Accessing Data from Multiple Sources Through Context-Aware Access Control

A. S. M. Kayes[1], Wenny Rahayu[1], Tharam Dillon[1] and Elizabeth Chang[2]
[1]*La Trobe University, Melbourne, Australia*
[2]*University of New South Wales, Canberra, Australia*
[1]{a.kayes, w.rahayu, t.dillon}@latrobe.edu.au, [2]e.chang@adfa.edu.au

Abstract*—With the proliferation of data and services in today's dynamic computing environments, accessing data from multiple sources and consequently providing appropriate integrated results to the users has become a key challenge, often involving large processing overheads and administrative costs. The traditional context-sensitive access control models have been applied in different environments in order to access such data and information resources. Recently, fog-based access control models have also been introduced to overcome the latency and processing issues by moving the execution of application logic from the cloud-level to an intermediary-level through adding computational nodes at the edges of the networks. These existing access control models mostly have been used to access data from centralized sources. However, we have been encountering rapid changes in computing technologies over the last few years, and many organizations need to dynamically control context-sensitive access to data resources from multiple sources. In this paper, we introduce a new generation of** *context-aware access control* **approach, combining the benefits of fog computing and traditional context-sensitive access control solutions. We first introduce a** *general data model* **and its associated** *policy and mapping models, in order to** *access data from multiple sources***. In particular, we present a** *unified set of context-sensitive access control policies* **with the aim of reducing administrative and processing overheads. We then introduce a** *unified data ontology together with its reasoning capability* **in realizing our formal approach. We demonstrate the applicability of our proposal through a** *prototype testing* **and several** *case studies***. Experiment results demonstrate a** *better performance* **of our approach with respect to our earlier context-sensitive access control approach.**

***Index Terms*—Context-Aware Access Control, Data Model, Mapping Model, Data Security Policies, Ontology**

I. Introduction

Accessing data and information resources from multiple sources (e.g., distributed cloud servers, different databases) has increasingly become challenging nowadays due to the heterogeneous nature of data sources. Efficiently controlling the users' access to such data from multiple sources is one of the main challenges. How to provide integrated results to the users with low processing overheads and administrative complexities is another key challenge. Such new challenges require a new form of policy-based access control model with the potential to include on-the-fly data integration in order to deliver an integrated data view to the users. The access control decisions might be restricted to different granularity levels according to the relevant contextual conditions (e.g., the temporal information, profile information). For example, a data analyst's request to access and analyze the data about driving license holders (like the age and address of the drivers') may be allowed from the inside of the office during his duty time, whereas a data scientist may access and use such records for research purposes in different contexts.

A. Background

Among the different access control models available in the literature, Role-Based Access Control (RBAC) [1] is a representative and reliable security model for many practical applications to protect data and information resources [2]. In accordance with the embodiments of the user-role and role-permission mappings, the traditional RBAC model [1] and spatial and temporal RBAC models (e.g., [3]) have been widely accepted by different scientific communities due to their flexibility and simplicity in administration when faced with a large number of users and large amount of data. Considering a wide range of relevant context information [4] explicitly for access control is another key research direction, mainly exploiting the context-aware policy models to prevent unauthorized access of such data and information resources. Thus, Context-Aware role-based Access Control (CAAC) models (e.g., [5], [6]) have been introduced over the last few decades, incorporating the dynamically changing contextual conditions into the RBAC policies. These context-dependent models are mostly domain-specific and consider specific types of context information. We have a successful history of developing a family of CAAC models, where we have considered the general context information [4], the relationship context information [7] and the situational context information [8]. These contextual conditions mostly are obtained by exploiting the classical crisp sets and have been incorporated into the access control policies [9]. However, there are some conditions that cannot be obtained directly from crisp sets. As such, we have exploited the fuzzy sets in terms of an appropriate way to derive such fuzzy context information by utilizing our fuzzy context model [10]. For example, a patient's current health condition is derived "*95% critical*", i.e., criticality level is "*very high*", from the low-level data such as pulse rate and body temperature.

Looking at the existing context-sensitive access control models, these solutions extensively have been used to access data and information resources from centralized sources [11]. These models do not provide adequate functionality to access different data sets from multiple environments, by utilizing a single set of access control policies. Different data integration

techniques have been developed over the last few decades to collate data from multiple sources, such as schema matching [12]. These integration techniques mostly have been used to map between original sources of data (i.e., different schemas) and result in a global schema. However, these techniques are still limited in order to provide the "granted" or "denied" access control decision to the users, supporting a single set of access control policies to overcome overhead issues.

Due to the technological advancements in the online environment, currently, different stakeholders need to access data from many distributed sources. For example, the current cloud-based Internet of things (IoTs) paradigm [13] seeks a new form of context-sensitive access control model for building mechanisms of controlling data and information resources from multiple Big Data sources. The integration of such data directly from distributed sources raises semantic namespace and latency problems [14] due to lack of semantics and cloud-based services. The richer semantics of data model is needed to resolve the semantic namespace problem, dealing with the heterogeneous nature of such big data sets. However, the latter is forcing the organizations to overcome the latency issue by adding intermediary computational nodes at the edges of the networks [15]. In recent years, fog computing models have been introduced to reduce the latency and processing overheads involved in managing and accessing cloud-based data and services (e.g., [16]). These fog nodes usually provide intermediary computation and networking services between the end-users and the data servers. Over the last few years, several fog-based access control models have been proposed (e.g., [17], [18]) in the literature.These fog-based access control models are developed to access data and information resources from centralized environments. However, they are not truly context-aware and robust enough to develop CAAC mechanisms for accessing data from multiple, distributed environments. Overall, there is a grand challenge that traditional access control solutions and measures cannot meet such requirements in today's dynamic computing environments. As a result, we need to build new CAAC solutions for data and information resources coming from multiple sources.

B. Research Issues and Requirements

From our analysis of the literature and based on the identified characteristics of data sources, there is still a gap relating to the data access from multiple environments and consequently providing integrated results to the users. Such a gap raises the following research issues.

(RP1) How to effectively model access control policies to access data from multiple sources by means of reducing performance overheads and administrative costs? Thus, there is a need to specify a single and unified set of policies instead of multiple sets of policies.

(RP2) How to define a unified data model to access different data sets from multiple sources? Usually, different organizations have their own local schemas with different data structures. There is a need to define a generic schema for all data sources, considering the identical attributes of the similar data objects.

(RP3) How to map these access control policies to multiple data sources? There is a need to codify the mapping rules in terms of correlating different data sources.

C. Our Contributions and Organization of Paper

Our aim in this research is to introduce a new generation of *context-aware access control* model, combining the benefits of fog computing, context-sensitive access control and traditional data integration solutions, in terms of defining a unified global data model and facilitating access control to data from multiple sources. The significant contributions are listed as follows. We present a *data access scenario* to motivate our research in Section II. We propose a *formal approach to a general data model* with the aim of considering different data sets from multiple sources in Section III. In Section IV, we introduce a *unified data ontology* to represent the common classes in the relevant data sets and a *mapping ontology* to correlate these common classes with other equivalent classes. In this perspective, the proposed ontology-based approach performs schema mapping and correlates the multiple data sources accordingly. In Section IV, we also propose a *policy ontology* to provide access control decisions to the users, specifying a unified set of context-sensitive access control policies for all the different data sources. We evaluate our proposed approach by demonstrating a *walkthrough of our entire mechanism* via several case studies and a *prototype testing* through healthcare scenarios (see Section V). Section V also demonstrates the practicality of our approach through an empirical evaluation with respect to our earlier CAAC approach. Section VI briefly discusses the related work and presents a comparative analysis of our approach with respect to existing access control approaches. Finally, the conclusion and a roadmap for future research are presented in Section VII.

II. Research Motivation

Nowadays, a large number of data has been produced as a result of the abundance of Big Data sources about business and government services, their environments, and their end-users. Such data collection might be coming from centralized and/or distributed environments. This data abundance creates new opportunities and also raises new challenges to develop new form of access control mechanisms along with data integration capabilities. In the following, we consider an application scenario for our Australian Defence Logistics [19] project, which illustrates an access control to multiple data sources for different types of users within the distributed system. One of the specific aims of this project is to explore how Australian Defence can better enable and use the infrastructure of Defence Logistics resources in order to more effectively and efficiently access data from multiple environments.

We consider the following very specific application scenario: *John, who is a data analyst, is currently working with the Australian Department of Defence. His role is to deliver high quality services to record and visualize data usage*

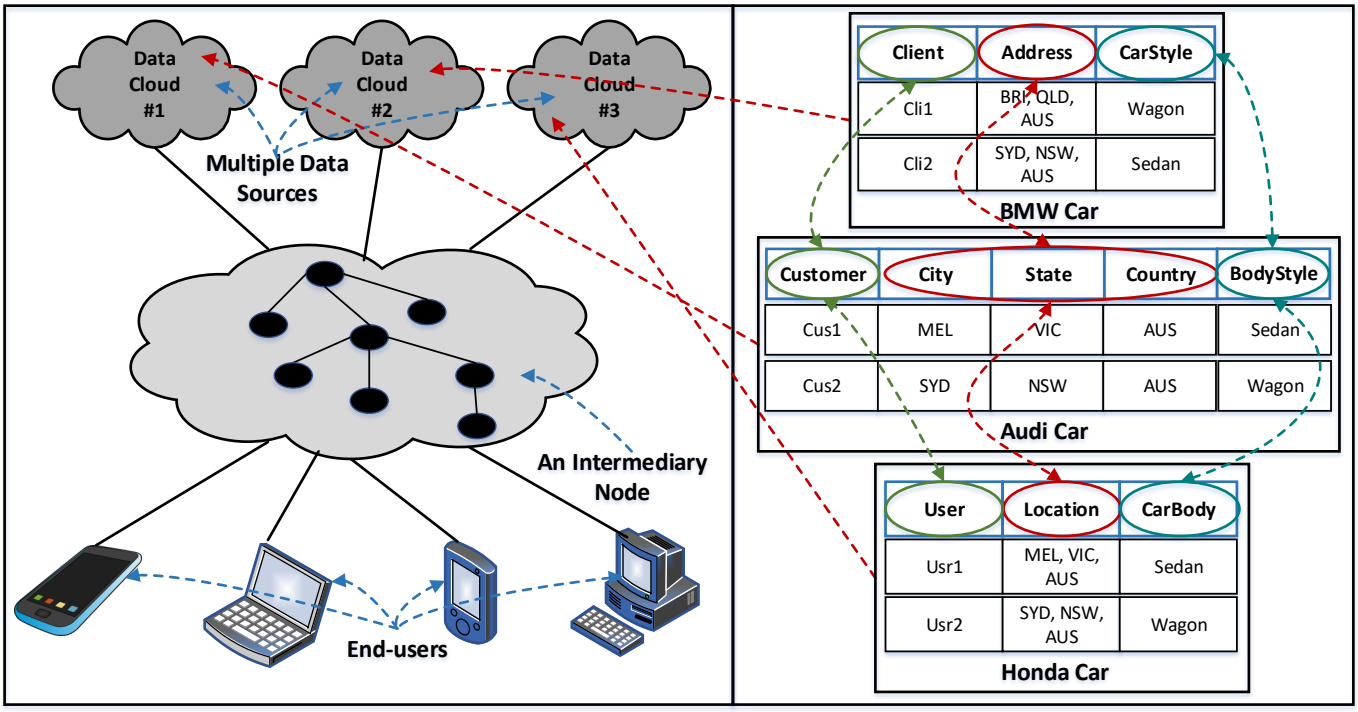

Fig. 1: The Relationship Chain from End-users to Multiple Data Sources (left) and Three Car Databases (right)

statistics based on the data from different sources. On the other hand, Richard, who is a data scientist, is working with the same department. His role is to further analyze these statistics to assist law enforcement and government policies through high-quality data analysis, using creative design and advanced statistical analytics on the resulting data sets. Currently, they both are assigned to the Defence Logistics project in a team to analyze the data on sedan cars (including the car owners' data) from all around Australia.

In this application scenario, John and Richard both need to access different types of car records (e.g., data about cars, car registrations, driving license owners and their insurance policies). However, they should maintain the security and privacy requirements of different stakeholders. For example, a data analyst only can access data about driving license owners' from his office location and during his working hours. Also, he only can see and visualize the statistical results, but not the detailed records. On the other hand, a data scientist can see such records from anywhere at anytime, even when he is on the move. In addition, he can access the detailed car data from recorded car details (such as the age and address of the car owners') for research purposes. Based on the analysis of the scenario, one thing is common here, the requesters (John and Richard) need to access data from multiple sources in different contexts. That is, such data might be associated with centralized or distributed environments. Also, the requesters need to deal with multiple data sets within different organizations (e.g., BMW, Audi and Honda companies). We can make two possible observations to facilitate context-sensitive access control to such data sets from distributed sources.

(1) ***Build a generic data model and specify a single set of policies subsequently to access data from multiple sources by utilizing mapping of generic schema to local schemas:*** In order to access data from multiple sources, we can build a unified data model to specify generic concepts and map all the local data schemas to the generic unified schema. Using this data model, we can introduce a policy model by taking into account a single set of data access policies for accessing data from distributed sources. In this fashion, we can reduce the number of access control policies, which in turn reduces the processing and administrative overheads. Our proposed data model and ontology can be found in Sections III and IV.

(2) ***Specify different sets of policies or use existing policies to access data from multiple sources:*** As an alternative to building a generic data model and specify a single set of policies accordingly, we can use different sets of policies individually to access data from multiple sources. In today's dynamic environments, this is really a big challenge to statically model all sets of access control policies according to the local data sources. On the one hand, it may impose extra burdens to policy administrators' to specify and manage such polices by means of multiple data sources. On the other hand, the number of policies involved in multiple data sources might potentially be quite large. However, in order to reduce the processing overheads for an access control system, we should avoid an excessive amount of access control policies [4]. The specification of a unified set of access control policies for this work can be found in Sections III and IV-B.

In the light of Observation 1, we illustrate the relationship chain among requesters (end-users), multiple data sources and an intermediary computational node, which is shown in the left part of Figure 1. Concerning the application scenario, different car companies such as BMW, Audi and Honda have their own data schemas. Three snapshots of raw data from these car databases are also captured in the right part of Figure 1. The relationship chain in Figure 1 mainly outlines the mapping between different end-users and multiple data sources. In order to support such mapping to different car databases and access data subsequently, there is a need for a context-sensitive access control application such as the Defence Logistics Information system (DLIS). In particular, an intermediary computational node is required to facilitate access control to the multiple car databases in such a DLIS application. In this paper, we only consider the homogeneous data from multiple sources. Our goal is to model a single set of policies to access necessary data from multiple sources based on the relevant contextual conditions. The computational node in Figure 1 will act as a ubiquitous tool in this complex structure to perform the integration of data sets that are coming from multiple sources.

III. Formal Approach to Access Data from Multiple Sources

In this section, we first provide some preliminary definitions with the purpose to illustrate our proposed solution approach. We then introduce our ontology-based approach in the next section, including a *unified data ontology* and its associated *access control policy and mapping ontologies*. In addition, we show the related examples from the application scenario.

Definition 1: ***Unified Data Model.*** A unified data model (UDM) is represented as a 2-tuple relation, including base and equivalent concepts. UDM also includes the associations involving these concepts, what we call relationships.

$$UDM \ = \ \langle BC, \ RE, \ EC \rangle \tag{1}$$

In our ontology, the base and equivalent concepts are represented by *classes* and *subclasses*, and the *object properties* are used to represent the associations or relationships between the base and equivalent concepts.

$$\begin{aligned} BC \ &= \ \{(bc_1, bc_2, ..., bc_i) | bc \in BC\} \\ EC \ &= \ \{(ec_1, ec_2, ..., ec_j) | ec \in EC\} \\ RE \ &= \ \{(re_1, re_2, ..., re_k) | re \in RE\} \end{aligned} \tag{2}$$

Thus, two sets of concepts (BC and EC) and a set of relationships (RE) form our UDM data model. We use $bc \in BC$ to represent a base concept, $ec \in EC$ to represent an equivalent concept and $re \in RE$ to represent a relationship between bc and ec.

Example 1: Looking at our application scenario, *Customer* (see the Audi car records in Figure 1) is a base concept that is equivalent to the concept of *Client* (see the BMW car records in Figure 1) and an association, named *equivalentOf*, is used to represent the relationship between them. In the next section, Figure 2 shows such relationships.

Definition 2: ***Policy Model.*** A policy model (Policy) is represented as a 4-tuple relation, including the following components: requesters, roles, contexts and permissions.

$$Policy \ = \ \langle Req, \ R, \ CC, \ P \rangle \tag{3}$$

In the above relation, Req represents a set of requesters ($req \in Req$), R represents a set of roles ($r \in R$), CC represents a set of contexts or contextual conditions ($cc \in CC$), and P represents a set of permissions ($p \in P$).

A permission is a set of 2-tuple relation on the base concept with different operations.

$$P \ \subseteq \ BaseConcept \times Operation \tag{4}$$

Similar to the basic RBAC model [1], in our policy model, a user can be assigned to a role under relevant policy constraints (e.g., the static conditions such as user's credentials), however the user needs to satisfy the necessary contextual conditions (e.g., the dynamic temporal and spatial conditions) [4]. Consequently, the user can access the necessary data from different sources (e.g., multiple databases, data clouds). Our policy model is based on the notions of different components and the associations that are included in the base concepts. In the following, we specify the mapping rules that are used to correlate these base concepts with other equivalent concepts with the aim of accessing data from multiple sources through a unified set of context-sensitive access control policies.

Definition 3: ***Mapping Rule.*** A mapping rule is represented as a one-to-one or one-to-many relationship between the base and equivalent concepts.

$$BC \ \equiv \ EC \tag{5}$$

An equivalent concept is either a single concept or can be formed based on the multiple concepts. Let us consider another set of concepts C ($c \in C$), each equivalent concept $ec \in EC$ is represented by the following relations.

$$\begin{aligned} C \ &= \ \{(c_1, c_2, ..., c_x) | c \in C\} \\ EC \ &= \ \{(..., (c_1), (c_2), (c_1 \wedge c_2), (c_1 \wedge c_2 \wedge c_3), ...) | \\ &\qquad ec \in EC \ \& \ c \in C\} \end{aligned} \tag{6}$$

In the above relations, we use $c \in C$ to represent a concept and $ec \in EC$ to represent an equivalent concept.

Example 2: Looking at the application scenario, the combination of three concepts ***City***, ***State*** and ***Country*** in the Audi car data snapshot is equivalent to the concept of ***Address*** in the BMW car data snapshot. Also, the concept ***User*** is equivalent to the concept of ***Customer***. These examples are represented in the following relations.

$$\begin{aligned} Address \ &\equiv \ City \wedge State \wedge Country \\ Customer \ &\equiv \ User \end{aligned} \tag{7}$$

For simplicity, we have used Address = {City, State, Country}, instead of Address $\equiv$ City $\wedge$ State $\wedge$ Country in our UDM ontology (see Figure 2).

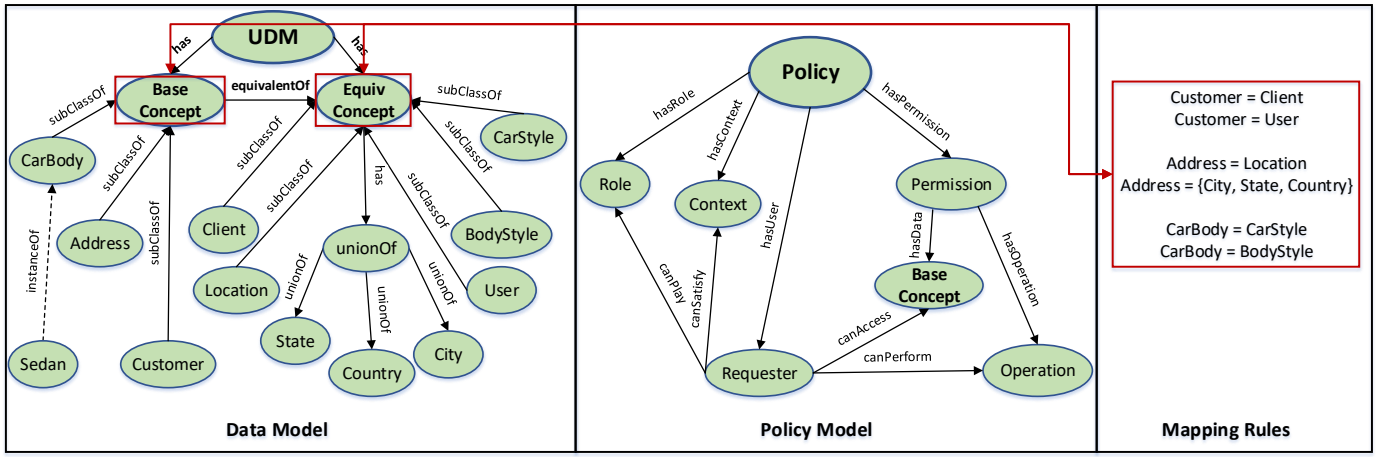

Fig. 2: An Excerpt of the Data Ontology (left) and its Associated Policy (middle) and Mapping Models (right) for DLIS application

IV. Unified Data Ontology

We introduce a unified data ontology (UDM ontology) where we model the concepts and associations with respect to the multiple data sources. It has two parts, the general ontology that includes the core concepts (e.g., classes) and the specialization ontology that includes the domain-specific concepts. The object properties are used to represent the associations (i.e., logical relationships) between these concepts.

Different modeling languages have been used in the literature to represent the concepts and the logical associations between concepts within different domains. The expressiveness and conceptual structure of the Web Ontology Language (OWL) [20] are very suitable for modeling information in accordance with different direct and RDF-based semantics [21]. The formal semantics of Description Logics (DL) [22] are embraced by the modeling constructs of OWL because of the knowledge representation schema that underlies the DL syntax. As such, we use the OWL language to model the UDM concepts and associations, and we use the DL grammar to specify the mapping rules and incorporate them into the UDM data ontology. We use the Protégé-OWL graphical API [23] to implement the data, mapping and policy ontologies. In addition, we use the SWRL language [24] and its built-in functions [25] to specify the reasoning rules for making context-sensitive access control decisions through context-sensitive access control policies.

Concerning our DLIS application, let us consider three car databases that have already been shown in Figure 1. Based on the DLIS application, a two-layered ontology, named *Unified Data Model (UDM)*, is represented in Figure 2 (see left part). The UDM ontology illustrates the main constructs, where we model the general core classes, domain-specific classes, and the logical relationships among them. The ontology has two core classes *BaseConcept* and *EquivConcept* which are organized into a hierarchy, named *UDM*. In this ontology, the logical associations between different classes are usually represented by is-a (*subClassOf*), union (*unionOf*) and equivalence (*equivalentOf*) relationships. For example, the classes *BaseConcept* and *EquivConcept* are linked by an arrow with a label *equivalentOf*. The UDM model shown in Figure 2 defines that the class *Location* is equivalent to the class *Address* and the *CarStyle* class is equivalent to the *CarBody* class. In the UDM ontology, the classes *Customer*, *Location* and *CarBody* are the three domain-specific concepts and they are the sub classes of the core class *BaseConcept*, which are represented by *subClassOf* relationships. In Figure 2, an individual named *Sedan* car is represented as an instance of the class *CarBody*, which is represented by a dashed arrow labelled *instanceOf*. For the sake of simplicity, in our UDM ontology, we do not show all the logical associations (object properties) between concepts.

A. Reasoning Rules

We incorporate the mapping rules into the UDM ontology. Our aim is to model and apply a unified set of access control policies for accessing data from multiple data sources as an efficiency improvement in terms of reducing the administrative costs and processing overheads.

We illustrate the domain-specific mapping rules in the right part of Figure 2. One of the mapping rules specifies that *CarStyle* is an equivalent concept of *CarBody*. As we consider multiple data sources, there are some equivalent classes/concepts which are formed based on the different concepts. For example, another mapping rule specifies the combination of *City*, *State* and *Country* classes is equivalent to the *Address* class.

B. Context-Sensitive Access Control Policies

A policy-driven data access model, simply policy model, has been proposed to access data from multiple sources by utilizing our UDM ontology. In particular, we specify a unified set of context-sensitive access control policies. In this research, we mainly focus on policy-driven data access from multiple homogeneous sources. Different policy languages have been

TABLE I: The Data Scientists' Policy

1	<**Policy** rdf:ID="$policy_1$">
2	<hasUser rdf:resource="#Requester_req"/>
3	<hasRole rdf:resource="#Role_$dataScientist$"/>
4	<hasContext rdf:resource="#Context_$anyLocation$"/>
5	<hasContext rdf:resource="#Context_$anyTime$"/>
6	<hasPermission rdf:resource="#Permission_p_1"/>
7	<hasData rdf:resource="#BaseConcept_p_1_$carAddress$"/>
8	<hasOperation rdf:resource="#Operation_p_1_$write$"/>
9	</**Policy**>

TABLE II: The Reasoning Rule

1	Policy(?policy$_1$) ∧
2	Requester(?req) ∧ hasUser(?policy$_1$, ?req) ∧
3	Role(?dataScientist) ∧ hasRole(?policy$_1$, ?dataScientist) ∧
4	Context(?anyLocation) ∧ hasContext(?policy$_1$, ?anyLocation) ∧
5	Context(?anyTime) ∧ hasContext(?policy$_1$, ?anyTime) ∧
6	Permission(?p$_1$) ∧ hasPermission(?policy$_1$, ?p$_1$) ∧
7	BaseConcept(?carAddress) ∧ hasData(?p$_1$, ?carAddress) ∧
8	Operation(?write) ∧ hasOperation(?p$_1$, ?write)
9	→ canAccess(?req, ?carAddress) ∧ canPerform(?req, ?write)

proposed in the literature. Our goal in this paper is to provide a guideline in which a unified set of access control policies can be applied to multiple data sources. As such, the basic elements of our policy model have been represented in the middle part of Figure 2.

The following core concepts are organized into a *Policy* hierarchy: *Requester*, *Role*, *Context*, *Permission*, *Operation* and *BaseConcept*. A policy can be read as follows: "*a user, who is the requester, by playing an appropriate role and under satisfying the necessary contextual conditions, can access data from multiple sources*". Our access control policies are specified and applied to the base concepts. The mapping rules are used to associate these base concepts with equivalent concepts (see the right part of Figure 2).

One of the main contributions of our proposal is its ability to model and apply a unified set of context-sensitive access control policies for accessing data from multiple sources targeting low processing and administrative overheads. Towards this goal, we have conducted two sets of experiments and demonstrated a prototype testing in the next section.

V. Evaluation of Our CAAC Approach

In this section, we demonstrate the applicability of our CAAC approach. We first provide a walkthrough of our entire CAAC mechanism via several case studies. We then demonstrate a prototype implementation and its associated application scenarios from the healthcare domain. In addition, we conduct two sets of experiments to evaluate our proposed approach with respect to our earlier CAAC approach.

A. Walkthrough of Our Proposal

We analyse the access requests from different users and the subsequent results with necessary data in the laboratory setups. The purpose of these case studies is to demonstrate the practical applicability of our proposed CAAC approach.

When a data access request arrives from a user, it includes the user-role and role-permission (data access permission) assignments based on our UDM ontology and its associated policy and mapping models. In the following case studies, we consider the dynamically changing context information, such as request times (e.g., John's data access request is within his duty time or not), locations (e.g., John is located in his office or not), inter-personal relationships between different persons (e.g., Jane is a treating doctor of the patient Bob or not), health conditions (e.g., Bob's current health status is highly critical, critical or normal), and co-located relationships (e.g., Jane and Bob are located in the emergency department of the hospital or not), as contextual conditions.

1) Revisiting Our Application Case Study: Consider our application scenario where Richard wants to access different car owners' records from multiple sources. Table I shows the specification of such data scientists' policy in OWL and Table II shows the relevant reasoning rule in SWRL for making context-sensitive access control decision through the applicable policies.

In this policy (see Table I), the access control decision is based on the following constraints: *who the requester is* (which is specified in *Line# 2*), *what role the requester can play* (*Line# 3*), *under what contextual conditions* (*Line# 4 and 5*) and *what resource is being requested* (*Line# 6 to 8*). For simplicity, we do not include the data type properties (and their values) in Tables I and II. Looking at the scenario, we can observe that Richard, who is a data scientist, can access different types of car records from multiple sources (e.g., Table II specifies a reasoning rule to access the records of different car addresses). Richard can access such records from any location at any time, however, a data analyst John only can access relevant car records from his office location and during his duty time.

The specification of the different contextual conditions is out of the scope of this paper. In this respect, we adapt our earlier context models [4][10] towards modeling the dynamic contextual conditions (fuzzy and normal contexts) and incorporating such conditions into our context-sensitive access control policies.

2) Other Real-World Application Scenarios: As contextual conditions are involved in the access control process, in our approach, the access control decisions depend on the wide range of contextual conditions, such as the request time, location, health status and so on. Our earlier context ontology [4] discusses the rich contextual conditions and extends in this research. We can apply our proposed CAAC approach in other real-world applications.

For example, concerning the emergency hospital scenario from our earlier research [4], Jane can play the emergency-doctor role when she is present in the emergency ward of the hospital, where a patient is admitted due to a severe heart attack. Consequently, she can access the emergency medical records (including other relevant records like previous medical history) of that patient to save his life from such a critical health condition.

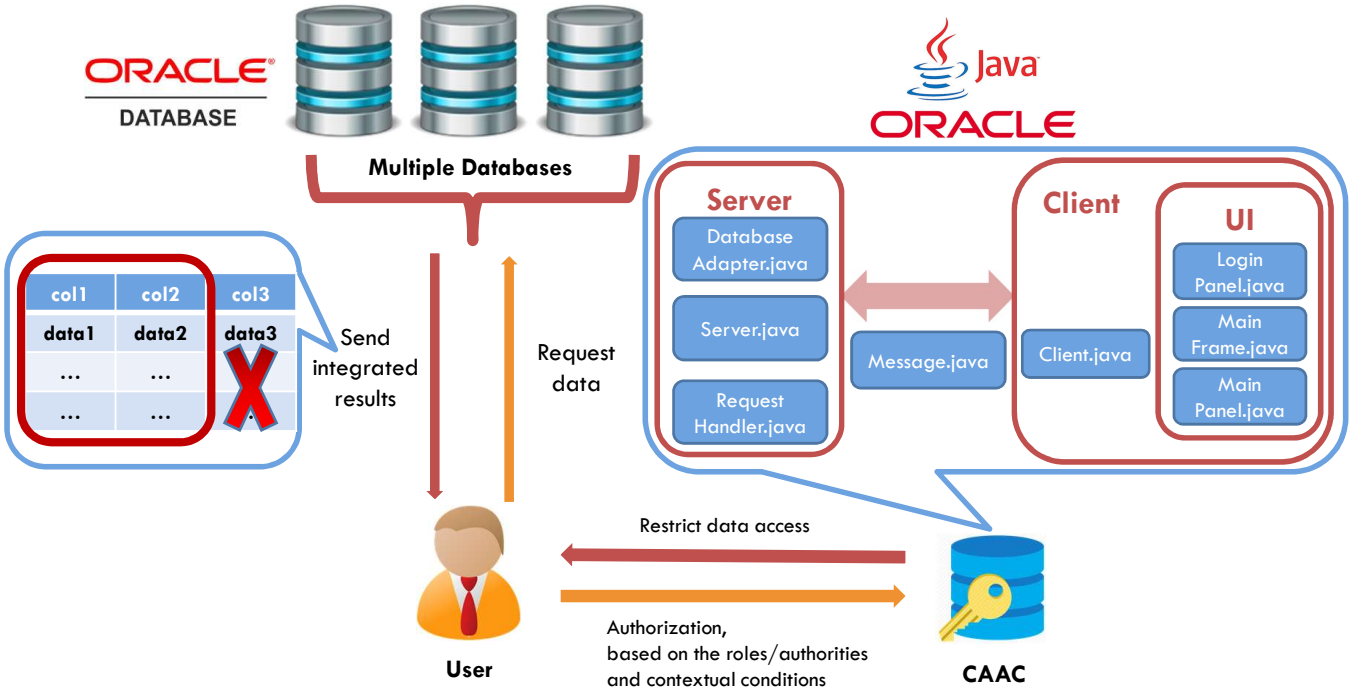

Fig. 3: Our Development Environment for the Healthcare Application

Let us consider another application scenario from our previous research [10], a paramedic John is allowed to play the emergency-paramedic role if he is co-located with the patient Tom at the scene of an accident. Using our proposed approach, he can acquire all the permissions (data access permissions) assigned to both paramedic and emergency-paramedic roles to provide emergency treatments.

Overall, this paper aims to address a significant research issue in the area of data access from multiple sources. Towards this goal, we introduce an intermediary computational node to control data and information resources from multiple sources, which mainly includes a unified data model and its associated policy and mapping models. In particular, our goal is to integrate multiple data sources by performing the corresponding schema mappings. We introduce a single set of context-sensitive access control policies to access data from multiple sources by utilizing this unified data schema. As such, we include the mapping rules about semantic mapping from individual local data schemas to unified data schema.

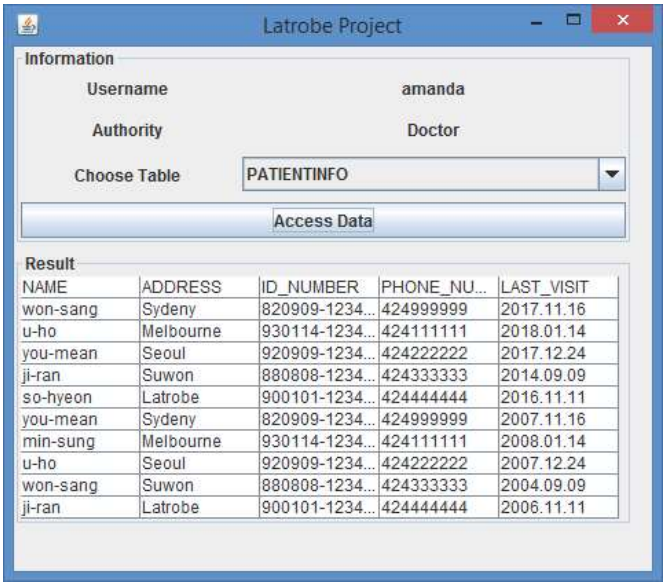

Fig. 4: A Screenshot of Our Application (Doctor's Request)

B. Healthcare Prototype and Its Associated Scenarios

We evaluate our proposed access control approach using another application scenario from the healthcare domain. As such, in this section, we present a CAAC application that is developed in our laboratory setup for the healthcare domain, in order to illustrate the use of our proposed CAAC approach. The main goal of this application is to access data from multiple databases based on the users' authorities/roles and relevant contextual conditions.

We have used the Java language and Oracle database to build our CAAC application. Figure 3 illustrates our development environment of the prototype implementation. When an access request comes from the user (client) using UI (user interface) part in Figure 3, the server part in our prototype generates the relevant query (data access query) according to the applicable context-sensitive access control policies, and the user can access the data from multiple databases accordingly. In this application, we have used three databases (one for access control logic and other two for different medical/heath records) and we actually limit the users to access data from

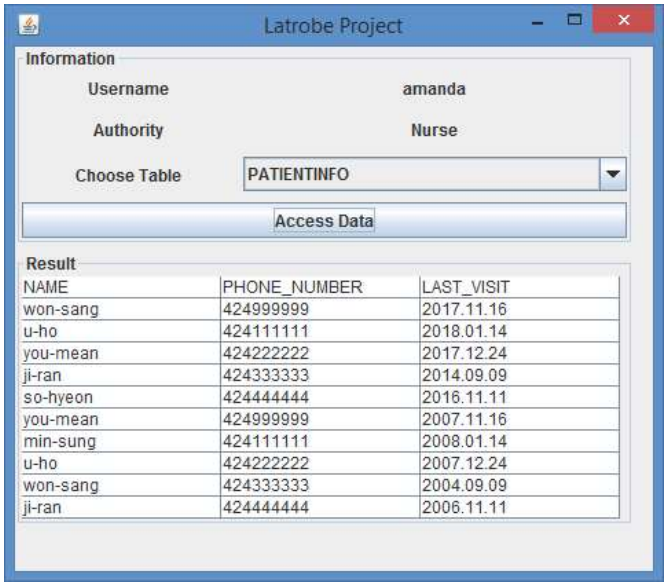

Fig. 5: A Screenshot of Our Application (Nurse's Request)

these databases based on their roles/authorities and the relevant contextual conditions. Figure 4 presents a screenshot from our healthcare application that shows the access control decision for doctor's access request. In this scenario, Amanda, who is a doctor, can access patients' information by satisfying the relevant contextual conditions. Figure 5 presents another screenshot that shows the access control decision for nurse's access request. In this scenario, Amanda, by playing a nurse role, can only have very limited access to patients' information, such as phone numbers and times for last visit.

The above-conducted case studies through different test scenarios demonstrate the applicability of our proposed approach to build CAAC applications in today's dynamic computing environments and facilitate access control to data from multiple databases accordingly.

C. Empirical Evaluation

In this section, we aim to demonstrate an empirical study on the performance of our proposed approach with respect to our earlier approach [4]. We conduct two sets of experiments and measure the query response time (i.e., processing overheads) with respect to different number of context-sensitive access control policies in conjunction with relevant contextual conditions. In order to model the healthcare roles (e.g., doctors, nurses, paramedics, researchers, data scientists and data analysts) and data resources (e.g., daily medical records, historical medical records and private health records), we adapt our role and resource ontologies from our earlier research [4][10]. The experiments are conducted in an Intel machine with Core i7@3.6GHz Processor and 16GB of memory. We deduce the average response time by executing the experiments 10 times and computing an arithmetic mean of them.

1) Experiment #1: Our Earlier Approach: In our first set of experiments, we specify all the different sets of context-

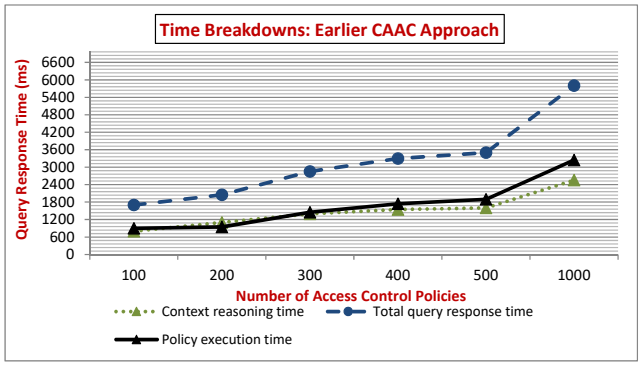

Fig. 6: Time Breakdowns of the Query Response Time

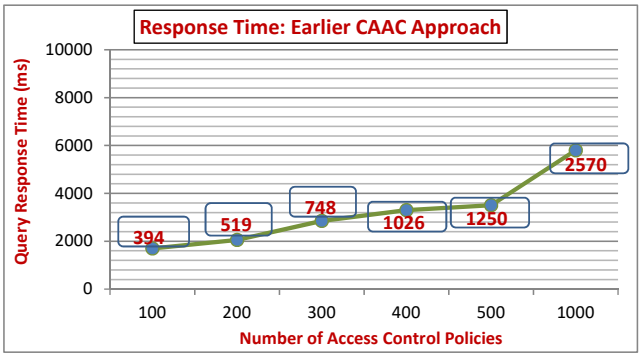

Fig. 7: Response Time with Respect to Number of Policies

sensitive access control policies for multiple databases and measure the CAAC performance. Based on the Australian standard classification of occupations (ASCO) of the health professionals [26], we model the healthcare roles (e.g., doctors, data scientists) and specify the context-sensitive access control policies for the health professionals. We vary the number of policies up to 1000 with respect to 138 different health professionals [26]. We also consider the different types of contextual conditions in these variations. We specify the separate access control policies for multiple databases/sources, an initial size of 100 policies and we increase this size up to 500 for an increment of 100 (see Figure 6). As such, we specify a large number of access control policies, which is 1000. In Figures 6 and 7, we can see that the performance overhead varies from 1.7 seconds (sec) to 5.8 sec with respect to the increasing size of the ontology knowledge-base. In this setup, using our earlier CAAC approach, we can see that the query response time is linearly increasing according to the number of policies up to 500, with respect to small size of the ontology knowledge-base. The performance overhead increases dramatically when the ontology size is big with respect to 1000 policies. This is due to the large number of policies and the reasoning task behind the data access query.

2) Experiment #2: Our Current Approach: In our second set of experiments, we specify a unified set of context-sensitive access control policies in order to access data from multiple databases. The number of policies in our current CAAC

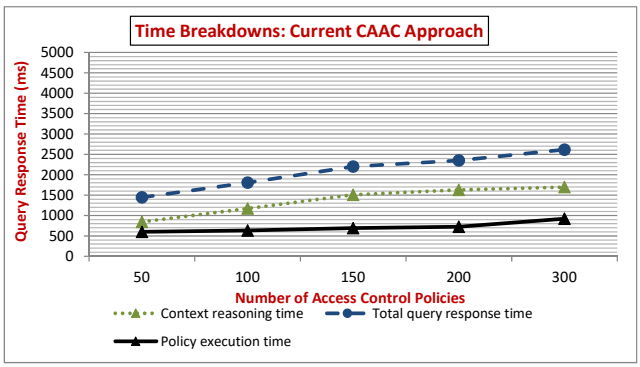

Fig. 8: Time Breakdowns of the Query Response Time

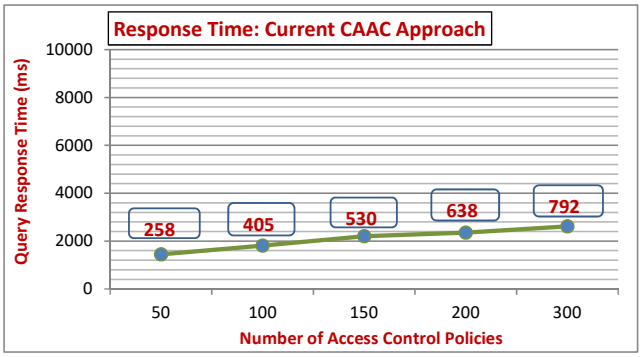

Fig. 9: Response Time with Respect to Number of Policies

approach is the main contributor in this current experiment setup, which is smaller than the previous approach (See Figures 8 and 9). Particularly, this is due to a unified set of context-sensitive access control policies for accessing data from multiple sources. On the other hand, the time taken to perform the reasoning task in our current approach is a little bit expensive than the earlier approach, as we have a data ontology and its associated mapping ontology in our proposed CAAC approach. However, we can see that an extra reasoning task concerning a unified set of policies does not have great impact in total query response time. The experimental results are illustrated in Figures 8 and 9. In our current CAAC setup, the query response time measures 2.6 sec with respect to 300 policies, which actually covers all the 1000 policies in our previous setup. Overall, we can see that we need an small number of policies using our current CAAC approach and subsequently the performance overhead decreases using our unified set of context-sensitive access control policies to access data from multiple databases.

In these two sets of experiments, we separate the access request processing time from the ontology loading time, as the ontology loading occurs once when our system runs for the first time. In this empirical study, we only consider the access request processing time, which is the main contributor in our experiments. Considering the above-conducted experiment results, we can conclude that our current CAAC approach offers better response time in controlling users' access to data from multiple sources with the benefits of a unified data model and its associated policy model. However, there is still a possibility of dealing with further performance overheads by using more powerful machines.

VI. Related Work and Comparative Analysis

In this section, we provide a short overview of some relevant access control approaches as the related area of our research. The overview includes the existing context-sensitive role-based access control approaches and the fog-based access control approaches. In addition, this section includes a brief analysis by positioning the new contributions of our proposed approach in relation to the current state-of-the-art.

A. Context-Sensitive Access Control

The Role-Based Access Control (RBAC) approach [1] is an established model of access control and is well recognized by security and privacy practitioners for its many advantages in large-scale authorization management [2]. It includes two fundamental parts: the first part provides the basic concept of user-role associations in which the users can play necessary roles that are usually organized in the organizational role hierarchies; and the second part provides another basic concept of role-permission associations in which the users can exercise necessary organizational functions that are associated with their roles. However, the computing technologies have been changing over time and in today's open and dynamic environments, many organizations have been targeted to build appropriate context-sensitive access control solutions for utilizing data and information resources from multiple environments.

Over the last few decades, different Context-Aware Access Control (CAAC) approaches have been introduced using role-based policies in conjunction with different contextual conditions. Bertino et al. [27], Joshi et al. [28] and Damini et al. [3] have extended the traditional RBAC approach by incorporating the temporal and spatial conditions into the access control policies. Recently, Schefer-Wenzl and Strembeck [29], Trnka and Cerný [5] and Hosseinzadeh et al. [6] have proposed several CAAC approaches in which access control is managed by means of different contextual conditions (e.g., locations, request times and resource-centric conditions). Similar to above-mentioned RBAC approaches, Colombo and Ferrari [11] have introduced a fine-grained access control approach utilizing NoSQL-based datastores. Using these context-sensitive RBAC approaches, users can access the necessary resources from centralized sources by playing their appropriate roles and based on the contextual conditions. These approaches are mostly domain-specific and are not adequate enough to utilize a wide variety of dynamically changing conditions of the environments (e.g., the interpersonal relationships, the critical situations). Towards this end, in this paper, we adapt our initial context model [4][10] to capture and infer the access control-specific contextual conditions. However, these extended RBAC approaches do not provide adequate methodological and implementation supports to model a unified set of access control policies with respect to accessing data from multiple sources

with low overheads. Different from these existing context-sensitive approaches, we in this paper propose a unified data model and its associated policy model in order to deal with the processing and latency overheads. As such, we introduce a global data ontology and its mapping model in order to utilize the benefits of a unified set of context-sensitive access control policies and overcome the overhead issues accordingly.

We have a successful history of using a wide range of contextual conditions for context-oriented decision making. In [4], we have introduced an ontology-based context-aware RBAC approach to information resources, where we consider the context information about the state of the users, resources and their surrounding environments (e.g., patients' profiles, users' locations, users' request times). In [7], we have introduced an ontology-based relationship-aware RBAC approach, incorporating the relationship context information (e.g., the different granularity levels of relationship, the relationship types, the relationship strengths) into the policies. In [8], we have introduced an ontology-based situation-aware RBAC approach, where we incorporate the purpose-oriented situation information (e.g., normal/emergency treatment purpose, research purpose) into the policies. Our earlier approaches do not provide adequate functionalities to access data from multiple sources utilizing a unified set of context-sensitive access control policies.

The access control policies in the above-discussed traditional and context-sensitive RBAC approaches are based on involving the normal contextual conditions, which can be usually derived from the crisp sets (e.g., an event such as "surgery in progress" or "not", a patient is located "in the emergency department of the hospital" or "not"). In [10], we have introduced a Fuzzy logic-based CAAC (FCAAC) approach in order to facilitate context-sensitive access control to resources according to the fuzzy conditions. Using our FCAAC approach, a fuzzy contextual condition such as a patient's current health status is "60% normal with criticality level 0.40" can be derived from other relevant information (e.g., pulse rate and body temperature of the patient). Our earlier CAAC approaches are developed to access data and resources from centralized environments. However, like the existing context-sensitive RBAC approaches, our earlier approaches are not adequate to access data coming from multiple sources by dealing with overheads and administrative issues.

B. Fog-Based Access Control

Recently, several fog-based access control approaches have been proposed to overcome the latency and processing overheads by moving the execution of application logic from the cloud levels to the edges of the network [15].

Due to the rapid development and technological advancements in the cloud-based environments, users need to access data and information resources from multiple sources nowadays. The integration of such data and information resources usually raises semantic namespace and latency problems [14], due to the lack of semantics and data coming from multiple environments. In order to deal with such issues, there is a need for the richer semantics of data model, dealing with the nature of such data sets from multiple sources. However, currently, these issues have been forcing the organizations to overcome the associated overheads by adding intermediary computational nodes at the edges of the networks.

Zaghdoudi et al. [17] have proposed a fog-based access control approach to overcome the overhead issue. They consider the information about the subjects, objects and operations as contextual conditions. Salonikias et al. [30] have presented a recent study on intelligent transport systems by utilizing the fog computing nodes and corresponding fog-based access control models. Both of the research works have been concerned with several important requirements of the fog-based access control schemes, such as context-awareness and processing overheads. The authors also have discussed the decentralization of authority from a single administrative location to other locations in order to overcome the associated overheads. Recently, Yu et al. [18] and Zhang et al. [31] have also proposed the fog-based access control approaches in order to share and access data along with the benefits of encryption and decryption mechanisms. Overall, these existing fog-based approaches have been developed to access data and information resources from centralized environments. However, these access control approaches are not truly context-aware and robust enough to build fog-based CAAC applications when accessing data from multiple sources according to the relevant contextual conditions.

In this respect, different from these existing fog-based access control approaches, our proposed CAAC approach in this paper is robust enough and truly context-aware. It considers a wide range of contextual conditions and introduces a unified data model and its associated policy model to deal with data from multiple sources.

C. Comparative Analysis

The context-sensitive role-based access control approaches have been applied to access data and information resources from centralized environments. However, these approaches are not adequate to access data from multiple sources due to the problems of administrative and latency overheads. On the other hand, the existing fog-based access control approaches are not truly dynamic and context-aware. With the increasing demand of accessing data and information resources from multiple sources, different stakeholders' requirements for security and privacy are becoming more challenging nowadays. Therefore, there is a grand challenge that traditional access control solutions and measures cannot meet such requirements in today's dynamic computing environments. As a result, in this paper, we introduce a new CAAC approach in order to support access control to data and information resources from multiple sources. We include both the formal and ontology-based implementation models to specify a unified context-sensitive access control policies with the benefits of mapping functionality. In particular, in includes a unified data model and a mapping model in order to correlate data and information resources from multiple sources. We present a walkthrough of

our entire CAAC mechanism by using several case studies and a prototype testing. Finally, we present an empirical evaluation to validate the feasibility of our proposed approach.

VII. Conclusion and Associated Research Challenges

Accessing data and information resources from multiple sources through appropriate access control mechanisms has been receiving increasing attention. A key factor in the success of such an approach is the need to access necessary data from multiple sources beyond that which normally associated with users' roles. To date, several role-based, fog-based and context-aware access control approaches have been introduced to access data and information resources from centralized sources. However, these approaches are not robust enough to develop CAAC mechanisms for accessing data from multiple sources due to the problems of latency and processing overheads and the lack of context-awareness as well. Many cloud-based organizations nowadays have been targeted to avoid such overheads and latency issues by adding intermediary computational nodes at the edges of the networks.

In this paper, we have introduced a new direction of CAAC solution that facilitates context-sensitive access control to data and information resources from multiple sources. Our proposed CAAC approach provides a flexible policy specification solution to the problem of reducing processing overheads, by specifying context-sensitive access control policies and consequently controlling users' access to data at multiple granularity levels. Our solution significantly differs from the existing access control solutions in that it utilizes the benefits of a unified set of policies and its associated mapping functions in order to access data from multiple sources. This paper provides a definition of the unified data model. It also defines the access control policies and the mapping rules to facilitate context-sensitive access control to data from multiple sources. We have introduced an ontology-based approach in realizing these preliminary definitions, including the data, policy and mapping ontologies. We have demonstrated the feasibility of our proposal through a walkthrough of our whole approach, using the OWL, DL and SWRL languages to model the core and domain-specific concepts of the ontologies. We have also demonstrated the applicability of our approach through a prototype testing. Finally, we have carried out two sets of experiments and presented an empirical comparison of the performance of our proposed approach compared to our earlier CAAC approach. The evaluation results show that our proposed approach with the benefits of a unified set of access control policies can be effectively used in practice.

Our proposed access control approach can be applied to deal with the issue of data heterogeneity, by accessing data from distributed and heterogeneous cloud sources. There is a need to investigate a generic data model to achieve semantic interoperability between heterogeneous data models from distributed sources. We can also extend our proposed data model, including an integrated data view model for the end-users with the goal of providing integrated results across many distributed and heterogeneous sources.

Acknowledgment

We acknowledge the contributions of our internship students, Yuho Lee and Minsung Han from Gachon University, South Korea, for the development of our healthcare application. They are partially supported by the Korea Ministry of ICT and Future Planning grant to Gachon University National Program of Excellence in Software. The statements made herein are solely the responsibility of the authors.

References

[1] Sandhu, R.S., Coyne, E.J., Feinstein, H.L., Youman, C.E.: Role-based access control models. IEEE Computer **29** (1996) 38–47

[2] Ferraiolo, D.F., Sandhu, R., Gavrila, S., Kuhn, D.R., Chandramouli, R.: Proposed NIST standard for role-based access control. TISSEC **4**(3) (2001) 224–274

[3] Damiani, M.L., Bertino, E., Catania, B., Perlasca, P.: GEO-RBAC: a spatially aware RBAC. TISSEC **10**(1) (2007) 1–42

[4] Kayes, A.S.M., Han, J., Colman, A.: OntCAAC: An ontology-based approach to context-aware access control for software services. Comput. J. **58**(11) (2015) 3000–3034

[5] Trnka, M., Cerný, T.: On security level usage in context-aware role-based access control. In: SAC. (2016) 1192–1195

[6] Hosseinzadeh, S., Virtanen, S., Rodríguez, N.D., Lilius, J.: A semantic security framework and context-aware role-based access control ontology for smart spaces. In: SBD@SIGMOD. (2016) 1–6

[7] Kayes, A.S.M., Han, J., Colman, A., Islam, M.S.: Relboss: A relationship-aware access control framework for software services. In: CoopIS. (2014) 258–276

[8] Kayes, A.S.M., Han, J., Colman, A.W.: An ontological framework for situation-aware access control of software services. Inf. Syst. **53** (2015) 253–277

[9] Kayes, A.S.M., Han, J., Colman, A.: A semantic policy framework for context-aware access control applications. In: TrustCom. (2013) 753–762

[10] Kayes, A., Rahayu, W., Dillon, T., Chang, E., Han, J.: Context-aware access control with imprecise context characterization through a combined fuzzy logic and ontology-based approach. In: CoopIS. (2017) 132–153

[11] Colombo, P., Ferrari, E.: Fine-grained access control within NoSQL document-oriented datastores. Data Science and Engineering **1**(3) (2016) 127–138

[12] Bellahsène, Z., Bonifati, A., Rahm, E.: Schema matching and mapping. Springer (2011)

[13] Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M.: Internet of things (iot): A vision, architectural elements, and future directions. Future generation computer systems **29**(7) (2013) 1645–1660

[14] Ylitalo, J., Nikander, P.: A new name space for end-points: Implementing secure mobility and multi-homing across the two versions of ip. In: 5th European Wireless Conference. (2004) 435–441

[15] Saurez, E., Gupta, H., Mayer, R., Ramachandran, U.: Demo abstract: Fog computing for improving user application interaction and context awareness. In: Internet-of-Things Design and Implementation (IoTDI), 2017 IEEE/ACM Second International Conference on, IEEE (2017) 281–282

[16] Stojmenovic, I., Wen, S., Huang, X., Luan, H.: An overview of fog computing and its security issues. Concurrency and Computation: Practice and Experience **28**(10) (2016) 2991–3005

[17] Zaghdoudi, B., Ayed, H.K.B., Harizi, W.: Generic access control system for ad hoc mcc and fog computing. In: International Conference on Cryptology and Network Security, Springer (2016) 400–415

[18] Yu, Z., Au, M.H., Xu, Q., Yang, R., Han, J.: Towards leakage-resilient fine-grained access control in fog computing. Future Generation Computer Systems **78**(2) (2018) 763–777

[19] Waters, G., Blackburn, A.J.: Australian Defence logistics: the need to enable and equip logistics transformation. Kokoda Foundation Limited (2014)

[20] OWL2: OWL 2 Web Ontology Language (W3C recommendation: 11 december 2012), https://www.w3.org/tr/owl2-overview/ (2018)
[21] Riboni, D., Bettini, C.: OWL 2 modeling and reasoning with complex human activities. Pervasive and Mobile Computing **7** (2011) 379–395
[22] De Bruijn, J., Lara, R., Polleres, A., Fensel, D.: OWL DL vs. OWL Flight: Conceptual modeling and reasoning for the semantic web. In: Proceedings of the 14th international conference on World Wide Web, ACM (2005) 623–632
[23] Protégé: OWL Graphical API, http://protege.stanford.edu/ (2018)
[24] SWRL: Semantic Web Rule Language, http://www.w3.org/submission/swrl/ (2018)
[25] SWRLB: SWRL Built-Ins for comparisons and Math Built-Ins, http://www.daml.org/2004/04/swrl/builtins.htm (2018)
[26] ASCO: Australian Standard Classification of Occupations of Health Professionals, http://www.abs.gov.au/ (2018)
[27] Bertino, E., Bonatti, P.A., Ferrari, E.: TRBAC: A temporal role-based access control model. TISSEC **4**(3) (2001) 191–233
[28] Joshi, J.B., Bertino, E., Latif, U., Ghafoor, A.: A generalized temporal role-based access control model. TKDE **17**(1) (2005) 4–23
[29] Schefer-Wenzl, S., Strembeck, M.: Modelling context-aware rbac models for mobile business processes. IJWMC **6**(5) (2013) 448–462
[30] Salonikias, S., Mavridis, I., Gritzalis, D.: Access control issues in utilizing fog computing for transport infrastructure. In: International Conference on Critical Information Infrastructures Security, Springer (2015) 15–26
[31] Zhang, P., Chen, Z., Liu, J.K., Liang, K., Liu, H.: An efficient access control scheme with outsourcing capability and attribute update for fog computing. Future Generation Computer Systems **78**(2) (2018) 753–762