Adaptive Context-Aware Access Control Model for Ubiquitous Learning Environment

Jagadamba G[1] and B Sathish Babu[2]

Submitted in April, 2014; Accepted in July, 2015

Abstract – In a ubiquitous environment the users access to any services at anytime, anywhere through any device is the new dictum. Thus, ubiquity and mobility of devices made the access control, adaptive in nature by using the contextual information. However, dynamically changing context does not leverage on access control for the resources requested. We propose an access control mechanism that adapted through means of gathering the dynamically changing contextual information that has an impact on access decisions. As a result, a fine- grained access control decisions is assessed through well-tuned analysis about a user behavior and need before granting or denying. Results and performance analysis is presented for the proposed context-aware access control mechanism.

Index Terms – Access Control, Authentication, Context-Aware, Security, Ubiquitous Environment.

NOMENCLATURE

ASCII- American Standard Code for Information Interchange, RBAC-Role Based Access Control, CAAC- Context-Aware Access Control, CS-Checksum, IP-Internet Protocol.

1.0 INTRODUCTION

Context is defined as, the information that is utilized to characterize the situation of an entity. Intuitively, it is reasonable to accept location, identity, activity and time [1, 2] as essential attributes of the context. In a ubiquitous environment, the context-awareness term was introduced by Schilit in 1994[3]. The Context-aware devices [4, 8] intellectually capture the circumstances under which they operate, based on rules, situations, requirements and react accordingly to their environment. Thus, ubiquitous computing personalizes services to the end users based on context.

Manipulative access to resources is usually done by listing access control rules or policies. Adaptability in access control consistently requires evolving change and requirements in a ubiquitous environment. However, the security related challenges, like authentication, authorization, access control, integrity, confidentiality, and availability needs to meet in any context-aware system. In our proposed system, we concentrate on authentication and authorization by considering the required data for access control. Authentication verifies the identity of an entity, i.e., what you know, what you are or what you have.

[1,2] Dept. of Comp. Science and Eng., [1,2] Siddaganga Institute of Technology, Tumakuru, Karnataka, India.
[1] jagadambasu@gmail.com, [2] bsbsit@gmail.com.

Passwords, passphrases, secret codes, certificate based [5]and personal identification numbers (PINs) found as what you know; keys to lock and unlock for what you have and biometric authentication methods like iris, image, fingerprints, and keystroke [6], presents what you are. Once the identity of the user gets declared in the authentication stage, the users were assigned a set of authorizations considered to be the rights, privileges, or permissions associated to do with the resources. The system administrator assigns the authorization by considering the system security policies. The policies define the right varying from the boundaries of not allowing anything (permit nothing) to allowing everything (permit everything) or to allowing in between [7]. In our proposed system, the user types are found accordingly to the change in context. The change in context and behavior is analyzed while fulfilling the need if the request comes from the genuine user.

1.1 Ubiquitous learning

Learning about the right thing at the right place and time in the right way is needed in ubiquitous computing environment [9]. Context-aware Ubiquitous environments adapt the user's real situation to provide adequate information for learning. Ubiquitous learning [10] creates a situation or surroundings to acquire the context information all around the user, where he/she may not be conscious of the learning developments. Developments in ubiquitous learning environment adds the advantages of adaptive learning with the benefits of ubiquitous to provide users, freedom to access to their individual needs with allowable flexibility.

1.2 Proposed Adaptive Context-aware Access Control

Learning based on user requirement, becomes necessary for access control based on the role the user is playing in a ubiquitous learning environment. Where a user can play a single or multiple roles at a time, but computing system in this needs a generous level of understanding of the situation they are and the complex relations between the various essentials. Thus, the ability becomes perception and malleable to the situation accordingly termed as "adaptability". Adaptive context-aware access control knowledge includes: monitoring users activity, understanding user's requirements and preferences, and using these newly gained information to facilitate access control.

The proposed Context-Aware Access Control (CAAC) adapts itself to the varying environment by developing a proactive centralized monitoring system that acts as a context server through acquiring and managing the context information for analyzing user requirements. The analyzed output will bring in the corresponding decisions.

This paper is organized as follows: Section 2, discuss related works in the area of context-aware models, section 3 describes the proposed CAAC system, section 4 discusses the performance analysis, section 5 presents the results and section 6 concludes with future work.

2.0 RELATED WORKS

Access control models were categorized to [11]: mandatory access control models, discretionary access control model, and role-based access control model. A role-based access control models allow users, computers, and applications to follow static policy for accessing computers, files, applications, servers, communication ports and devices.

The Role-based Access Control (RBAC) [12] started with multi-user and multi-application on-line systems pioneered in the 1970s. The RBAC grants the access permissions based on the roles and various job functions the user is playing in an organization. All RBAC models restrict the access control through static policies. But, the model [13] allows for the definition of context-aware security policies for requested tasks and makes it easy to define, and understand complex security systems. An effective access control model that is aware of context modifications and change authorizations when location, date, time of access, and resources settings changed in Context-Aware Access Control Model (CAACM) [14]. Cerberus [15] included context-aware identification, authentication and access control and reasoning about context, but the process was complicated to implement. Generic context-based software architecture (Gaia) [16] was aware of physical spaces based on geographic region with limited and well-defined borders, containing physical objects, assorted networked devices, and users performing a varied of activities. This model uses the context of first-order logic and Boolean algebra, which allowed them to describe straight forwarded rules. The Kerberos authentication [17] proposed the process to enable activation or deactivation of roles assigned to a user depending on his/her context. If we consider a user, in an un-secure place like a canteen, the access to sensitive data is not allowed, but when the user is in a research lab, private chambers or hostel building rooms is permitted to access the confidential data. The system [17] is context aware in nature to block the accessing if the user is in unsecured context or allow access if the user is in a secure context.

Context-aware access control based on ontology [18] was aware of developing the policies based on the user, device, and place for software services that are static in nature. In [19] the context-aware access control policy T consists of two parts: the context attributes set TS = {teacher/student, teaching time/spare time, Pad/mobile phone/personal comp.} and the operation attributes set TO = {Read/Write/Read-write}, namely T = {TS AND TO}. The context attributes set TS consists of four sub-attributes, which denotes user uses the device at a particular time in a place, namely TS = {Who AND When AND Where AND Which}. For example, for the resource R, following policy is established to control access:

Tro = {{Teacher OR Student} AND {School Teaching OR Spare Time} AND {At School OR Outside School} AND Only}.{Personal Comp. OR Pad OR Mobile Phone} AND Read Trw = {Teacher AND Spare Time AND At School AND Personal Comp. AND Read Write}.

The policy Tro represents the context condition under which the resource R is allowed to read-only mode. The policy Trw defines the context condition under which the resource R is operated in the read/write mode. This is almost acceptable with the [14]. Another system [21] adopts access control based on a need-to-know principle, where the requests for access are allowed when relevance to the requester's project is identified. For example, if we consider a data analyst's present the project development of a mechanical project, it would be illegitimate for the analyst to have access to documents on other aspects, e.g., feminist activities. Above examples contributed to move towards adapting to the situation and need. Adopting the user contexts [14, 19] to his need motivated to implement an adaptive access control according to the changing context.

3.0 PROPOSED SYSTEM

This section describes the architecture and a working model of the proposed system.

3.1 Architecture

Centralized approach or decentralized (distributed) method can follow while building context-aware access control system. However, selection of access control method is based on ubiquitous environmental requirements and associated risks. We adopt a centralized approach for access control model for the users in small ubiquitous environments like a college campus, centralized offices, restaurants, and malls. The architecture of the proposed CAAC model is given in Fig.1.

The architecture comprises of three units: Device Unit, Validation Unit, and Allocator Unit. The Device Unit is the user's device or user registered in the central system maintained on the academic campus. Validation Unit is responsible for authentication and authorization of a user. Allocator Unit manages required credentials of context and context processing for accessing the higher priority resources for the individual user.

The proposed system uses the following types of profiles [15]:

Personal Profile: It contains data about the user like name, User_Id, password, designation or Usertype, department, mail-Id and address uploaded while registering for the system. These details are used to define the user type while authentication and authorization. For example, usertype may be a student, researcher, guest, teaching faculty or administrative officers. A database in the Allocator unit stores the personal profile in the structure way as per the application requirement.

Explicit Profile: This contains clear user data collected from past interactions. When authorization change is required, this profile is used for customization and synchronized for future analysis. For example a User_Id/username, password, accessed web resources, log details, mobility to the user and device, time

and date of the log and last level defined. The logger of *Allocator unit* stores the exact profile data.

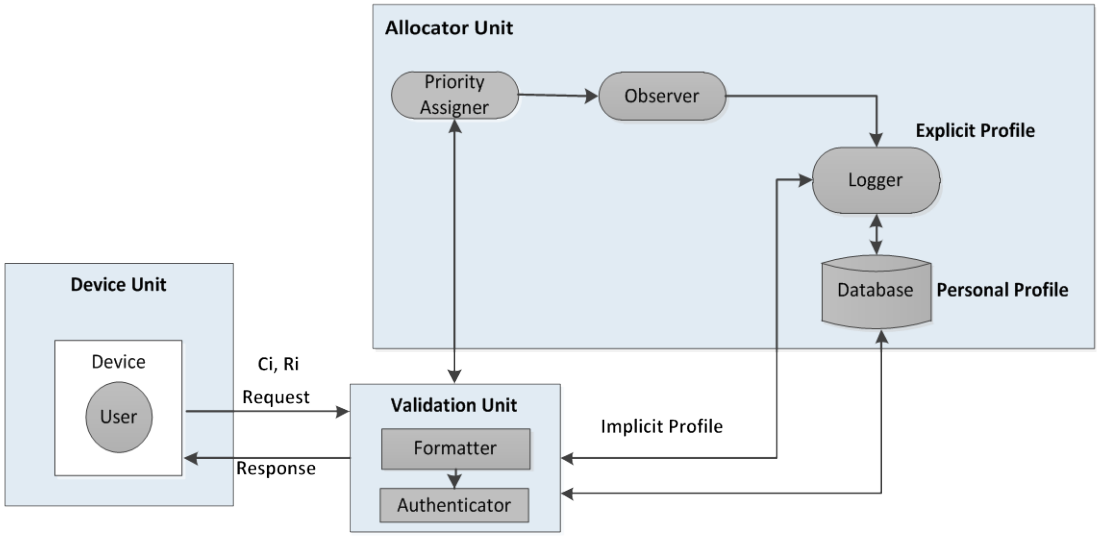

Figure1: Architecture of CAAC model

Implicit Profile: It is the set of all data objects within a system processed by user profile and explicit user profile data of users for new validations sent on to *Validation unit*.

3.2 Working Model of the proposed CAAC

This section describes the working of the proposed CAAC model. The working of CAAC is enforced through identification of the subject before granting or denying access to the object. For example, user U authenticates himself to a CAAC system for identification. After the user identification, requests are verified to access resource R. The computing system verifies the privilege before the permission P is granted for the user U for accessing resource R. If the privilege of accessing the resource R for user U is not included in the access policy set, permission is granted or denied only after processing of contextual information C about a user. The Algorithm.1 depicts the operation of CAAC.

3.2.1 Device Unit

As mentioned earlier, this unit defines the user device and user who access the resource. The authentication of the user is delivered from the device the user is using on the campus that has been already registered in the central system.

3.2.2 Validation Unit

Validation Unit is responsible for authenticating and authorizing the user by obtaining necessary data from the *Allocator Unit*. The *Validation Unit* consists of *Authenticator* and *Formatter* for the identification of an individual.

Algorithm 1

1. **INPUT: Access to web resources**
2. **OUTPUT: Access or deny**
3. **Start: Request for resource access**
4. **if: username and password are verified**
5. **allow authorized resources**
6. **else reject as unauthenticated user**
7. **while: access request is for higher privilege resource**
8. **do: training for access control by collecting contextual information about the user**
9. **Set a window size T for observation**
10. **if: logger has supporting data about a user**
11. **Threshold levels are verified**
12. **Implicit data from a database is set and sent to validation unit**
13. **if received data conform to extend an authorization**
14. **allow the user to access**
15. **else: deny by rejecting the request**
16. **end if**
17. **end if**
18. **end while**
19. **end if**
20. **END**

Authenticator: Authenticator verifies the user identity whenever a request arrives from Device unit. The user-Id and password is sent to the database in Allocator unit for confirming the identity. In our proposal, user U is associated

with the user type defined permanently at the time of registration. The user is authorized to access the resources coming under his privilege.

The User_Id identify the usertype and associated privilege p. The usertype for a user U belonging to five categories with privilege level p from 1 to 5 given in Table.1 are expected from eqn.(1). For example, if User_Id is a registration number of a student starting from 1 to 1000, then the User_Id associates the privilege p1 to define the Usertype-1 or if the User_Id is an Employee_Id starting from 1000 to 2000, it associates the privilege p2 as Usertype 2.

U_{id}=User type*P_i ……(1)

Formatter: *Formatter* is used to convert the heterogeneous context information into predefined ASCII values. These representations are defined as checksums and the conversions are done for the processing of data values for training. Checksums are scaled between 0-25 in the *Priority Assigner* and *Observer*, discussed in detail in section 3.2.3.

3.2.3 Allocator Unit

The whole process of collecting the contextual information for training and modifying of the usertype requires the participation of *the Database, Priority Assigner, Observer and Logger.*

Database: The database is used for storing all the relevant user data and threshold values for authentication and authorization. If the resources are found increasing, a resource optimization using XML can be opted [20]. A query for authentication and authorization are presented in <Username, hash_Password> and <Username, Resources> format. The password is hashed and stored to avoid the malicious hand attack on the database.

Priority Assigner: It assigns the priority for different contextual information. This contextual information is retrieved from the user device from the repositories either in a pull or push mode. In a pull mode, the context-aware system explicitly requests for context information from GPS systems periodically, or GPS system can drive the contextual information whenever a new location is discovered. The extracted context data are transferred to the *logger* for further processing of the profile data mentioned in Sec. 3.1. Earlier to this contextual information is assigned a static priority level ranging from 1 to five as shown in Table 2. Where 1 represents the lesser priority level and 5 represents the higher priority level. The priority levels may change with time depending on user behavior. For example, if a PG student spends lots of time in a lawn rather than in a research lab or classroom and trying to access the research related resources. Subsequently, user priority level for lawn changes from 1 to 2 to that user.

Observer: The *observer* gets the time interval and window size for training the user. The window size is used to verify the allowable time for observation before changing the authorization. The defined static Usertype is modified from eqn.(1) according to the requirements of the user by putting him/her to training/observation for a window size of T_o. The Window size is defined [22] for each user for a time interval t_i from eqn.(2). For a number of requests x_n, the proportional time interval t_i is found for i values from 1 to 1440 min(24*60) from eqn.(2). Eqn.(3) provides required window size for the user U_i and this remains to be predetermined forever for N number of access requests. A total number of five Usertypes and resource type were considered for the experimental purpose.

$t_i = x_n/\min$ ..... (2)

User Type	Users	Privilege	
		Sites	Download capacity
1	Undergraduate students	Education related sites	2Mbps
2	Postgraduate students	Resources of the first level with technical sites	4Mbps
3	Faculties & Research Students	Resources of the second level with scientific related resources	6Mbps
4	Head of the dept., deans, placement officers	Resources of third level with commercial resources	8Mbps
5	Administrative officer or head of the institute	No restriction	10Mbps

Table 1: Statically defined User type and privilege levels

$T_o = t_i * (N/5)$……. (3)

Logger: This verifies and stores the new access permission for the user. As mentioned in the Observer section, the training is adopted when an unauthorized resource request is made. Logger performs the process of training by gathering the contextual information through the following steps:

- Gather priority for each context attribute from assigner.
- Threshold fixation for accessing each resource through data analysis, which acts as the restriction parameter for the unauthorized access.

Table 2: priority levels defined for Context information

In the process of registration the CAAC model, the every Usertype is defined with the set of resources in his/her privilege. In the process of training the new resource is allotted by calculating the dynamic threshold T to decide the allowable resources. The threshold for the new resource (0 , 1), where 0 represents resource with high priority, and 1 represents a resource with low priority. The allowable upper threshold UT ($\mu + \sigma$) and lower threshold LT ($\mu - \sigma$) are evaluated through the standard deviation and mean from eqn.(4) and eqn.(5) respectively. The y_i is the captured value about the particular resource among N types. These values supports to take the access decision on resources. The context data are validated by the checksum value of eqn.(6) using the eqn.(7). Where the eqn.(7) gets the checksum value for the serving resources (R1, R2, ... Rn) at the priority level (P1, P2,..... Pn).

$$\mu = \frac{\sum_{i=1}^{n} y_i}{N} \quad \text{..... (4)}$$

$$\frac{\sigma = \sqrt{\sum_i (Y_i - \mu)^2}}{N} \quad \text{..... (5)}$$

$$CS_{CD} = \text{User type} \times \sum_{n=1}^{4} C_n \quad \text{.....(6)}$$

$$CS_{Ri} = P_i \times R_i \times CS_{CD} \quad \text{..... (7)}$$

$$CS_{T_o} = t_i \times \text{Usertype} \quad \text{..... (8)}$$

The overall checksum is found by the formatter through eqn.(6) and eqn.(8) for the user U. A final checksum CS_f is considered from eqn.(9) for promoting the user to a higher level. Where CS_{Ri} is the checksum value of resource R_i and CS_{To} is the checksum value of the threshold values of the contextual analysis. Thus, a new Usertype is authenticated in the Validation unit with allowable threshold and forwards the access permission without any interruption or interaction with the user, otherwise Validation unit responses with a suitable message.

$$CS_f = CS_{Ri} + CS_{To} \quad \text{..... (9)}$$

Before we proceed further, let us consider a case study below to build the required scenario for our proposed CAAC model.

Let us consider the user is an undergraduate student trying to download software or videos from the project lab on the date Dec. 25th. But according to the access control policy, the student belongs to the Usertype-1 and don't have the privilege of obtaining commercial resources. At this situation, the context-aware system extracts the contextual information related to the user and requesting resource. The system identifies the location of the user as project lab through an IP or MAC. The assignor has a priority level of 4 out of 5 for the project lab, and date has a priority value of 4 out of 5. The combination of these context checksum values provides proper values to upgrade the Usertype to higher levels as per the allowable threshold values in the database. On the same, logger collects the past data regarding the involvement of the user in the research or development activities. After identifying these, training the request for some time interval is required for better analysis. Completion of all these learning produces a change in Usertype for the temporary session. Once the session is completed or if he/she becomes idle, the system is attentive to find that the user is not active. Thus, the new Usertype is discarded and predefined Usertype is allocated.

Keeping the above scenario in mind, we consider C1, C2, C3 and C4 as context data related to the user while requesting for the resource. Where C1 may be the location extracted and C2 may be the date and time of the request, C3 is past access level provided, and C4 is the resource requested. The combination of these contexts verifies the considerable change in the Usertype. From CS_{CD}, we find the level of change required, and every incident of change in Usertype are stored in the logger as recent historical data.

4.0 PERFORMANCE ANALYSIS

Every model, system or mechanisms are analyzed for its performance with respect to various aspects concerning its application. In this section, we discuss a statistical foundation by considering a performance cost as a quality metric for performance analysis of context-aware access control model.

Average Access Delay: The adaptive access control models bring in more the performance cost compare to traditional models [19] as new procedures are considered for access decisions. The investigation of the level of performance value is measured with a quality metric called Average Access delay (AAD). AAD is the difference between the time of access request sent and the time of access control decision made.

AAD in milliseconds for n access request made by a user *U* on the same resource *R* is considered. Each access request varies with the context; priority levels (1, 2, 3, 4 and 5) but operates on the same attribute cardinality (i.e. after 3). The overall ADD was assumed to be about 0.2% higher in sensible context levels 4 and 5 in Table.1.

Contextual data	Type	Level of Priority
Location	Canteen ,Recreation club, stadium and lawn	01
	Classrooms, Laboratory	02
	Faculty Chambers, research lab and Project lab	03
	Heads and Dean Chamber	04
	Principal office, Placements office and administration office	05
Date	August and February	01
	September and March	02
	October, November, April, and May	03
	December and June	04
	July and January	05
Resources	Educational	01
	Educational and Technical	02
	Educational, Technical and Scientific	03
	Educational, Technical, Scientific and Commercial	04
	Educational, Technical, Scientific, Commercial and Entertainment	05

Sensible Iterations: To ensure the statistical significance of the experimental result in sec.5 we identified the AAD for multiple iterations *r*. A particular value of *r*, at which the unpredictability tends to disappear is selected and used subsequent with the access control delay factor. In accordance,

a larger value is considered, and it leads to the less effect of randomness on the AADs. So *r* from eqn.(10) is satisfied to reach the precision level for the effect.

$$r \geq \left(\frac{t_{\alpha/2} \times S}{\varepsilon}\right)^2 \quad \text{.....(10)}$$

For small values of *r*, $t_{\alpha/2}$ is used, as t_α is the student's t-distribution. Where *S* is the standard deviation and ε is the specified error.

As sample size *r* increases, the distribution became just about customary to normal standard distribution and found for a population of unknown mean μ and an unknown standard deviation. A student distribution $t_{\alpha/2}$ is computed with *(r-1)* degree of freedom from eqn.(11) for *n = (3, 4,…..12)*. The n = *12* is predicted to be appropriate which is almost equal to the normal standard deviation.

$$t = (\bar{x} - \mu)/(\frac{S}{\sqrt{n}}) \quad \text{.....(11)}$$

5.0 RESULTS

The testing of CAAC model includes different parameters of its subject to evaluate the system efficiency by discovering the AAD.

Table 3: Time required for processing i/p & o/p data

Unit	Input	Output	Time(ms)	
			RBAC	CAAC
Formatter Unit	UC	FC	15	15
Allocator Unit-Observer	FC	Flag	-	16
Allocator Unit- Priority Assigner	FC	FC	21	21
Allocator Unit- Logger	FC, UC	Void	-	16
UC –Unformatted Context information, FC– Formatted Context information				

The addition of context in access control always attaches additional burden on computation, intern increasing the performance cost. Hence, evaluated the time required to process i/p request to access resources for our proposed model by considering the context attributes for a user with the RBAC model which is based on the roles the user. We found that 32ms of extra delay was introduced due to the consideration of context information shown in Table.3. In the same form, three sample cases for 2000, 5000 and 10000 access requests are considered for observing the deviation in the performance cost measured in msec. The observation was done from Fig.2, that higher performance cost for 2000 access request, an increase of 0.2 % for 5000 access requests and the normal range for 10,000 access requests. The proposal extends only by 0.2% for n number of access requests for different contextual information.

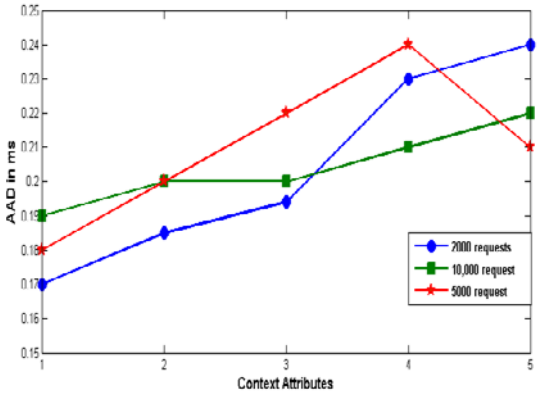

Figure 2: Context attributes v/s number of requests

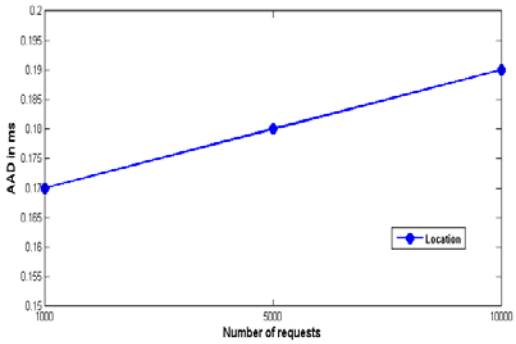

Figure 3: AAD for the location attributes

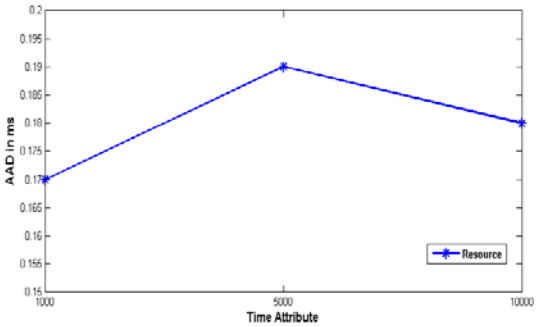

Figure 4: AAD for resource attributes

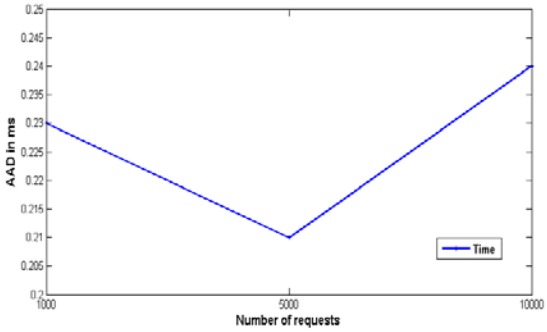

Figure .5 AAD for time attributes

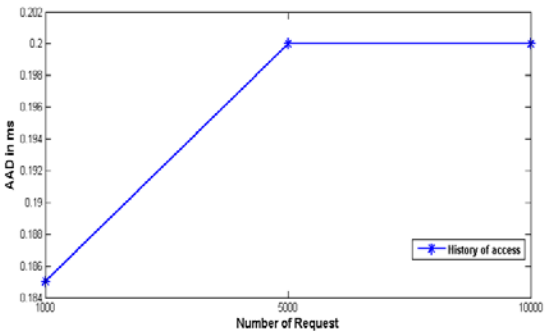

Figure 6: AAD for history of access attributes

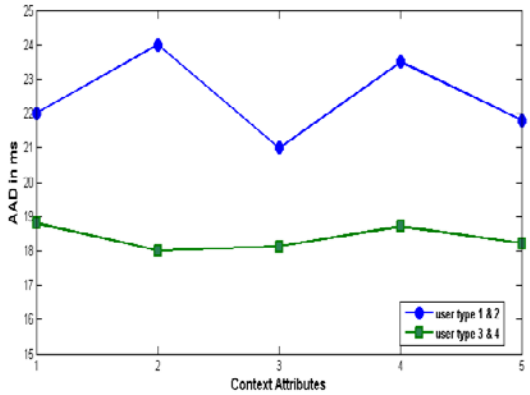

Figure 7: AAD for change in the authorization

The flexibility in access and memory to store related data add performance cost. So evaluation of AAD for change in authorization for different Usertypes is demonstrated to decide access control of the requested resource. A more AAD was observed for low level Usertypes compared to higher level Usertypes as the promotion to a higher level needs a clear and perfect analysis. For example Usertype 1& 2 requires more analysis compared to Usertype 3 & 4 because, Usertype 3 & 4 has almost all privileges except commercial and entertainment as mentioned in Table.2, but Usertype 1 & 2 have minimum rights. Fig.7 confirms the above discussion that, Usertype 1 & 2 requires 20% extra processing cost compared to Usertype 3 & 4.

6.0 CONCLUSION AND FUTURE SCOPE

The request to access particular resource is authorized when the content of request information is relevant to the user's current context information. The user's information is extracted through contextual information like resource, location, time and history. All these together, determine a required authorization for a user and makes the access control adaptive in nature.

In our future works, we intend to adopt a trust based adaptive context-aware access control, as trust enhances the action and activity to change the access control for the user in the security system.

REFERENCES

[1]. Ryan, J. Pascoe, D. Morse, "*Enhanced reality fieldwork: the context-aware archaeological assistant*", Computer applications in Archaeology, 1997.

[2]. A. K. Dey and G. D. Abowd, "*Towards a better understanding of context and context-awareness*", Tech. Report-Georgia Institute of Technology - GVU-99-22, 1999.

[3]. Shilit B, Theimer M, "Disseminating active map information to mobile hosts", IEEE Network, 1994.

[4]. A. Dey and G. Abowd, "*Towards a better understanding of context and context-awareness*", Proc. Human Factors in Computer Systems Conf., New York, 2000.

[5]. Shahin Fatima, Shish Ahmad and P. M. Khan, "*Certificate Based Security Services in Adhoc Sensor Network*", in BIJIT Issue 12, Vol .6, No 2, ISSN 0973 – 5658, 2014.

[6]. Jagadamba. G, Sharmila. P and Tejus. Gouda, "*A secured authentication system using effective keystroke dynamics*", in proceedings of International Conference on Emerging Research in Electronics, Computer Science and Technology, and also in Lecture notes on Electrical engineering, Springer, 4, (pp. 53–460), 2014.

[7]. Alexios Vasileiadis, "Security concerns and trust in the adoption of m-commerce", *Master thesis*, Mykolas Romeris University, 2013.

[8]. Ashema Hasti, "*Study of Impact of Mobile Ad – Hoc Networking and its Future Applications*", BVICAM's International Journal of Information Technology, Vol. 4, No. 1, 2012.

[9]. Yahya, S., Ahmad, E.A. and Jalil, K.A, "*The definition and characteristics of ubiquitous learning: A discussion*", Int. J. of Education and Development using Information and Communication Technology, Vol. 6, No.1, pp. 117-127, 2010.

[10]. Vicki Jones and Jun H. Jo, "*Ubiquitous learning environment: An adaptive teaching system using ubiquitous technology*", Proc. 21st ASCILITE Conf., 2004.

[11]. A. S. Patrick, A.C. Long and S. Flinn, "*HCi and security system*", Proc. of workshop at CHI, USA, 2003.

[12]. R.S. Sandhu et al, "Role-based access control models," IEEE Computer Society Press 29(2), pp. 38–47, 1996.

[13]. M. L. Wullems Chris and A. Clark, "*Toward context-aware security: an authorization architecture for intranet environments*", ACM press, Newyork, 2004.

[14]. N. Ryan, J. Pascoe and D. Morse, "*Enhanced reality fieldwork, the context-aware archaeological assistant*", in Gaffney, V. Et al. (Eds.) Computer Applications in Archaeology, 1997.

[15]. Gerhard Fischer, "*Context-Aware Systems-The 'Right Information', at the 'Right Time', in the 'Right Place', in the 'Right Way', to the 'Right Person'* ", Proc. of Advanced Visual Interfaces Conf., ACM, pp. 287-294, 2012.

[16]. Campbell and K. Nahrstedt, "*Gaia- A middleware infrastructure to enable active spaces*", IEEE Pervasive Computing, pp. 74-83, 2002.

[17]. Michael J. Covingtony et al, "*A context-aware security architecture for emerging applications*", Proc. 18th Annual Computer Security Applications Conf., IEEE, 2002.

[18]. A.S.M. Kayes et al, " *An Ontology-Based Approach to Context-Aware Access Control for Software Services*", Proc. Web Information Systems Engineering Conf., Part I, pp. 410–420, Springer, 2013.

[19]. Ali Ahmed, "context-aware access control in ubiquitous computing (CRAAC)", PhD dissertation, University of Manchester, 2010.

[20]. Gaurav Kumar and Anu Suneja, "*Resource Optimization Using XML*", BVICAM's International Journal of Information Technology, Vol. 1, No.2, pp.133, 2009.

[21]. Seo, Y. W. et al., "*A multi-agent system for enforcing Need-To-Know security policies*", Proc. Autonomous Agents and Multi Agent Systems, Workshop on Agent Oriented Information Systems (AOIS-04) Conf., pp. 163-179, 2004.

[22]. Barbara Rossi and Atsushi Inoue, "*Out-of-Sample Forecast Tests Robust to the Choice of Window Size*", Journal of business and economic statistics, Vol.30, No.2, pp. 432-453, 2012.