales. The limitation of resources, the distribution of applications, the mobility of terminals, the discovery of services and the automatic deployment of software on these terminals make complex and difficult the implementation of ubiquitous applications. Large-scale implementation of these applications is a first major challenge that many research communities are trying to overcome by proposing different approaches.

In pervasive computing, the context plays a crucial role. IT applications extend their interactions with the environment: new inputs and outputs are used, such as sensors and other mobile devices that interact with the real and physical environment. Ubiquitous applications must therefore be aware of the environment in which they are running, what type of terminal they are running, which user profile to consider for the configuration, what type of network they have for communication where the mobile terminal is, and what computing devices are in the environment. A context-aware application is an application that meets the requirements imposed by this context information. Such an application must be able to capture and manage context information to provide appropriate services.

Natarajan Meghanathan et al. (Eds) : CCSEA, CLOUD, SIPRO, AIFU, SEA, DKMP, NCOM - 2019
pp. 13-32, 2019. © CS & IT-CSCP 2019 DOI: 10.5121/csit.2019.90902

However, access to certain equipment or users' personal data must be highly secure. Setting up a security system requires considering the following issues:

• Authentication: The identification of a given user must be possible and must take into account his context (time, familiar location, surrounding people etc.)
• Access Control: In pervasive environments information is accessible anywhere and anytime. As a result, the administration and integration of different security policies becomes more complex as they are heterogeneous from a structural (role) and semantic (coding, language) point of view.

• Confidentiality and protection of privacy: The misuse of the new technology can compromise the privacy of users. A user has a very limited perception of potential risks from different on-board equipment.

Access control consists of checking whether a human or logic actor has the rights to access a resource. With access control, we are interested in guaranteeing two fundamental properties: Information confidentiality and integrity. Access control must evolve to integrate a dynamic and reactive authorization, based on information related to the environment and more simply on user trust. Contextual information evolves according to an application-independent dynamic. It is also necessary to detect context changes to re-evaluate the authorization.

Context awareness and access control are two similar concepts. They both go through a decision based on the information collected in input, to output an application whose features can be activated and allowed or inactivated and prohibited. In addition, the security of contextual information, especially for the protection of privacy, is a growing problem in computing.
Fine grain access control for a given service must incorporate all relevant contextual information that has significance on access control. In this paper, context-aware access control models are studied to lead to a proposition of a trust-based role-based access control and taking into account context as location, time and social situation.

After this introduction, section 2 is a summary of related works. The proposed solution is given in section 3 and its validation is presented in section 4. A conclusion and some perspectives finish this paper.

2. RELATED WORKS

Discretionary Access Control (DAC) is a conceptual model whose principle is to limit access to objects in relation to the user's identity (humans, machines, etc.) and / or groups to which they belong. Controls on a resource are said to be discretionary in the sense that a user with defined access permission is able to transmit it (indirectly or directly) to any other user. Discretionary access control is generally defined as opposed to Mandatory Access Control (MAC), which imposes mandatory rules to ensure that the intended security objectives are achieved. In this type of access control, subjects can not intervene in the allocation of access rights [2].

From the company access control policy organizational structure, the Role Based Access Control (RBAC) model is used to facilitate the access control administration and to provide an access control model which matches the existent policy[3]. A role represents in an abstract way a particular function in an organization. The role is an intermediate entity between the access

permissions, also called privilege or access right, and the users. It groups together a set of privileges that will then be assigned to users based on their organizational positions. A role can have multiple permissions and permission can be associated with multiple roles. A user can have multiple roles and a role can be assigned to multiple users.

Traditional access control models (RBAC, MAC, DAC) are rigid. Access control decisions cannot be more than allowing or denying access. With these models access authorizations are considered to be known in advance, but in real-world contexts, errors are made and unforeseen situations or emergencies can occur.

Several model proposals have emerged which have proposed the integration of additional features like context. Among the different contexts taken into account in the access control, existing in the literature, we find the location, the time, the state of execution of the applications, the state of the resources, the bandwidth of networks, the activities of each entity, users' intentions, user emotions and environmental conditions.

Taking into account the context imposes new requirements for the definition of access control solutions. These concern, in particular, context awareness. For this, taking into account the notion of context sensitivity is a priority for the development of access control policies.

From the literature study, we have been able to classify these models into four classes:
- Context-aware access control.
- Role-based context-aware access control.
- Trust-based context-aware access control.
- Hybrid access control based on role, trust and context awareness.

2.1. Context-aware Access Control

The role-based access control model, RBAC and its extensions are not suitable for open and peer-to-peer environments that do not assume predefined roles or permissions.

The authors in [4] propose a context-aware and context quality-aware access model for an access request. They explain that the use of the context alone is not enough. Context-awareness is the contextual information quality that is taken into account to decide on the access granting. They propose to base on the authorization or the prohibition of access to resources thanks to indicators of quality of context, in a ubiquitous environment. The quality of context can be defined by the following parameters: Precision, completeness, freshness and accuracy.

CoDRA (Context-based Dynamically Reconfigurable Access) [5] is an Android's access control system. It offers dynamically configurable restrictions based on context and fine-granular policy and enforces various policy configurations at different levels of system operation. Context based on resource features is used to reach the fine-granular policy and policy diversifications. Resource contexts are identified and their appropriate OS handlers are modified to accommodate and apply the restrictions through policies. In access control systems based on environmental context, the operating environment of an entity decides on access policies. This environment can be a location, a time, available energy and network bandwidth for the operation of the device.

2.2. Role-based Context-aware Access Control

The role-based model uses roles to manage privileges. It is naturally applied to organizations where users are assigned roles with well-defined access control privileges. However, with the new requirements for ubiquitous applications, the basic RBAC have quickly shown its limits and several extensions have been developed to improve its security. As users are mobile and the number is large enough, the context becomes a factor of first order in access control.

Given the recognized success of RBAC approaches, several solutions have been proposed to enrich RBAC in order to support contextual constraints. In the work cited in [6], the authors use the RBAC model principle, so for each user equipped with an RFID card for identification and authentication is associated a role with predefined access rights but the context adds restrictions on these rights.

Zhang and Parshar [7] propose a DRBAC model (Dynamic RBAC), RBAC roles and model permissions dynamically adapted to the context. Each role is associated with a subset of a set of permissions and each subject is associated with a subset of a set of roles. DRBAC dynamically adjusts the authorization assignment as well as the role assignment based on contextual information of a subject. Contextual information could be any piece of information not just time or location.

Hansen et al. [8] propose an RBAC extension with spatial constraint SRBAC which takes into account the location of users when they require resources which could limit these permissions. In the SRBAC solution, roles / permissions are granted to a user in specific time intervals and / or if the user is in a particular location. Inclusion of the location constraint provides a mechanism to apply access based on location. Thus, a role is activated or deactivated if and only if a certain location constraint is satisfied. The location space is divided into multiple areas. An access authorization, is granted if the condition on the role is satisfied and the subject is in a specific area.

Other RBAC model extensions have been studied in the literature; in GEO-RBAC [9], RBAC is extended by including the positions of the user who would see his permissions vary, but also his role can also vary according to the connection area and resources location. The C-RBAC model (Context-RBAC) proposed by Park et al. [10], the Spatial-Temporal Role-Based Access Control Model ST-RBAC proposed by Ray et al. [11], etc.

2.3. Trust-Based Context-Aware Access Control

The authors, in [12], proposed a framework for retrieving and classifying contextual data from a mobile device and then deciding which access control to provide. They studied two use cases: Smartphone lock to prevent misuse, and defense against the sensory malware, where the user's private information is revealed to unauthorized ports. The authors used two contextual models: location to specify familiar places, using GPS to capture the user's significant locations in the outer areas of a building, for example, and using the wifi for capture significant areas of the user in interior areas; the social context to specify familiar people with the detection of smartphones surrounding the user. This defines the user’s trusted environment.

Cloud File is a personal cloud data access control is proposed in [13], it is based on evaluation of trust in mobile social networks. Social trust is used to protect and control access to personal mobile cloud data and storage using Key Policy–Attribute Based Encryption (KP-ABE). Trust can be evaluated based on the clue showed in mobile social networking. Types of social networking and communications are classified as: a) mobile voice calls; b) voice/video calls via mobile Internet (e.g., VoIP); c) short messages; d) instant social messages; e) pervasive interactions based on local connectivity. Social closeness and trust between two persons is evaluated using the number of voice calls (called and received), the number of interactions and the number of messages (sent and received).

In [14], a scheme using either a General Trust (GT) level issued by a core network or a Local Trust (LT) level evaluated by a device, or both, to control Device to Device (D2D) communication data access by applying Attribute-Based Encryption (ABE) is proposed. Only the devices holding the eligible trust level of GT and/or LT can access the data. Each user's equipment (UE) would select at least one kind trust level of GT or LT to secure communications. If the core network is available and a user device would control its data using GT only, then GT keys are used to encrypt and decrypt data. Otherwise, i.e. the core network is not available, LT-keys are generated for the allowed devices. When the core network is available and a user device would like to use both GT and LT to control its data access, the attribute keys are generated under the control of both GT and LT. Moreover, UE evaluates local trust levels with pseudonyms in order to enhance communication privacy.

TIRIAC (Trust-drIven RIsk-aware Access Control) framework [15] is proposed to enrich Grid access control services by adding an evaluation of trust and a risk management unit. A request may be permitted or denied according to the access policies and without consideration of risk.

But for the other risky accesses, risk policies and utility theory are used to process them. A risk manager evaluates the expected loss and benefit of the access request according to the characteristics of the involved resource as well as the subject's trust degree and confidence. Subsequently, the request may be denied, or extra risk mitigation obligations may be demanded.

2.4. Trust And Role Based Context-Aware Access Control

In [16], to redefine the role-based access control model and entities (users, roles, session, permission) using context, the authors analyzed context factors to classify them and to formalize them according to four system Security, User Confidence, Location and Time.

• The system security is defined in four levels PLT = {TL, HL, ML, JL} such that TL (top level) = 100%, HL (high level)> 80%, ML (middle level)> 50%, JL (junior level) <50%
• Confidence in the user, calculated from access history and usage for each resource in the system.
• Location: for each resource, a set of IP addresses are associated SC = {IP1,. . ., IPi} which represent the familiar locations
• Time: for each user, a set of time intervals are associated: TC = {T1, T2, TJ}.

The authors of [17] propose a Context-Aware Trust and Role-Based Access Control model CATRAC, for access control in composite web services. The assigned roles must be validated by

a third party (role authority) and the trust levels are vectors from 0 to 10 and the new clients have a level of 5. The trust is based on the user's access history to special resources.

In [18], a secure, automated PrBAC architecture and prototype system referred to as the Context-Aware System to Secure Enterprise Content (CASSEC) is introduced. It dealt with two proximity scenarios usually encountered in enterprise: Separation of Duty (SoD) and Absence of Other Users (AOU). A geo-spatial RBAC is used in a monitored space to localize persons using a wireless infrastructure. However, a malicious actor can grant privileges by manipulating the sensors in the monitored space, thus perverting control access decisions. To avoid this scenario, sentience-like constructs have been added to the geo-spatial RBAC, to emulate the confidence in the proximity evaluation as *degrees of reliability* in extracted context, thus allowing CASSEC to make more inferable decisions. Continuous authentication based on co-proximity is performed using PMs and users Bluetooth Low Energy (BLE) capabilities. To guarantee the safety of the client localization with a certain degree of confidence, despite multiple PMs detection, the system uses BLE beacons transmitted during this authentication. The role is constructed such that an access control policy is specified to grant with high confidence that the current devise user is the true owner of the device.

It is clear that context and context awareness must be included in any access control to a service. Access control can be based on context, role and trust and/or a combination of these. The solution we propose is in the last class.

3. PROPOSITION

With the needs of ubiquity and mobility, users can access anywhere and at any time to data from diverse and heterogeneous sources. Thus, access control must take into account the context of the user to manage access and preserve the confidentiality of data.

The concept of role is used as an intermediary between users and permissions. These are granted to roles activated by users during a session, following the RBAC model. However, new requirements for applications in ubiquitous environments are driving us to review and improve the access control system. Indeed, relying solely on the role to make a decision for an access request is not always convincing. In this respect, we add to this concept the notion of trust. Thus, each service will have a level of confidence that the user according to his assigned role can reach or have access to it.

But, the level of trust may be influenced by other types of contexts in this case, the temporal context and the spatial context being of prime importance. Indeed, when a user accesses the same service from two geographically distant locations during a short period of time it is understood that it is not the same person. The trust level becomes low and the system in this case must automatically lock access. The social context can also influence the system decision. If the user is in the right place at the right time but surrounded by unfamiliar people, his confidence level decreases, and access may be denied. Given that the concept of context is dynamic, taking into account the three contexts already mentioned would further strengthen the concept of access control.

We propose a system that establishes a trust relation in the user according to the role and the three contextual constraints and according to this trust the system decides to grant or refuse the request for access. Thus, our system will be classified in the fourth class.

3.1. Case Study

We present in the following our model applied to an access control at the level of a request for a bank service.

The banking system is an important element of the economic life of a country. Banks play a major role in the daily lives of households and businesses: ensuring the fluidity of transactions by providing economic agents with fast, convenient and secure means of payment. For this purpose, the aim of access control systems is to reduce the risk of interference with managed data.

Our work is to propose a system for controlling access to context-aware services. The example of the banking system to validate our proposal is adequate given its required high level of security. We propose a role-based access control, where access rights are assigned to users based on the role they play in the system, and based on trust, where the level of trust varies upon the user's history and context.

3.2. Operations

Users of a system are associated with a defined role in advance. When they want to perform an operation, they will activate a role during a session and they are supposed to be able to perform all the operations allowed by the role during this session, thanks to the privileges with which they are associated. In our example we define three roles: the client, the counter agent and the bank administrator. For each role we associate the following services (assuming that users are already authenticated):

A client can:
• Consult his account balance.
• Transfer money.
• Remove / deposit money on his account.
An agent can:
• Open / Close a client account.
• Make a transfer on the client behalf.
• Retrieve / deposit the client's money.
An administrator can:
• Open / Close a client account.
• Make a transfer on the client behalf.
• Retrieve/ deposit money from a client's account.
• Check transactions made during a day.
• Confirm opening of a new account of a client.

The services of the bank are classified into groups each having a minimum confidence threshold to authorize access, this confidence threshold is determined according to its sensitivity based on one or more contextual constraint(s). The user's access request is checked according to their level of confidence, the confidence value, is determined by:

- A client access history and his behaviour, this level of trust, is used to differentiate between a proven client and an untested client.
- As the client is mobile, hiss context is dynamic. Indeed the context influences the confidence and thus makes it variable, the level of trust as well, can increase or decrease depending on the contextual constraints.
  The level of service access requester confidence is determined in real time, according to the following set of contexts: location context, social context and time context.
- Location context

The location context is an important context in a ubiquitous environment. We note TL (Loc) is the degree of confidence in an access location. By hypothesis, home and workplace are familiar locations for a client or administrator. To determine the degree of access requester trust, table 1 is proposed:

Table 1. Description of Location Context Trust Level

TL(Loc)	Description
Level 0	If the user requests access from two locations very distant geographically and this in a very short period of time.
Level 1	If the user requesting access to the service is not in a location defined as familiar.
Level 2	in the case where a user requesting the service is in a location defined as familiar

• Social Context

We define the social context as being all connected people surrounding the user when the user accesses the service. Once a user logs in, the context manager detects people around him through their login device. All the people familiar with the user are defined when opening a bank account. TL (Social) is the level of trust in the user according to his social context. Table 2 determines the level of trust in the user according to his social context:

Table 2. Description of social context trust level

TL(Social)	Description
Level 0	In the case where the persons surrounding the access requester are foreign people (are not defined as familiar).
Level 1	In the case of the existence of at least one familiar person between those surrounding the access requester.
Level 2	If the user is surrounded only by familiar people.

• Context of time

The context of time is defined by the work hours of the requester of access to the service in the bank, in our case the context of time is taken into account if the role of the user, is an agent at the counter or an administrator. We note TL(Time) the trust level in user according to the context of time. These trust level of trust are defined in table 3:

Table 3. Description of trust level for context of time

TL(Time)	Description
Level 0	If the counter agent requests access to an out-of-hours working service.
Level 1	If the administrator requests access to the service outside work time.
Level 2	In the case the user requests access during working hours.

3.3. Trust Manager

After all types of context are identified as well as their trust levels, we proceed to determine the total value which is the value of the level of trust of an access requester based on the trust level of the different contexts. The value of the Total Trust Level (TTL) will be calculated as follows:

$$TTL = min(TL(Loc), TL(Social), TL(Time)) \quad (1)$$

where *min* is the function that returns the minimum value between these parameters.

3.4. Proposed System Architecture

Figure 1 presents the proposed architecture of the role and trust -based access control system. This implementation includes the following components:

• Context manager: Captures the context of the user to know its location, its social context and its time context, and stores it in a database "user's context". The context manager is not part of our work.
• User Context: Contains the user's contextual information. This context is dynamic, so when the context manager captures the same context with 3 different accesses of a user, this context is therefore defined as familiar and is updated by the context manager.
• Service Context: Contains the different contextual constraints for each service. Contexts for each service are defined according to each role as shown in Table 4.
• Role / permission: Contains all the permissions associated with user roles.
• Trust evaluator: Calculates the access requester's trust value based on his behavior after completing his operations and updating the reports.

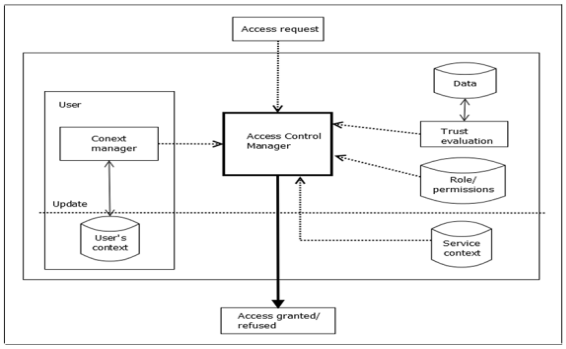

Figure 1. Proposed control access system architecture and operation

• Access control manager: When sending an access request, the access control manager retrieves the user's contexts from the context manager, the role from the database, the user's trust level from the trusted evaluator and finally the contextual constraints for each service registered in the service context database. Based on these information's, the access control manager decides whether or not to accept the access request.

Table 4. Contexts assigned to roles and services

Services \Role	**client**	**Agent at counter**	**Administrator**
	Context		
Account balance consulting	no context consideration	No access	No access
Make a transfer	Familiar location + surrounded by familiar persons	To be in the bank during working hours + client presence	Familiar location + surrounded by familiar persons
money retrieval	Familiar location + surrounded by familiar persons	To be in the bank during working hours + client presence	To be in the bank during working hours
money deposit	Familiar location + surrounded by familiar persons	To be in the bank during working hours	To be in the bank during working hours
Check transactions made during a day	No access	No access	Familiar location + surrounded by familiar persons
Confirm new account opening/closing	No access	No access	Familiar location + surrounded by familiar persons
Open a new account	No access	To be in the bank during working hours + client presence	Familiar location + surrounded by familiar persons

3.5. Case Study Scenarios

In this section we used this terminology and assume some hypothesis:

• U: User of a service whose access to the service is controlled.
• Client: The role is a client.
• Agent: The role of the user is an agent at the counter.
• ADmin: The role is an administrator.
• RR: Role of the user. Can take three values: Client, Agent, Admin.
• SRi: The set of services provided by the bank.
• TLU: System trust level in client. This trust level is calculated at each session and varies mainly according to parameters related to the client behavior. TLU is a real belonging to the interval [0, 0.5].
• TTL: The instant trust level of the system in the client. This trust level is calculated at each session and varies according to parameters related to the context.
• C: The user's trust level, it is the sum of the two TTL and TLU levels.
• STi: Service i threshold, each service has a minimum trust threshold from which the service can be provided to the user if the trust level TL is greater than or equal to STi.

• It is assumed that the user is already authenticated.
• Initially the level of trust of the service in user is equal to 0.3 (half of the interval).
• If the trust level (TLU) is zero then the user U is malicious. In this case, the user must approach the bank for a reset of his profile and verification of past transactions (client history).

In the following scenarios, we will consider that web services have in their databases user identifiers, their role, contextual constraints and trust levels.

3.5.1. Scenario 1: When The User's Role Is A Client

After being authenticated, the user sends an access request request to the banking service. The access control manager obtains user information including its role in the (role / Permission) database and its contextual information through the context manager and its trust level.

The system first checks whether its trust level according to behavior is different from level 0. In case the condition is not satisfied, access is denied. Otherwise, the system checks whether the client contextual information such as its location and the people around him at a given time, are appropriate for those existing in the database (service context). Following this audit, the trust level by location is determined from Table 4.1, and the social context trust level is determined from Table 4.2. Then the access control manager determines the client trust level as follows:

$$TTL = min(TL\ (Loc), TL\ (Social)) \qquad (2)$$

The total trust value is converted from level to a rate (weight) using the following:

• Level 0 = 0
• Level 1 = 0.33
• Level 2 = 0.5

The access control manager adds the two TTL and TLU values to obtain a trust level C which is compared with the requested service trust level STi. When the trust value C in a client is not lower than the confidence threshold STi predefined to a service SRi, the access is then granted to the client.

In the opposite case a subtraction is performed between the service confidence threshold STi and the new trust value C: DIF = STi - C. Access in this case is granted to the client if the result is less than 0.1, otherwise the request for access to the service is refused.

The following algorithm summarizes the access rules for the client.

Algorithm1 : Algorithm of processing a client service request

```
Service request (ID, SR)
SELECT role
WHERE ID=ID // retrieve the role according to the identity from the database
SELECT TLU
WHERE ID=ID // retrieve the trust level depending on the behavior.
if TLU≥0 then
        // retrieve client location and social context
        TTL=min(TL(Loc), TL(Social)) ;
        C=TTL+TLU ;
        if C≥ST then access granted ;
        else
                DIF←STi-C ;
                if DIF≤0.1 then access granted ;
                else access denied ;
                endif ;
        end if;
end if
end.
```

3.5.2. Scenario 2: When The User's Role Is The Counter Attendant

The agent at the desk supposed to be in the bank only during his working hours sends a request for access to the banking service. The access control manager obtains user information including its role in the (role / Permission) database and its contextual information through the context manager and its confidence level determined by the trust evaluator.

First, the system checks if its trust level, according to the client behaviour, is different from level 0. If it is the case, the access is refused. Otherwise, the system checks whether the contextual information of the user such as its location and the time context of his access request are appropriate to those existing in the database (service context). Following this verification, the trust level of the location is determined from Table 4.1 and the time context trust level is defined in Table 4.3. Then the access control manager determines the user's trust level as follows:

$$TTL = min(TL\ (Loc), TL(Time)) \qquad (3)$$

The total trust value is converted from level to a rate (weight) using the following:

- Level 0 = 0
- Level 1 = 0.33
- Level 2 = 0.5

The access control manager adds the two TTL and TLU values to obtain a trust level C which is compared to the requested service confidence threshold STi. When the trust value C in a user is not lower than the confidence threshold STi predefined to a service SRi, the access is then granted to the client.

In the opposite case, a subtraction is performed between the service confidence threshold STi and the new trust value C: DIF = STi - C. Access in this case is granted to the user if the result is less than 0.1, otherwise the request for access to the service is refused.

If the user is outside his working hours and / or outside the bank, his request for access is automatically refused.

The following algorithm summarizes the access rules for the counter attendant.

Algorithme2: Algorithm for processing the counter attendant service request

```
Requête de service (ID,SR)
SELECT role
WHERE Id=ID // retrieve the role according to the identity from the database
SELECT TLU
WHERE ID=ID // retrieve the trust level depending on the behavior.
if TLU≥0 then
        // retrieve client location and time context
        TTL=min(TL(Loc), TL(Time));
        C=TTL+TLU
        if C≥ST then access granted;
        else
                DIF←STi-C;
                if DIF≤0.1 then access granted;
                else access denied;
                endif
        endif
endif
end.
```

3.5.3. Scenario 3: When The User's Role Is An Administrator

An administrator, after being authenticated, sends an access request to the banking service. The access control manager obtains user information including its role in the (role / Permission) database and its contextual information through the context manager and its trust level.

As before, first, the system checks whether its trust, according to the behavior, is different from level 0, and if it is the case, the access is refused. Otherwise, the system checks whether the user's contextual information such as location, social context, and time context of the access request matches the existing information in the database (service context). According to this verification, the trust level of the location is determined according to Table 4.1, the social context trust level is defined in Table 4.2 and the time context trust level is defined in Table 4.3. Then the access control manager determines the user's trust level as follows:

$$NCT = min(NC\ (loc), NC(Social), NC\ (temps)) \qquad (4)$$

The total trust value is then converted from level to a rate (weight) using the following:

• Level 0 = 0
• Level 1 = 0.33
• Level 2 = 0.5

The access control manager adds the two TTL and TLU values to obtain the trust level C which he compares with the requested service confidence threshold STi. When the trust value C in a user is not lower than the confidence threshold STi predefined to a service SRi, the access is then granted to the user.

In the opposite case, a subtraction is performed between the service confidence threshold STi and the new trust value C: DIF = STi - C. Access in this case is granted to the user if the result is less than 0.1, otherwise the request for access to the service is refused.

An administrator is both a client and / or a counter attendant. He plays the role of a client when he confirms an account or checks transactions and acts as a counter attendant in other cases.
The following algorithm summarizes the access rules for the administrator.

Algorithme3: Algorithm for processing the administrator service request

```
Requete de service (ID, SR)
SELECT role
WHERE ID=ID // retrieve the role according to the identity from the database
SELECT TLU
WHERE ID=ID // retrieve the trust level depending on the behavior.
if  TLU≥0 then
        //retrieve client location, social and time context
        TTL=min(TL(Loc), TL(Social), TL(Time) ;
        C=TLU+TTL ;
        if C≥ST then access granted ; //
        else
                DIF←STi-C;
                if DIF≤0.1 then access granted;
                else access denied
                endif
        endif
endif
end
```

3.6. Generalization

In this paper we proposed a new access control approach for ubiquitous systems. In order to highlight the dynamic changes in the environment, the proposal, is based on the RBAC model and employs the notion of trust evaluated by measuring environmental context and social context. When the access requester trust value is not less than the predefined trust threshold, the user can then execute the permissions associated with his role. In this way we retain the administrative benefits of RBAC and at the same time mitigate the inflect ability and static nature of the RBAC by exploiting dynamism through context awareness in the access control decision.

Our context-aware access control approach for banking services has major implications for other context-aware services. For example, it treats contextual attributes as access decision parameters

to provide effective and more appropriate security. In addition to location and time, we have taken into account other context that can be used in other context-aware services such as health services, education services and the military.

The proposal is valid for any system where users are identified and the three contexts are paramount. If a system does not meet so many contexts the case study has a situation where two contexts are taken into account. Our example is based on a banking system, but the general architecture can be applied to other systems that have similar context constraints.
The access authorization process is summarized below:

Step 01: After a session is started, a user requests access to a service.

Step 02: The access control manager retrieves the role of the user and the trust value in the database (Role / permission) and in trust Evaluator respectively.

Step 03: This manager verifies the level of trust if it is greater than 0, otherwise access is denied.

Step 04: It calculates the trust value according to the context after the user's context recovery and the service context.

Step 05: The access control manager adds up the two trust values, if the result is greater than or equal to the confidence level then access is granted. Otherwise a subtraction is made between the confidence level and the trust value, if the result is less than 0.1 then access is granted, otherwise access is denied.

The general operation of the access control process is illustrated by the flowchart of figure 2.

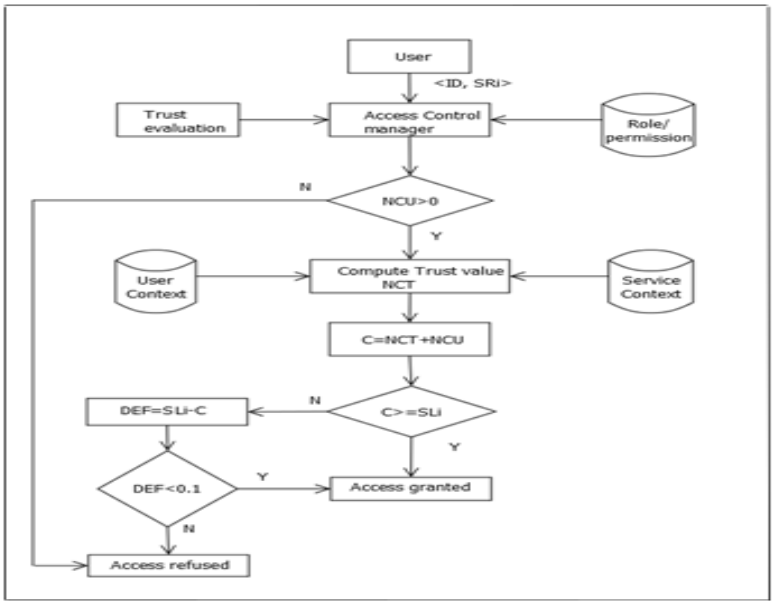

Figure 2. The access control process

4. SYNTHESIS

Our proposal retains the idea of considering the context parameters because it will allow us to have a dynamic access control system, context aware and which adapts the permissions according to current context. We also retain the notion of trust for our system, where we will establish a relationship of trust with the user according to the contextual constraints and it is with respect to this trust that we will decide to grant or refuse access.

In order to offer a better adapted access control, we include other concepts. Indeed, in addition to the concept of predefined role three types of contexts will be used to enrich the access control system with context-aware access control. The contexts taken into account in our approach are the temporal context, location and social context. Enriching an access control system with different types of contexts makes it more adaptable for any situation and also more flexible.

1) Context-aware access control models:

In [4], only time and location is taken into account in access control. Quality of context is an interesting parameter to be considered in the future, and CoDRA [5] takes into account time and location, but we think that energy and connectivity are not useful for granting or denying access.

2) Role-based context-aware access control models:

RBAC only proposes the consideration of functional roles in access control, but not contextual roles, even if they evoke the usefulness of temporal authorizations. Note that this allows ordering the roles between them, but not to model constraints involving the time explicitly compared to the proposed model where the time context is taken into account. In other words, our approach essentially implements context awareness. In this way, we retain the administrative benefits of RBAC and, at the same time, mitigate the inflexibility and static nature of RBAC by the dynamism exploited through context awareness in access control decision making.

In the SRBAC solution [8], roles / permissions are granted to a user in specific time intervals and / or if the user is in a particular location. But SRBAC does not support the use of a defined role in a limited geographic space. On the other hand, our proposed model allows it thanks to the determination of the familiar places for example, the workplace, the house and the bank in our case study. DRBAC [7] dynamically adjusts the authorization assignment as well as the role assignment based on contextual information of a subject. DRBAC does not really meet our requirements for a context-aware access control solution. But once the roles are determined, which is the case in our proposal, significantly reduces the complexity of managing security.
The rest of the cited models [6, 9, 10, 11] have as inconvenient the fact that only time and location are used to define context. We think that social feature is an important context parameter in access control.

3) Trust-based context-aware access control models:

ConXsense [12] defines familiar locations as user's trusted environment but do not manage trust. Trust is a dynamic relative parameter, and CloudFile [13] considers social context through social networks, but trust is determined only on this social context. In [14], the author's contribution is

to consider two trust levels based only on communications behaviour. The trust management is complex and not refined.

TIRIAC [15] takes into account risky accesses and trust is given degrees of confidence which enriches the access control. The advantage of TIRIAC is that it is a framework for grid access control and so the level of security must be as refined as that.

4) Hybrid access control based on role, trust and context awareness models:

The proposed solution belongs to this class, where the models are complete. In[16], the authors consider confidence in the user (trust) as context information and so manage context information to decide on access control. The two managers, trust and context, should be separated for more strongness. CATRAC [17] uses a role authority to assign roles which is not very relevant in our case (roles are fixed), and levels of trust based on user's access history, which limits context. CASSEC 2.0 [18] extends a geo-spatial RBAC by adding degrees of reliability (or trust) in the contextual information to confirm the user's location. Other context features would have been necessary.

Table 4. summarizes the previous comparison based on context role and trust.

Table 4. Context-aware access models comparison

Model	Role	Time	Location	Social	Trust	Other
S-RBAC [8]	X	X	X			
GEO-RBAC [9]	X	X	X			
ST-RBAC [11]	X	X	X			
D-RBAC [7]	X	X	X			
C-RBAC [10]	X	X	X			
RFID-RBAC [6]	X	X	X			
Fine-grained Access Control [16]	X	X	X		X	
CATRAC [17]	X	X	X		X	
ConXsense [12]			X	X	X	
Context Quality-Aware [4]		X	X			Quality of context
General Trust/Local Trust [14]					X	
CoDRA [5]		X	X			Energy/ Bandwidth
TIRIAC [15]			X		X	Risk management
CASSEC 2.0 [18]	X		X		X	Confidence specifiers
CloudFile [13]				X	X	
Proposed approach	X	X	X	X	X	

As mentioned earlier, previous systems check if the client has the proper role to access a particular web service. However, the verification of the client reliability gives more assurance to the provider. Our work has similarities to trust-based access control models. Indeed, the trust level of the client is verified during each access attempt to ensure that it is high enough to access the requested service while being based on contextual constraints. This is what makes our approach more efficient and reliable.

Our model has the following advantages:

- In addition to role concept, a set of three contextual attributes: Time, Location and Social has been taken into account.
- The presence of a correlation between the three contexts.
- Trust is dynamic.
- The role-based policy analysis concludes that they are relatively easy to administer and flexible enough to adapt to each organization.

5. CONCLUSIONS

Using context information is an asset for creating access control models. We focused on context-aware access control in ubiquitous systems. Entities operate in environments that are dynamic and unpredictable, forcing them to be able to guarantee security. In particular, banking systems are complex, feature-rich systems that are becoming more and more demanding in terms of security: confidentiality, integrity, availability and accountability. Therefore, it is essential to first define a security policy that is robust, efficient, flexible, generic and easy to verify.

Our context-aware access control model is based on a cross-use of role; contextual constraints and trust, as flexible enough structuring tools that complement the gaps presented by some models. In the same way that the role binds the users to the privileges, the trust according to the contextual constraints makes it possible to establish a relation between the operations and the objects to be protected. We also frame the concept of context awareness to allow access control respecting the principle of least privilege while ensuring flexibility favoring the user's profit. We have detailed the operation of our system by applying it on a bank where we presented a case study with service, entities, roles and contextual constraints for each service and entity.

Like any research, our model has some limitations. The main thing is that we have not been able, yet, to perform a simulation to highlight the real benefits of this mechanism. So, it would be opportune to pursue research along several lines. First, it would be wise to study the integration of a mechanism to deal with attacks. Indeed, some attacks aim to oppose traditional systems of blocking or protection. It is thus necessary to be able to quickly detect the intrusion to reduce the damage which can be caused by the hacker or the steal of data that it can realize. So the proposed access control system should deflect known attacks, detect ongoing attacks, including those coming from within, and react quickly.

Then, in order to be able to test our mechanism in real size, it is necessary to have realistic context information, through context simulators and also direct captures on a physical environment. The next step is to create a realistic map representing the environment of our case study, in other words, to reproduce a realistic plan of a banking space, then to:

- Simulate the behavior of clients accessing a banking service and carrying out different activities.

- Periodically retrieve the contextual information (Time, location and people around) from the user and send this information periodically to the context manager.

- Implement and deploy a prototype of the proposal in the appropriate environment.

REFERENCES

[1] Weiser, Mark (1999) *Some computer science issues in ubiquitous computing*, Mobility: Processes, Computers, and Agents, ACM Press/Addison-Wesley, NY, pp420-430.

[2] Jemili, S. (2013) *Analyse de risqué dans les systèmes de contrôle d'accès*, Master report, University of Quebec en Outaouais, Canada.

[3] Ferraiolo, D. F, J. A. Cugini, & O. H. Kuhn (1995) "Role-Based Access Control (RBAC) : Features and Motivations". Proceddings of 11th Annual Conference on *Computer Security Applications* pp 241-248.

[4] Filho, J. B. & H. Martin (2008) "A quality-aware context-based access control model for ubiquitous applications", Proceedings of International Conference on *Defects in Insulating Materials (ICDIM)*, Aracaju, Brazil, pp 113-118.

[5] Thanigaivelan, N. K., E. Nigussie, A. Hakkala, S. Virtanen & J. Isoaho (2018) "CoDRA: Context-based dynamically reconfigurable access control system for Android", Journal for *Network and Computer Applications*, Vol. 101, pp1-17.

[6] Khan, M. F. F. & K. Sakamura (2012) "Context-Awareness : Exploring the Imperative Shared Context of Security and Ubiquitous Computing", Proceedings of the 14th International Conference on *Information Integration and Web-based Applications & Services*, Bali, Indonesia, pp101-111.

[7] Zhang, G. & M. Parashar, (2003) "Dynamic Context-aware Access Control for Grid Applications", Proceedings of 4th international Workshop on *Grid Computing,* Washington, USA., pp101-108.

[8] Hansen, F. & V. Oleshchuk (2003) "SRBAC : A Spatial Role-Based Access Control Model for Mobile Systems ", Proccedings of 7th Nordic Workshop on *Secure IT, Systems*, Gjvik, Norvège, pp 136-152

[9] Bertino, E., B. Catania, M. Damiani, & P. Perlasca (2005) "GEO-RBAC : A Spatially Aware RBAC", Proceedings of the tenth ACM symposium on *Access control models and technologies*, Stockholm, Sweden, pp 29-37.

[10] Park , S.H, Y. J. Han & T. M. Chung (2006) "Context-role based access control for context-aware application", In: Gerndt M., Kranzlmüller D. (eds) High Performance Computing and Communications. HPCC vol. 4208.

[11] Ray, I.. & M. Toahchoodee (2007) "A spatio-temporal role-based access control model", Proccedings of 21st Annual International Federation for *Information Processing Working Group*, pp211-226.

[12] Miettinen, M., S. Heuer, W. Kronz, A.-R. Sadeghi & N. Asokan (2014) "ConXsense Automated Context Classification for Context-Aware Access Control", Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security, Kyoto, Japan, pp293-304.

[13] Yan Z. & W. Shi (2017) "CloudFile: A cloud data access control system based on mobile social trust", Journal of *Network and Computer Applications*, vol. 86, pp46-58

[14] Yan, Z., H. Xie, P. Zhang & B. B. Gupta (2018) "Flexible data access control in D2D communications", *Future generation computer systems*, pp738-751.

[15] Noggorani, S. D. & R. Jalili (2016) "TIRIAC: A trust-driven risk-aware access control framework for Grid environments", *Future generation computer systems*, Vol. 55, pp238-254.

[16] Hong-Yue, L., D. Miao-Lei & Y. Wei-Dong & (2012) "A Context-aware Fine-grained Access Control Model", Proccedings of 21st the International Conference on *Computer Science and Service System*, pp1099-1102.

[17] Ghali, C., A. Chehab & A. Kayssi (2010) "CATRAC : Context-Aware Trust- and Role-Based Access Control for Composite Web Services", Proceedings of 10th International Conference on *Computer and Information Technology (CIT)*, pp1085-1089.

[18] Oluwatimi, O., M. L. Damiani, E. Bertino (2018) "A context-aware system to secure enterprise content: Incorporating reliability specifiers", *Computers & Security*, vol. 77, pp162-178.