Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 775 – 782

ISBN 978-83-60810-14-9
ISSN 1896-7094

A Context-Risk-Aware Access Control Model for Ubiquitous Environments

Ali Ahmed
University of Manchester, School of Computer Science, Oxford Road, Manchester, M13 9PL, UK
Email: ahmeda@cs.man.ac.uk

Ning Zhang
University of Manchester, School of Computer Science, Oxford Road, Manchester, M13 9PL, UK
Email: nzhang@cs.man.ac.uk

***Abstract*—This paper reports our ongoing work to design a *C*ontext-*R*isk-*A*ware *A*ccess *C*ontrol (*CRAAC)* model for Ubiquitous Computing (UbiComp) environments. *CRAAC* is designed to augment flexibility and generality over the current solutions. Risk assessment and authorisation level of assurance play a key role in *CRAAC*. Through risk assessment, resources are classified into groups according to their sensitivity levels and potential impacts should any unauthorised access occurs. The identified risks are mapped onto their required assurance levels, called Object Level of Assurance (*OLoA*). Upon receiving an object access request, the requester's run-time contextual information is assessed to establish a Requester's Level of Assurance (*RLoA*) denoting the level of confidence in identifying that requester. The access request is granted iif *RLoA* ≥ *OLoA*. This paper describes the motivation for, and the design of, the *CRAAC* model, and reports a case study to further illustrate the model.**

I. Introduction

UBICOMP, sometimes referred to as pervasive computing, envisages a new computational environment in which heterogeneous devices with varying levels of capabilities and sensitivities interact seamlessly to provide smart services. By gathering information about surroundings (contextual data), the environment adaptively and non-intrusively provides context-aware services to users [1]. A context, defined as " *any information that can be used to characterise the situation of an entity* " [2], could relate to users (e.g. a ccess location) or to systems (e.g. network channel security level).

Context is dynamic, and its values may change from one session to another, and even during the same session. While UbiComp adapts its services to the surrounding context, the security services in the underlying environment should also be adaptive to the relevant context. In Access Control (AC) for example, we emphasis that an AC solution for UbiComp environments should be context-aware; it should react not only to value changes of individual contextual attributes, but also to the composite effect as caused by multiple contextual attributes value changes. It is inappropriate for an AC solution to take multiple contextual attributes directly as additional AC constraints while disregarding their composite effect on the risk level of unauthorised access in the underlying system. Furthermore, users' mobility further exacerbates the AC challenges in such environment. Mobile users with known/unknown devices move in/out of the environment without the protection of infrastructure-based firewalls exposes the environment to more security threats and attacks. An effective UbiComp AC solution should take into account the level of confidence in the entity trying to gain access to sensitive resources as well as the confidence level in the provided contextual information.

To realise this vision of context-aware AC, we have designed the *CRAAC* model, which uses the notion of risks and risk-linked levels of assurance (*LoA*) to govern the AC decisions. Through risk assessment, resources/services are classified into different groups each with a distinctive *OLoA*. In fact, the *OLoA* of a given resource object is determined based on its sensitivity level and the potential harm or impact should any unauthorised access to that object occurs. When an object access request is received, the requester's run-time contextual information is assessed to establish an *RLoA* denoting the confidence level in that requester. The access request is granted iif $RLoA \geq OLoA$.

This idea of using a risk linked *LoA* is inspired by the OMB/NIST e-Authentication Guidelines in [3], [4]. Similar to the OMB/NIST approach, *CRAAC* uses risk assessment to identify the risks for resource objects, and maps the identified risks to appropriate assurance levels (*OLoA)* for that object group. Our work, however, features the following distinct characteristic over the OMB/NIST work. The OMB/NIST work addresses the issue of authentication in the context of electronic transactions in a static environment, whereas our work focuses on context-aware AC in a dynamic UbiComp environment. As a result of this fundamental difference, our work differs from the OMB/NIST effort in the following ways. Firstly, the OMB/NIST guidance only considers issues related to users identification via the use of electronic credentials (e-credentials) that are largely static, whereas we handle a broader range of AC attributes, not only static e-credentials but also dynamic contextual attributes. Secondly, unlike the OMB/NIST work that only considers risk impact as caused by a single attribute (i.e. e-authentication credentials), we consider risk impacts by multiple attributes, as well as their composite effect on the authorisation assurance level. In addition, unlike the OMB approach by

Would like to thank the Faculty of Computers and Information, Cairo University, for its financial support.

which appropriate authentication technologies are chosen and implemented prior run-time to ensure that the underlying system achieves the required level of authentication assurance, *CRAAC* derives an *RLoA* for each requester at run-time based on their real-time dynamic contextual information, then compares this *RLoA* against *OLoA,* an AC threshold for the requested object, to make an AC decision.

The rest of this paper is structured as follows: section II gives an overview of the related work in context-aware AC. Section III describes the *CRAAC* model in detail. A case study is given in section IV. Finally, section V concludes the paper and outlines the future work.

II. Related Work

The main objective of an AC system is to restrict the actions a legitimate user can perform on a given resource object [5]. Role-Based Access Control (RBAC) [6] is a powerful model to specify and enforce organisational policies in a way that seamlessly maps to an enterprise structure [7]. Instead of assigning access rights to users directly, RBAC assigns access rights to roles that users can have as part of their organisational responsibilities. RBAC is considered as a policy natural authorisation approach particularly suited to large-scaled distributed environments [8]. However, the major weakness of the RBAC model is that it can not capture any security relevant information from its environment due to the subject-centric nature of its roles [8]. As a result, it can not enforce context-aware security policies, and therefore it is not adequate for UbiComp environments. To overcome this weakness, there have been proposals, will be discussed later on, to extend the basic RBAC model to equip it with the context-awareness capability. Those proposals use contextual information directly as additional constraints to govern the AC decision, as depicted by Fig. 1.

One of the earliest proposals is the Generalized Role-Based Access Control (GRBAC) model [9]. It introduced the concept of environment roles to capture contextual information from the underlying access environment. GRBAC, However, requires the use of complex system architecture to support the extended roles [10].

Another notable proposal is the Temporal RBAC (TRBAC) model [11] which extends the traditional RBAC model by introducing a temporal constraint into the AC specification to provide a mechanism to enforce time-dependent AC policies [12]. A subsequent proposal, the Generalized Temporal RBAC (GTRBAC) model [13], further extends the TRBAC model by introducing the notion of an activated role. More precisely, GTRBAC differentiates enabled roles, which subjects can activate, from active roles, which are being activated by at least one subject, for a more fine-grained AC.

The Spatial RBAC (SRBAC) model in [14] introduces a location-dependent constraint. A location space is divided into multiple *zones* , and an access permission is granted if the role condition is satisfied and the user is within the specified *zone* . The SRBAC model, as observed by [15], suffers from a lack of a semantic meaning of the position information, and it does not support the use of geometrically bounded roles.

The Dynamic Role Based Access Control (DRBAC) model [16], specifically designed for AC adaptation in UbiComp environments, is an interesting piece of work. Any change in an access context will be captured by an '*agent*' that will, in turn, trigger an '*event'* to cause a transition between the current role/permissions set to a new role/permissions set. DRBAC is considered as a pioneering effort in achieving context-aware authorisation in UbiComp environments. However, as noted by the authors themselves, implementing DRBAC can significantly increase the complexity of the applications concerned. This is particularly troublesome for the resource-restricted devices typically seen in UbiComp environments.

Young-Gab *et al* in [17] proposed a context-aware AC model which considers location, time, and system resources as AC constraints. The role is activated only if all the constraints are satisfied. The model has failed to consider the potential composite effects of, or the correlations between, these context attributes. In a similar approach, the model in [12] divides the location information in a levelled manner. The work formalised the model and conducted a case study, but again, it only focuses on temporal and spatial context attributes.

The work by G. Motta in [18] focuses on preserving patients' privacy and protecting the confidentiality of the patients' data in smart hospitals. The proposed contextual RBAC model classifies the patients' records based on their sensitivity levels, and an AC decision is made based upon the sensitivity level of the data being requested. The work, however, does not show how to adjust AC decisions in adaptation to the requesters' dynamic changes of the contextual information.

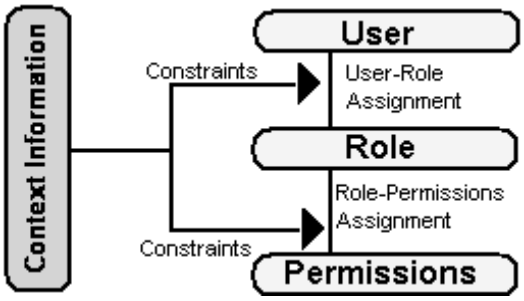

Fig 1. Existing Context-aware RBAC Approaches

A recent published work [19] has tried to address the need for evaluating the effect of multiple contextual attributes on an authorisation decision coherently. The model introduces the notion of risk-aware AC. The context information is used as the input to a risk assessment process to compute a risk value that is then fed into the authorisation decision engine. However, the scope of the risk assessment is quite broad covering confidentiality, integrity and authentication, so the delay incurred in the risk value calculation may be quite large, which may adversely affect the performance of the underlying AC system. Whether this delay would decrease the system ability to promptly adapt its decisions to context changes is yet to be investigated.

From the above discussions, it is clear that more investigation is needed for designing an AC model that could accommodate multiple contextual attributes in a generic and a coherent manner, and adapts its decisions to the dynamic changes of context, while, at the same time, keeping the costs down. The next section describes the design of such a model.

III. *CRAAC* Model

The *CRAAC* model aims at achieving *context-aware adaptation* (i.e. requirement 1 — capable of capturing, and adapting its decisions to the surrounding information for fine-grained AC), *flexibility* (requirement 2 - flexible enough to accommodate different contextual attributes and should not tie itself to a particular application domain), *extensibility* (requirement 3 - extensible to allow easy addition of new, and removal of obsolete, contextual attributes and any alterations imposed to the architecture as caused by such contextual attribute changes should be refrained within the context management part of the AC system), and *low performance costs* (requirement 4 — performance costs incurred in achieving context-awareness should be kept as low as possible).

A. Methodology Overview

The architecture of a context-aware AC system should include two major functional blocks, one for the *Context Management (CM)* and the other for AC. *CM* encompasses components for context acquisition, interpretation and representation while *AC* is responsible for making authorisation decisions. *CRAAC* is built on the basic RBAC model. While satisfying the requirements outlined above, the model has the two functional blocks (CM and AC) loosely coupled. That is, any change made to the attributes set managed by the CM block should not lead to changes in AC algorithms and policy representations managed by the AC block, and vice versa. One way to facilitate this loose coupling is to use a generic attribute that, on one hand, can capture the impact on AC as caused by context changes (in CM block), and, on the other hand, to feed that impact into the AC block as an additional AC constraint. In addition, it is desirable to link that attribute value to the resources sensitivity levels. Based upon these considerations, we introduce the notion of authorisation Level of Assurance (*LoA*) and use it as that generic attribute.

In details, resources/services are classified into object groups each with a distinctive *OLoA*. The determination of *OLoA* of an object group can be done via risk assessment for that group. The assessment identifies the risks, assesses their potential impacts, and maps the identified risks to an appropriate assurance level, i.e. *OLoA*. When an object access request is received, the decision engine will compare the *RLoA,* derived based on the requester's contextual information, against the *OLoA* of that object. The request is granted iif $RLoA \geq OLoA$. In fact, the *OLoA* of an object is the minimum authorisation requirement a user has to satisfy to gain access to that object. The more sensitive the object is, and/or the higher the potential impact, the higher the *OLoA*. Consequently, the higher the *RLoA* a requester would have to satisfy before the request can be granted.

One of the challenging tasks for designing this context-aware *LoA* linked AC is how to derive an *RLoA* value for a given access request based upon the requester's real-time contextual information. To achieve this, we need to, firstly, identify a set of contextual attributes that have impacts on the degree of certainty that the access request is from an entity that it claims to be from, secondly, to investigate, analyse, and define the respective assurance levels for these attributes, and thirdly, to devise a method that can derive the *RLoA* value based upon the attributes' LoA. In the remaining part of this section, we are going to address these issues respectively.

B. Contextual Attributes and their LoA Definitions

There are a number of factors that can increase the risk of unauthorised access, e.g. weak authentication protocol/token, less trustworthy access location, lack of intrusion detection and response systems, unprotected communication channels, … etc. In this paper, the focus is on the authentication token types, the access locations, the channel security, and the ability to respond to intrusion attacks (intrusion response). We name these factors as contextual attributes. At run-time, the risk associated to these attributes, if materialised, may lead to unauthorised information access. In the rest of this section, these attributes will be discussed in details.

1. eToken Attribute

Many factors in an e-authentication process affect the confidence level (i.e. *LoA*) in verifying a claimed identity. These include identity proofing, credentialing, credential management, record keeping, auditing, authentication protocols and token types. The assurance levels of some of these steps are achieved through procedural and process governance, while others may be left to the requesters' decision. For example, a requester may choose to use a particular authentication credential when making an access request. As our focus here is on the derivation of an authentication *LoA* and on linking it to the authorisation decision making, we exclude the procedural factors (i.e. user registration, credential management and storage procedures) from the *LoA* derivation. We rather focus on the types of e-credentials/tokens that are collectively called *eTokens*. Different *eTokens* provide varying degrees of confidence in entity identification and authentication. To quantify that degree of confidence, we introduce the notions of LoA_{eToken}.

Definition 1: *LoA_{eToken} refers to the service provider's degree of confidence that an eToken presented by a user is linked to his/her identity*.

The *eToken* types versus their assurance levels have been recommended by NIST [4], as shown in Table I. NIST recognises the token types of hard tokens, soft tokens, one-time password (OTP) device tokens, and user-name/password pairs, and defines four levels of LoA_{eToken}.

2. Access Location (ALoc) Attribute

Authentication services are of two main types; one is e-authentication by which a user is identified through the use of an eToken, and the other is physical authentication (p-authentication) by which a user is identified through the use of biometrics, sensors or location based services. *CRAAC*

recognises both of these authentication services. This is because, firstly, a combined use of e-authentication and location-based p-authentication not only provides optional services to users, but also offers a more reliable user identification. Secondly, location based services are commonly seen in UbiComp environments, and the access location is an important contextual attribute in such environment. Therefore, in addition to the *eToken* attribute, we introduce another authentication attribute, Access Location (*ALoc*). LoA_{ALoc} in relation to *ALoc* is defined below.

Definition 2: *LoA_{ALoc} refers to the degree of confidence in a claimed a ccess location .*

Depending on the application context, there are various ways to represent the location alternatives [20], [21]. As our As our focus is on the degree of confidence in a claimed location, we use the ' *zone* ' representation method [20], [14] to describe different location alternatives. Table II shows some possible location alternatives versus their likely assurance levels. The table is meant for illustration purpose only, as, unlike the case of *eToken* , *ALoc* attribute does not have any international consensus on their LoA_{ALoc} .

As outlined above, the two attributes, *eToken* and *ALoc,* both make direct contributions to the overall confidence level in the user identification. For example, as a password token is more vulnerable to guessing attacks than a PKI credential, then using it inside a secure room with a biometric physical authentication facility may be comparable, in terms of authentication assurance level, with a PKI credential used in a public area. To quantify such correlation between the two attributes, we introduce the notion of $LoA_{authN.}$

Definition 3: *LoA_{authN} is the overall confidence level associated with the composite authentication solution consisted of token based e-authentication and location based p-authentication .*

The derivation of LoA_{authN} given LoA_{eToken} and LoA_{ALoc} will be discussed later on.

3. Channel Security (CS) Attribute

The level of security protection of the channel running between the requester and the service provider may indirectly influence the risk level of unauthorised access. For instance, if a channel is more vulnerable to eavesdropping attacks, some credentials sent over the channel may experience a high risk of being compromised. In addition, the requested data may also experience a high risk of being disclosed to unauthorised entities via channel interceptions. For this reason, we introduce another contextual attribute, Channel Security (*CS*) and its *LoA* is defined below.

TABLE I.
TOKEN TYPES VERSUS LoA_{eToken} [4]

Token type	Levels			
	1	2	3	4
Hard token	√	√	√	√
One-time password token	√	√	√	
Soft token	√	√	√	
Password token	√	√		

TABLE II.
LOCATION ALTERNATIVES VERSUS LoA_{ALoc}

Alternatives	LoA_{ALoc}
Zone-0	**Level 0**; public area which does not have any provision for p-authentication.
Zone-1	**Level 1**; semi-public area which uses p-authentication to identify a group of users, e.g. through the use of a shared building key.
Zone-2	**Level 2**; personal area – access to this zone is controlled by the use of a locker key owned by a single user or a sensor based user identification (e.g. RFID).
Zone-3	**Level 3**; secured personal area – this zone uses some strong form of physical identification method that is less vulnerable to theft or loss than locker keys, e.g. Biometrics (physical) authentication facility.
Zone-4	**Level 4**; highly secured personal area – this zone may use multiple physical authentication methods.

***Definition* 4:** *LoA_{CS} refers to the degree of confidence in the channel (linking requesters and service providers) security .*

Similar to the *ALoc* attribute, the *CS* attribute does not have any international consensus on its assurance level definition. Table III describes an exemplar setting of a 5-level LoA_{CS} mimicking the NIST's *eToken LoA* definition. The exemplar setting is for a demonstration purpose and to be used later on in section V.

4. Intrusion Response Attribute

The last contextual attribute addressed by *CRAAC* is the Intrusion Response (*IR*) attribute, and its *LoA* is denoted by LoA_{IR}.

Definition 5: *LoA_{IR} refers to the degree of confidence in the ability of the underlying system to detect intrusion attacks and the ability to respond to such attacks.*

Sundaram in [22] divides IDS detection mechanisms into; anomaly, and misuse. An IDS may monitor host computers or network activities. The ability to detect intrusions and to respond promptly to these intrusions varies from an IDS to another. An IDS class may have an associated level of confidence as they vary in capabilities. Again there is no international consensus on the LoA_{IR} definition. Table IV is an exemplar LoA_{IR} setting for illustration purposes and to be used later on in the case study.

Once the AC related attributes are identified, and their assurance levels are specified, the next step is to estimate an overall/aggregate *LoA* value for an access requester based upon the assurance levels of such attributes.

C. Requesters' Aggregate LoA Derivations

1. Relationships among Multiple Attributes

As mentioned earlier, the confidence level in identifying a user may be influenced by multiple methods or attributes, either directly (eTokens and Access Location) or indirectly (Channel Security and Intrusion Response). To quantify the confidence level as influenced by the combination of a requester's multiple contextual attributes, we introduce the no-

tion of an overall (aggregate) assurance level for a requester, *RLoA*.

***Definition* 6:** *RLoA refers to an overall LoA in identifying a requester based upon the requester's contextual information associated to multiple contextual attributes (eToken, ALoc, CS, and IR).*

The derivation of *RLoA* depends on the types of the attributes used in the access session, the correlation among the attributes, and the used security policy. Formally, given a set of contextual attributes (A_1, A_2, ..., A_n) and their associated assurance levels (LoA_{A_1}, LoA_{A_2}, ..., LoA_{A_n}), *RLoA* can be expressed using a generic function, *f*, as:

TABLE III.
LOA ASSOCIATED TO CHANNEL SECURITY ATTRIBUTE

LoA_{CS}	Descriptions
Level 0	This attribute is disabled, or not used.
Level 1	Little or no confidence in channel security.
Level 2	Some confidence in channel security.
Level 3	High confidence in channel security .
Level 4	Very high confidence in channel security.

TABLE IV.
LOA ASSOCIATED TO INTRUSION RESPONSE ATTRIBUTE

LoA_{IR}	Descriptions
Level 0	IDS is disabled or no IDS is installed.
Level 1	Little or no confidence in the installed IDS.
Level 2	Some confidence in the installed IDS..
Level 3	High confidence in the installed IDS.
Level 4	Very high confidence in the installed IDS.

$$RLoA = f(LoA_{A_1}, LoA_{A_2}, \ldots, LoA_{A_n}) \quad (1)$$

The function *f* is determined by the relationship among the multiple attributes. We have identified two types of relationships, one is the *elevating* relationship and the other is the *weakest-link* relationship. In the *elevating* relationship, the combined use of two or more contextual attributes may result in the overall confidence level being higher than that provided by any one of the contextual attributes.

Elevating security is used by Microsoft in Windows Server 2003 to enable regular users to install applications even if they do not have the required permissions [23]. In our problem context, attributes, *eToken* and *ALoc*, are in an *elevating* relationship, as it is obvious that a combined use of e-authentication and location-based p-authentication will result in a more reliable user identification, thus a higher *LoA*.

In the *weakest-link* relationship, on the other hand, the value of *RLoA* is equal to the lowest attribute *LoA* value in the attributes set. This is in line with the *weakest-link* principle in system security. For example, attributes, {*eToken*, *ALoc*}, *CS* and *IR*, resembles more the *weakest-link* relationship (here *eToken* and *ALoc* are treated as one whole attribute in terms of *LoA*). This is because, even if the underlying authentication procedure is strong (thus difficult to impersonate), and channel security has a high assurance level (thus difficult to intercept useful information), provided that the service provider's system is easy to break into, there will still be a high risk of compromising server end of the identification and authentication procedure, e.g. by directly attacking credential files stored in the system. This implies the overall assurance level should not be higher than the lowest attribute *LoA* involved.

2. Converting LoA to Ratings

Later in this section, when we describe the *RLoA* derivation method, the attributes' *LoA* values (e.g. LoA_{eToken}, LoA_{ALoc}, LoA_{CS}, and LoA_{IR}) will need to be converted from levels (or ranks), as shown in Tables I-IV, to values in the real interval [0,1] (i.e. ratings or weights). To accomplish this rank-to-rating conversion of *LoA* values, we employ the Rank Order Centroids (*ROCs*) method, a well-known rank-to-rating (or weight) conversion technique. This subsection focuses on describing the *ROCs* method.

ROCs is often used in solving an *MCDA* (Multiple Criteria Decision Analysis) problem. It takes a set of attributes ordered by importance (ranks) and converts them into a set of approximated weights (ratings) as sometimes it may not be realistic to determine the precise weights [24]. *ROCs* originally proposed by Barron in [25] with an appealing theoretical rational for its weights [26]. In addition, the weights are derived by a systematic analysis of implicit information in the ranks which would give more accurate outcome [24].

Using the *ROCs* method, the weights are derived from a simplex $w_1 \geq w_2 \geq .... \geq w_n \geq 0$ restricted to:

$$\sum_{i=1}^{n} w_i = 1 \quad (2)$$

where *n* is the number of attributes (system cardinality). The vertices of the simplex are $e_1 = (1, 0, ..., 0)$, $e_2 = (1/2, 1/2\ 0, ..., 0)$, $e_3 = (1/3, 1/3, 1/3, 0, ..., 0)$, ... $e_n = (1/n, 1/n, ..., 1/n)$. The coordinates of the centroids (weights) are calculated by averaging the corresponding coordinates of the defining vertices [24]. In general, the weight of the k^{th} most important attribute is calculated as:

$$(\sum_{i=k}^{n} \frac{1}{i})/n \quad (3)$$

ROCs is a light-weight method for the rank-to-rating conversion as *ROC*-based analysis is straightforward and efficacious [24]. Therefore, it is particularly suited to the UbiComp environment. In addition, the weights can be calculated off-line, and uploaded into a rank-to-weight conversion table as shown in Table V to further reduce run-time overheads incurred in the conversion.

Tables, VI and VII, describe the corresponding LoA_{eToken}, *and* LoA_{ALoc} values in ratings converted by *ROCs receptively.*

It is worth noting that $level_4$ in both cases of LoA_{eToken} and LoA_{ALoc} is the most significant level and corresponds to the first rank. The same rule can be applied to convert the *LoA* values from levels to their corresponding ratings for the Channel Security and the Intrusion Response attributes.

3. RLoA Derivation in Elevating Scenarios

Given that a requester has *n* contextual attributes, (A_1, A_2, ..., A_n), and all the attributes are in an *elevating* relationship, and assume that each of the attributes has a confidence value associated to it, (LoA_{A_1}, LoA_{A_2},..., LoA_{A_n}), then, under the assumption that $LoA_{A_i} > 0$, where $i \in \{1, n\}$, the overall confidence value, *RLoA*, can be calculated (using probability theory) as [27]:

$$RLoA = 1-(1-LoA_{A_1})(1-LoA_{A_2})....(1-LoA_{A_n}) \quad (4)$$

where LoA_{A_i} is a real value in the interval [0, 1], 1 denoting the highest confidence and 0 the lowest. An advantage of this equation is that an attribute with a higher assurance value would have a higher impact on *RLoA*, and an attribute with a lower assurance value would have a lower impact on the overall assurance value. Applying equation (4) to attributes, *eToken* and *ALoc*, we can calculate LoA_{authN} That is:

$$LoA_{authN} = 1-(1-LoA_{eToken})(1-LoA_{ALoc}) \quad (5)$$

where the values of LoA_{eToken} and LoA_{ALoc} are given in Tables VI and VII, respectively.

Further applying equations (4) to all the attributes concerned in this section (i.e. *authN, CS and IR)* we obtain an *RLoA* value as:

$$RLoA = 1-(1-LoA_{authN})(1-LoA_{IR})(1-LoA_{CS}) \quad (6)$$

TABLE V.
OFF-LINE RANK-TO-RATING CONVERSION

Cardinality	W_1	W_2	W_3	W_4	W_5	... W_n
2	0.7500	0.2500				
3	0.6111	0.2778	0.1111			
4	0.5208	0.2708	0.1458	0.0625		
5	0.4567	0.2567	0.1567	0.0900	0.04	
... n						

TABLE VI .
ETOKEN RATINGS, LoA_{eToken}

Cardinality	$Level_4$	$Level_3$	$Level_2$	$Level_1$
4	0.5208	0.2708	0.1458	0.0625

TABLE VII.
ACCESS LOCATION RATINGS, LoA_{ALoc}

Cardinality	$Level_4$	$Level_3$	$Level_2$	$Level_1$	$Level_0$
5	0.4567	0.2567	0.1567	0.0900	0.0400

It is worth noting that with the *elevating* method, every attribute component *LoA* contributes towards the overall *RLoA*. As a result, the overall *RLoA* will be greater than the maximum *LoA* value afforded by any one of the contextual attributes involved. In some application scenarios or under a certain system setup, Equation (4) may not always be applicable to the attributes of *CS* and *IR*, i.e. equation (6) may not always be true. In such cases, the *weakest-link* method may be more appropriate.

4. RLoA Derivation in Weakest-Link Scenarios

When the composite influence of multiple contextual attributes follows the *weakest-link* principle, *RLoA* should then be calculated using the *minimum* function. That is, *RLoA* can be calculated using the following formula:

$$RLoA = min(LoA_{authN}, LoA_{IR}, LoA_{CS}) \quad (7)$$

where *min* is the minimum function that returns the minimum value of those enclosed in the brackets. Note that the calculation of LoA_{authN} remains the same, as the two attributes, *eToken* and *ALoc*, follow the *elevating* relationship due to its two-factor authentication nature.

IV. CASE STUDY

In this section, the *CRAAC* model is applied to a real-life context-aware authorisation scenario. The scenario describes a Smart Hospital (SH), where Drs. Alice and Bob work. They both have the same organisational role, and use their wireless devices to access the SH restricted services from anywhere (within the hospital). In detail, Alice uses a PDA while Bob uses a wireless laptop. Assuming the patients' data are divided into four categories, denoted by $Types_{1-4}$, and their respective *OLoA* values are given in Table VIII. Users of these services have subscribed to four contextual attributes, namely, *eToken*, *ALoc*, *CS*, and *IR*. Approximately at the same time, Alice and Bob are seeking access to a $Type_4$ service. Alice is in $Zone_4$ (i.e. with LoA_{ALoc} of $level_4$) and uses a user-name/password pair as her authentication token. Bob is in $Zone_1$ (i.e. with LoA_{ALoc} of $level_1$) and has got a PKI smart card. The installed IDS is of a '*High confidence*' type. As both access requests are made at approximately the same time, the LoA_{IR} value associated to both requests corresponds to $level_3$ (i.e. LoA_{IR} level is 3). However, LoA_{CS} varies from Alice to Pop where Alice uses a channel with LoA_{CS}=$Level_1$, Bob is on a $Level_3$ channel protection. Table IX summarises the *LoA* values of all these contextual attributes in both levels and ratings (converted using *ROCs*) for both Alice and Bob.

Now, let us compute the *RLoA* for Alice under the assumption the SH is running a strict security policy, i.e. using the *weakest-link* principle. Applying equations (5) and (7), we have LoA_{authN} = 1-(1- 0.1458) (1- 0.4567) = 0.5359 where *RLoA* = *min* (0.5359, 0.2567, 0.0900) = 0.0900. The authorisation decision engine compares Alice's *RLoA,* just computed, to the service $Type_4$ *OLoA* (0.1458). Obviously, as *OLoA*($Type_4$) > *RLoA*(Alice), which means the assurance level required by the resource $Type_4$ is higher than what Alice could achieve via her current context information, and therefore Alice is denied access to $Type_4$ services.

However, if the service provider uses an *elevating* security policy, LoA_{authN} remains the same but the overall *RLoA* would be (using equation (6)): *RLoA* = 1 - (1-0.5359) (1-0.2567) (1-0.09)=0.686. In this case, Alice will be granted access to $Type_4$ services as now $OLoA(Type_4) < RLoA$(Alice).

For Bob's access request, when the *weakest-link* security policy is used, *RLoA* is calculated as 0.2567 that is sufficient to grant Bob the access, as $OLoA(Type_4) < RLoA$(Bob). When the *elevating* policy is used, however, Bob's *RLoA* is 0.7591, which will also enable him to access $Type_4$ services.

As shown, *CRAAC* provides the flexibility to allow a service provider to adapt its AC decisions based on the requester's run-time contextual attribute values and the chosen AC policy model (e.g. the *elevated* or the *weakest-link* policies). For instance, if Alice upgrades her channel access software to use a stronger encryption algorithm and crypto key, she would be able to obtain the access permission even if the SH is running the *weakest-link* AC policy. This is because the associated LoA_{CS} value will be increased to 0.4567 (as a result of the channel security upgrade), and *RLoA* using *min* function would produce 0.2567 which is greater than the *OLoA* required by $Type_4$ services.

TABLE VIII.
OLOA REQUIREMENTS

Service	OLoA		Description
	Level	Value	
$Type_1$	4	0.5208	Patients' DNA data
$Type_2$	1	0.0625	Anonymous patient data
$Type_3$	3	0.2708	Patients' profiles
$Type_4$	2	0.1458	Statistical results for a group of patients

TABLE IX.
CASE STUDY ATTRIBUTES LOA VALUES

Context Attribute	Alice's LoA		Pop's LoA	
	Level	Value	Level	Value
eToken	2	0.1458	4	0.5208
Access Location	4	0.4567	1	0.0900
Channel Security	1	0.0900	3	0.2567
Intrusion Response	3	0.2567	3	0.2567

V. CONCLUSION AND FUTURE WORK

In this paper we have introduced a new AC model, *CRAAC*, for achieving fine-grained AC in UbiComp environments. Risk assessment and level of assurance play a key role in the *CRAAC* model. Resources are classified into different groups based upon their sensitivity levels and the potential impacts of unauthorised access. Each group is assigned a distinctive *OLoA* denoting the minimum required authorisation level of assurance for that resource group. Upon receiving an object access request, the requester's run-time contextual information is assessed, and an *RLoA* is derived based upon these contextual information. The access request is granted iif $RLoA \geq OLoA$.

CRAAC has a number of major advantages over the existing AC approaches. Rather than directly using context information as additional AC constraints, *CRAAC* uses an abstract parameter, the authorisation *LoA*, to decouple the context management from the AC functional module, thus achieving context-aware AC without loosing generality, extensibility and flexibility. Through identifying and grouping users' contextual attributes in relation to authorisation *LoA*, and quantifying and aggregating the contextual information of these attributes into assurance levels, *CRAAC* achieves context-based *LoA* linked AC that allows different contextual information to be captured, and different *LoA* algorithms to be used without affecting the AC module. In other words, through the use of *LoA*, we can have a generic approach to context-aware AC, which can easily be applied to different application domains.

Future work includes designing the architectural components of the *CRAAC* model, and prototyping and evaluating the model to investigate its efficiency and efficacy.

REFERENCES

[1] R. J. Hulsebosch, A. H. Salden, M. S. Bargh, P. W. G. Ebben, J. Reitsma. "Context sensitive access control", in *Proc. 10th ACM Symposium on Access Control Models and Technologies (SACMAT '05),* New York, 2005, pp. 111-119.

[2] A. Dey, "Understanding and Using Context", *Personal Ubiquitous Computing, vol. 5(1),* Springer-Verlag, 2001, pp. 4-7,London.

[3] US Office of Management & Budge, "Memorandum M-04-04: E-Authentication Guidance for Federal Agencies" , December, 2003

[4] W. E. Burr, D. F. Dodson, W. T. Polk, "Electronic authentication guideline", *NIST special publication 800-63 version 1.0.2,* April 2006.

[5] R. Sandhu, P. Samarati, "Access control: principles and practice", *IEEE Communications Magazine, vol. 32(9),* 1994, pp. 40-48.

[6] R. Sandhu, E. Coyne, H. Feinstein, C. Youman, "Role-based access control models", *IEEE Computer, vol. 29(2),* February 1996, pp. 38-47.

[7] S-. Chou, "An RBAC-based access control model for object-oriented systems offering dynamic aspect features", *IEICE - Trans. Inf. Syst., vol. 88(9),* Oxford University Press, 2005, pp. 2143-2147.

[8] S-. Park, Y-. Han, T-. Chung, "Context-role based access control for context-aware application". *High Performance Computing and Communications, vol. 4208,* September 2006, Springer Berlin/Heidelberg, pp. 572-580.

[9] M. J. Moyer, M. Ahamad, "Generalized role-based access control",in *Proc. 21st International Conference on Distributed Computing Systems (ICDCS '01),* Washington DC., April 2001, IEEE Computer Society, pp. 391-398.

[10] M. J. Covington, P. Fogla, Z. Zhan, M. Ahamad, "A context-aware security architecture for emerging applications", in *Proc. 18th Annual Computer Security Applications Conference (ACSAC '02),* Washington DC., 2002, pp. 249, IEEE Computer Society.

[11] E. Bertino, P. A. Bonatti, E. Ferrari, "TRBAC: a temporal role-based access control model", *ACM Trans. Inf. Syst. Secur., vol. 4(3),* New York, ACM Press, 2001, pp. 191-233.

[12] S-. Chae, W. Kim, D-. Kim, "Role-based access control model for ubiquitous computing environment", *Information Security Applications, vol. 3786,* February 2006, Springer Berlin / Heidelberg, pp. 354-363.

[13] J. Joshi, E. Bertino, A. Ghafoor, "Hybrid role hierarchy for generalized temporal role based access control model", in *Proc. 26th International Computer Software and Applications Conference on Prolonging Software Life: Development and Redevelopment*

(COMPSAC '02), Washington DC., IEEE Computer Society 2002, pp. 951-956.

[14] F. Hansen, V. Oleshchu, "SRBAC: a spatial role-based access-control model for mobile systems", in *Proc. 7th Nordic Workshop on Secure IT Systems (NORDSEC'03)*. Gj'vik, Norway 2003, pp. 129-141.

[15] H. Zhang, Y. He, Z. Shi, "Spatial context in role-based access control", *Information Security and Cryptology – ICISC 2006, vol. 4296, November 2006,* Springer Berlin/Heidelberg Lecture Notes in Computer Science 2006, pp. 166-178.

[16] Z. Guangsen, P. Manish, "Context-aware dynamic access control for pervasive applications", in *Proc. Communication Networks and Distributed Systems Modeling and Simulation Conference, San Diego, California, January 2004,* pp. 219-225.

[17] Y-. Kim, C-. Mon, D. Jeong, J-. Lee, C-. Song, D-. Baik, "Context-aware access control mechanism for ubiquitous applications", *Advances in Web Intelligence,* Springer Berlin/Heidelberg, May 2005, vol. 3528, pp. 236-242.

[18] G. H. M. B. Motta, S. S. Furuie, "A contextual role-based access control authorization model for electronic patient record", *IEEE Transactions on Information Technology in Biomedicine, vol. 7(3),* pp. 202-207, 2003.

[19] N. N. Diep, L. X. Hung, Y. Zhung, S. Lee, Y-. Lee, H. Lee, "Enforcing access control using risk assessment", in *Proc. 4th European Conference on Universal Multiservice Networks (ECUMN '07),* Washington DC., IEEE Computer Society, 2007, pp. 419-424.

[20] K. K. Konrad, T. Konrad, D. David, S. Howard, D. Trevor, "Activity zones for context-aware computing", *UbiComp 2003: Ubiquitous Computing,* Springer Berlin/Heidelberg Lecture Notes in Computer Science, vol. 2864, October 2006, 2003, pp. 90-106.

[21] F. Meneses, A. Moreira, "A flexible location-context representation", in *Proc. 15th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, (PIMRC 2004),* vol. 2, September 2004, pp. 1065-1069

[22] A. Sundaram, "An introduction to intrusion detection", *ACM Crossroads,* vol. 2(4), New York 1996, pp. 3-7.

[23] S. Giles, D. Bersinic, MCSA Windows server 2003 all-in-one exam guide (exams 70-270,70-290,70-291), McGraw-Hill Osborne Media, 2003, pp. 614.

[24] H. Barron, B. Barrett, "Decision quality using ranked attribute weights", *Management Science, vol. 42(11),* November 1996, pp. 1515-1523.

[25] H. Barron, "Selecting a best multiattribute alternative with partial information about attribute weights", *Acta Psychologica, vol. 80,* 1992, pp. 91-103

[26] B. S .Ahn, K. S. Park, "Comparing methods for multiattribute decision making with ordinal weights", *Computers & Operations Research,* Part Special Issue: Algorithms and Computational Methods in Feasibility and Infeasibility, vol. 35(5), May 2008, pp. 1660-1670.

[27] A. Ranganathan, J. Al-Muhtadi, R. H. Campbell, "Reasoning about uncertain contexts in pervasive computing environments", *IEEE Pervasive Computing, vol. 3(2),* Los Alamitos, IEEE Computer Society 2004 , pp. 62-70.