2-2012

SecDS: A Secure EPC Discovery Services System in EPCglobal Network

Jie SHI

Darren SIM

Yingjiu LI
Singapore Management University, yjli@smu.edu.sg

Robert H. DENG
Singapore Management University, robertdeng@smu.edu.sg

Citation

SHI, Jie; SIM, Darren; LI, Yingjiu; and DENG, Robert H.. SecDS: A Secure EPC Discovery Services System in EPCglobal Network. (2012). *CODASPY '12: Proceedings of the 2nd ACM Conference on Data and Application Security and Privacy, February 7-9, 2012, San Antonio, TX*. 267-274.
Available at: https://ink.library.smu.edu.sg/sis_research/1643

SecDS : A Secure EPC Discovery Service System in EPCglobal Network

Jie Shi, Darren Sim, Yingjiu Li, Robert Deng
School of Information Systems, Singapore Management University, Singapore 178902
{jieshi,darren.sim.2009,yjli,robertdeng}@smu.edu.sg

ABSTRACT

In recent years, the Internet of Things (IOT) has drawn considerable attention from the industrial and research communities. Due to the vast amount of data generated through IOT devices and users, there is an urgent need for an effective search engine to help us make sense of this massive amount of data. With this motivation, we begin our initial works on developing a secure and efficient search engine (SecDS) based on EPC Discovery Services (EPCDS) for EPCglobal network, an integral part of IOT. SecDS is designed to provide a bridge between different partners of supply chains to share information while enabling them to find who is in possession of an item. The most important property of SecDS is: while efficiently processing user's search, it is also secure. In order to prevent unauthorized access to SecDS, an extended attribute-based access control model is proposed and implemented such that information belonging to different companies can be protected using different policies.

Categories and Subject Descriptors

D.4.6 [**Security and Protection**]: Access controls; H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

Design, Security

Keywords

EPC Discovery Service, Access Control, EPCglobal Network, Internet of Things

1. INTRODUCTION

As an integral part of future Internet, Internet of Things (IOT) has drawn considerable attention from the industrial and research communities around the world. Through IOT, we can look forward to a world where physical objects and virtual data interact [12], generating mass amount of data that will exceed that of what we have on the world-wide-web (WWW) today. There is an urgent need for a relevant search engine, to help us make sense of this data, just as how BING and Google are helping us navigate through the trillion-page Internet today.

EPCglobal network [7] is an important part of IOT. As a global standard RFID data sharing infrastructure, EPCglobal network is made up of Electronic Product Code (EPC) [10], EPC Information Services (EPCIS) [11], EPC Discovery Service (EPCDS) [9], amongst others.

In EPCglobal network, each physical product is associated with an RFID tag, represented by a unique EPC. This EPC can be retrieved from the RFID tags wirelessly via RFID readers as it transits between locations without contact-of-sight. These read events are usually processed by a middleware [8], and are stored locally at each supply chain partner's location-centric EPCIS [24]. With dynamic churn rates of partners and EPCIS, EPCDS becomes a unifying figure, helping partners locate information about a product in the supply chain. Through EPCglobal Network, participants can avoid information blackouts, and reaping the benefits of the RFID promise.

As the search and discovery component of EPCglobal network, EPCDS is designed with the intention of providing a bridge between supply chain partners, allowing them to share information, getting a step closer to achieving an automated supply chain. Due to the sensitivity and high value of the data transacted in EPCDS, a suitable access control mechanism is required. In this paper, we attempt to design and implement a secure and efficient EPCDS (SecDS) with an effective and efficient access control mechanism.

The road to achieve this is paved with the following challenges: (1) information transacted through EPCDS is constantly increasing, while churn rates of users is highly dynamic. This dynamism makes access control policies highly complex. (2) Each partner publishes information independently to EPCDS applying a myriad of access control policies. This disparate collection of access control policies in EPCDS makes it difficult to process, manage and maintain these policies effectively. Adding to this complexity, partners may not know of the existence of all participants in the supply chain. These made traditional access control mechanisms based on identity of users unsuitable. (3) As EPCDS is introduced to increase the visibility of RFID- related objects [9], it is important to support visibility policies (e.g. event information of an EPC is only allowed to be accessed by these partners who have handled the product with this

CODASPY'12, February 7–9, 2012, San Antonio, Texas, USA.
Copyright 2012 ACM 978-1-4503-1091-8/12/02 ...$10.00.

EPC). It is thus necessary to provide an efficient approach to specify and enforce these policies.

This paper presents the design and implementation of a secure EPCDS system — SecDS. Our contributions are summarized as follows:

- An extended attribute based access control (ABAC) model is proposed for SecDS that enriches the expressiveness of access control policies, while supporting visibility policies.
- We design and implement SecDS where this extended ABAC model is enforced without compromising on the efficiency of users' queries.

We begin with a description of background and motivations for our work in the following section. The extended ABAC model is presented in Section 3 and the implementation of SecDS is introduced in Section 4. Section 5 summarizes our experimental work and finally we introduce related works, conclude the paper, and describe future works.

2. BACKGROUND AND MOTIVATION

2.1 EPCglobal Network

As an important part of Internet of Things, EPCglobal network is a global standard for RFID supply chain networks providing a platform for trading partners to share product information [7]. As a participant of the EPCglobal Network, a company publishes event information of products into the EPCglobal Network, to share with each other. These information gives EPCglobal Network participants visibility of the location and movement of objects within supply chains.

The architecture of EPCglobal Network is described in the standard document [7], which is made up of many components, such as EPC Discovery services (EPCDS), EPC Information Services (EPCIS), and EPC Object Naming Services (EPC ONS). While EPCDS provides query service for serial-level information, EPC ONS provides search service for class-level information [9].

The information in EPCDS is published by EPCISes and searched by users. When a product with RFID tag passes through a supply chain, event information is obtained by RFID readers in each company and transmitted to its EPCIS. When an EPCIS captures event information about an EPC for the first time, it publishes this information into the EPCDS. When a user searches for information about a product with a given EPC, he first issues a query to EPCDS to get the addresses of EPCISes which handled the product with the given EPC. Next, the user queries each EPCIS by using the addresses returned by EPCDS to find the detailed event information. These processes are based on the Directory Look-up design [12], which SecDS complies to.

Based on the description above, we are aware that EPCDS mainly stores information published from EPCISes. When EPCDS uses relational databases as storage engine, the tables are illustrated in Table 1. In Table 1(a), the column **Time** represents when the product (with EPC) is handled, the column **PubisherId** represents the ID of the user who published the entry, and the column **CompanyId** represents the company associated with the record. Tables 1(b) and 1(c) store the information of users and companies. However, these tables only enumerate the basic attributes, while other additional attributes used to provide precise query are left out for clear presentations.

As shown in Table 1(a), EPCDS stores the information of when, where and what products are handled by a company, which expose business information of companies. To prevent unauthorized access of such sensitive information, a suitable access control mechanism should be implemented in the EPCDS, as highlighted in the corresponding standard and related research work [7, 15].

In Example 2.1, we list many access control policies which should be supported in EPCDS. This example will be used throughout this paper.

EXAMPLE 2.1. *Suppose there exist two production* Pro_1 *and* Pro_2 *respectively with EPCs urn:epc:id:sgtin:4049588:-083309.61157415873* [1] *and urn:epc:id:sgtin:4049588:083309.-89605325977. For production* Pro_1*, it comes from manufacturer M1 and is moved to distributor D2 and then shipped to retailer R1. Similar to* Pro_1*, production* Pro_2 *is passed through manufacturer M1, distributor D1, and retailers R1 and R2. All of these companies immediately published information (shown in Table 1(a)) to EPCDS when they handle these productions.*

However, the information in Table 1(a) cannot be released without any restriction. Different companies define different access control policies to protect their information. In the following, four representative policies are enumerated:

- pol_1 *(defined by security administrator of manufacturer M1): For the information about any product handled after 2011-01-01, it is allowed to access by the users of these companies who also handle this production and are distributor companies.*
- pol_2 *(defined by security administrator of distributor D1): For the information about any product whose EPC likes urn:epc:id:sgtin:4049588:083310.* (these production are valuable), it is only allowed to access by the users of manufacturer M1, distributor D1, and retailer R1 who are all partners.*
- pol_3 *(defined by security administrator of distributor D1): For the information about the product whose EPC doesn't like urn:epc:id:sgtin:4049588:083310.*, it is allowed to access by the users of these companies who also handle this production.*
- pol_4 *(defined by security administrator of retail R1): For the information about any product handled after 2011-03-01, it is allowed to access by the users of these companies who handle this production before the time R1 handles.*

2.2 Visibility Policy

For two different EPCISes a, b and an EPC e, the data in EPCDS reflects two possible relationships between a and b: they are both in the supply chain of e or they are not. These relationships can be used to specify access control policies. In Example 2.1, policies pol_1, pol_3 and pol_4 all use these relationships to specify access control policies, which

[1] The EPC number here is represented by a SGTIN notation that is why the "urn:epc:id:sgtin" is used, and then comes the company prefix(4049588), item reference(083309), and serial number (61157415873)

Table 1: Tables in EPCDS

(a) EPCDS-records

ID	EPC	Time	PublisherId	CompanyId
1	urn:epc:id:sgtin:4049588:083309.61157415873	2011-01-15 11:00	U1001	C101
2	urn:epc:id:sgtin:4049588:083309.89605325977	2011-01-20 14:30	U1001	C101
3	urn:epc:id:sgtin:4049588:083309.89605325977	2011-01-23 12:00	U1002	C102
4	urn:epc:id:sgtin:4049588:083309.61157415873	2011-02-01 10:00	U1003	C103
5	urn:epc:id:sgtin:4049588:083309.89605325977	2011-02-05 16:00	U1004	C104
6	urn:epc:id:sgtin:4049588:083309.61157415873	2011-02-10 15:30	U1004	C104
7	urn:epc:id:sgtin:4049588:083309.89605325977	2011-02-15 14:00	U1005	C105

(b) User-Companies

UserId	Name	CompanyId
U1001	Bob	C101
U1002	Andy	C102
U1003	John	C103
U1004	Peter	C104
U1005	Jack	C105

(c) Companies

CompanyId	Name	Role	URI
C101	M1	Manufacturer	http://www.m1.com
C102	D1	Distributor	http://www.d1.com
C103	D2	Distributor	http://www.d2.com
C104	R1	Retailer	http://www.r1.com
C105	R2	Retailer	http://www.r2.com

are expressed as "who also handle this product". Because the access control policies specified by these relationships directly affect the visibility of the location and movement of objects within supply chains, we call them *visibility policy.*

In [19], the authors also considered visibility policies, but their work is related to supply chain, and cannot be used in EPCDS. Following their work, we considered three kinds of visibility policies: *whole-stream policy*, *up-stream policy* and *down-stream policy.*

Definition 2.2. ***(whole-stream policy for EPC** e **defined by company** p) The event information published by company p of EPC e, is allowed to access by users of any companies who also publish event data of EPC e.*

Definition 2.3. ***(up-stream policy for EPC** e **defined by company** p) The event information published by company p of EPC e, is allowed to access by users of any companies who also publish event data of EPC e before the time when p publishes event data of EPC e.*

Definition 2.4. ***(down-stream policy for EPC** e **defined by company** p) The event information published by company p of EPC e, is allowed to access by users of any companies who also publish event data of EPC e after the time when p publishes event data of EPC e.*

Based on the schema of Table 1(a), for a record of an EPC e belonging to company c_1 which is protected by the whole stream policy, up-stream policy or down-stream policy, any user u_2 belonging to company c_2 is allowed to access this record if the following corresponding SQL predicate is satisfied:

- **whole-stream policy**:
  *exist (select * from EPCDS-records T where T.companyId* $= c_2$ *and T.EPC = e)*
- **up-stream policy**:
  *exist (select * from EPCDS-record T where T.companyId* $= c_2$ *and T.EPC = e and T.Time* $< t$*)*
- **down-stream policy**:
  *exist (select * from EPCDS-record T where T.companyId* $= c_2$ *and T.EPC = e and T.Time* $> t$*)*

where t is the time when c handles the EPC e.

Due to the complexity of visibility policy, traditional access control models (DAC, MAC, and RBAC) are not suitable for SecDS. In next section, we will extend attribute based access control model to protect the information in SecDS.

3. ATTRIBUTE-BASED ACCESS CONTROL FOR SECDS SYSTEM

Different from traditional access control models (DAC, MAC and RBAC), attribute based access control (ABAC) policies are defined based on attributes of subjects and objects [30].

3.1 Attributes

In SecDS, subjects are users while objects are event information in the table *EPCDS_records.*

- *Subject Attributes:* A subject is a user, who takes actions on event information in SecDS. Each subject is associated with a set of attributes which define the identity and characteristics of the subject. Such attributes may include the subject's identifier, name, country and so on. As a subject represents a company in SecDS, the attributes of a company are also considered as attributes of a subject belonging to it.
- *Object Attributes:* Objects are event information published from EPCISes. Naturally, object attributes will contain *EPC*, *Time*, and so on, which are the columns of table *EPCDS_records.*
- *Visibility Attribute:* To support visibility policy, we set visibility attribute which takes one of the following three values: *whole-stream*, *up-stream* and *down-stream.*

3.2 ABAC Authorization Language

An authorization language (AUL) is used to define who is allowed to perform what operations on what data. In this paper, we only focus on query operations; therefore AUL does not contain operations. The ABAC authorization language (AUL) is shown as follows:

AUL:=object_condition ∧ subject_condition
| object_condition ∧ visibility_condition
| object_condition ∧ subject_condition ∧ visibility_condition

where *subject_condition*, *object_condition* and *visibility_condition* are all boolean conditions for subject attributes, object attributes and visibility attributes respectively. All these conditions are constructed by the rules shown in Figure 1.

```
condition := expression | condition op condition
           | (condition op condition)
expression := attribute comp value |
              attribute comp attribute |
              attribute comp {value_set}
op := and | or
comp := < | > | =| <= | >= | [NOT] LIKE | [NOT] IN
value_set := value | value, value_set
```

Figure 1: Condition Language

Example 3.1. *The access control policies described in Example 2.1 are specified in Table 2 according to the authorization language in Figure 1.*

Expressiveness: The authorization language of SecDS is semantically richer and more expressive. Firstly, as an extension of ABAC, this AUL inherits the expressiveness from ABAC which encompasses the functionality of identity based access control, such as DAC and MAC, and RBAC [30]. Secondly, while conforming to the standards of EPCDS, this AUL supports visibility policies which are not considered in other access control models.

4. SECDS SYSTEM

4.1 Architecture of SecDS System

Before introducing the implementation approach, we first provide the architecture of SecDS as shown in Figure 2.

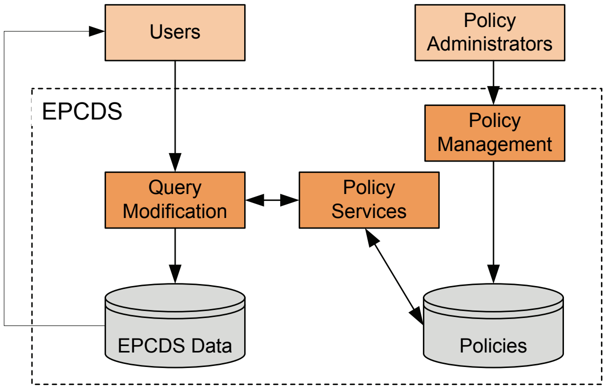

Figure 2: Architecture of SecDS system

SecDS contains the following components: Data Storage Server, Policies Storage Server, Policy Management, Policy Service, and Query Modification. Data Storage Server stores the event information published from EPCISes and some auxiliary information, while Policies Storage Server stores access control policies. Policy Management, Policy Service and Query Modification components are related to access control. We will focus on these three components in the following sections.

4.2 Policy Management

Policy Management Component (PMC) is used by policy administrator to manage access control policies. There are two different types of policies in SecDS system. First is global policy which is defined by the security administrator of SecDS. Second is local policy which is defined by the security administrator of each EPCIS. The local policy is used to control users' query. Other types of accesses from users, such as publishing data and querying data, are controlled under global policy.

Query operation is the most important access in SecDS, while access control policy over query operation is the most complicated policy. In the following, we only consider local policies which are defined by security administrators of different EPCISes to protect their published data.

For managing local policies, PMC provides services for security administrator of each EPCIS to create, modify and delete their access control policies. We take creation of access control policies as an example to illustrate the process in PMC.

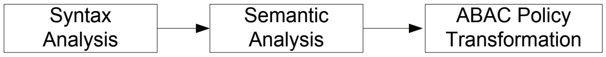

Figure 3: Process of creating policy in policy management component

As shown in Figure 3, there are three steps for creating policies in PMC: Syntax Analysis, Semantic Analysis, and ABAC Policy Transformation. The Syntax Analysis and Semantic Analysis make sure syntax and semantic of newly created access control policies are correct, which are similar to the corresponding components in most access control systems.

In order to reduce the cost of users' queries, ABAC policies are transformed into fine-grained access control (FGAC) policies which use SQL predicates to express users' privilege [1, 22, 25, 5, 28, 27]. In FGAC policy, it assigns a predicate to a user to express which data is allowed to access by the user. FGAC policy is a special case of ABAC policy where the attributes of object are the columns of relational tables and the attribute of subject is user's ID.

The transformation of an ABAC policy into FGAC policies can be taken in two steps. First, an ABAC policy is divided into three different predicates: subject predicate, object predicate and visibility predicate, which contain subject attributes, object attributes, and visibility attributes, respectively. Second, PMC finds all users in SecDS which satisfy the subject predicate, and then assigns the object predicate and visibility predicate to these users. The transformation is similar to the technique given in [17]. In the following, we take an example to further illustrate the ABAC policy transformation.

ID	Name	Predicate	Creator	CompanyId
1	pol_1	*Time* > 2011-01-01 ∧ (*Visibility* = whole-stream ∧ *Role* = Distributor)	C1001	C101
2	pol_2	*EPC* LIKE urn:epc:id:sgtin:4049588:083310:* ∧ *Name* IN (M1, D1, R1)	C1002	C102
3	pol_3	*EPC* NOT LIKE urn:epc:id:sgtin:4049588:083310:* ∧ *Visibility* = whole-stream	C1002	C102
4	pol_4	*Time* > 2011-03-01 ∧ *Visibility* = up-stream	C1004	C104

Table 2: ABAC policy table.

UserID	ABACPolicy	ObjectPredicate	Visibility	Creator	CompanyId
U1002	pol_1	TIME > 2011-01-01	whole-stream	U1001	C101
U1003	pol_1	TIME > 2011-01-01	whole-stream	U1001	C101
U1001	pol_2	EPC LIKE urn:epc:id:sgtin:4049588:083310:*	NULL	U1002	C102
U1002	pol_2	EPC LIKE urn:epc:id:sgtin:4049588:083310:*	NULL	U1002	C102
U1004	pol_2	EPC LIKE urn:epc:id:sgtin:4049588:083310:*	NULL	U1002	C102
0	pol_3	EPC NOT LIKE urn:epc:id:sgtin:4049588:083310:*	whole-stream	U1002	C102
0	pol_4	TIME > 2011-03-01	up-stream	U1004	C104

Table 3: FGAC policy table.

EXAMPLE 4.1. *Consider the tables in Table 1 and the ABAC policies in Example 3.1. Policy pol_1 can be divided into the following three parts:*

- *Subject Predicate: role = Distributor;*
- *Object Predicate: TIME > 2011-01-01;*
- *Visibility Predicate: visibility = whole-stream.*

Then, by using subject predicate, we construct the following SQL statement where the subject predicate is added in the WHERE clause:

```
SELECT UseId FROM User-Companies UC, Companies C
WHERE  UC.companyId = C.CompanyId and
       Role = Distributor;
```

After executing the above SQL statement the results {U1002, U1003} is returned. Then, the first two records in Table 3 are constructed. All policies in Example 3.1 are transformed into FGAC policies in the table 3. The users' IDs are both 0 *in the last two records in Table 3, which means these policies should be checked for all users' queries. When there is no subject predicate in an ABAC policy, the user ID in the record stored the transformed FGAC policy is* 0*.*

ABAC Policy Transformation has both advantages and disadvantages. It improves the efficiency of users' queries but increases the complexity of policy management. While supporting flexible and highly expressive access control policies, attribute based access control takes more time to make access decision. In ABAC system, it needs to determine whether a user satisfies an ABAC policy. This will take time if there are many ABAC policies and users as explained clearly in [17]. After transforming ABAC policies into FGAC polices, the cost of determining where a user satisfies an ABAC policy is relatively easy by checking the user's query against relevant FGAC policies. However, as stated in [17], the work of maintaining the consistence between ABAC policies and transformed FGAC policies is cumbersome since there are many situations to be taken into account such as values of users' attributes being changed, ABAC policies being added, deleted or updated, and users being added, deleted, or updated. We adapt the approaches proposed in [17] to solve these problems. In a nut shell, we improve the query performance at the cost of policy management.

4.3 Policy Service

The **Policy Service Component (PSC)** interacts the **Query Modification Component (QMC)** in access control. There are two types of services in PSC: FGAC Policy Searching Service (FPSS) and FGAC Policy Combining Service (FPCS). When a user issues a query, QMC knows which companies' data is required to access by this query and then sends the current user's ID and these companies' IDs to PSC. PSC first calls FPSS to search in the FGAC policy table for the FGAC policies that are assigned to this user and are created by the companies with the given companies' IDs. Then, PSC invokes FPCS to combine the returned FGAC policies into a single predicate before return it to QMC.

The implementation of FPSS is simple. First, when getting the current user's ID and many companies' IDs, FPSS constructs an SQL query to search in the FGAC policy table for all FGAC policies which are assigned to this user and created by these companies. Then FPSS sends these FGAC policies to FPCS.

Upon receiving a set of FGAC policies from FPSS, FPCS combines these policies into one single predicate. Before introducing combination algorithm, we first revisit visibility policy.

In section 2.2, we already mentioned that what SQL predicates are equal to whole-stream policy, up-stream policy and down-stream policy. However, for understanding easily, in those predicates we directly used the values of EPC and time. These predicates cannot be used in our query modification algorithm. So, we further transform these predicates into SQL predicates which can be directly used in query modification algorithm.

In the following SQL predicates, $T1$ represents the table same to T, which are both table *EPCDS_records*; u_1 represents the current user's ID who belongs to company c_1.

- **Whole-stream policy**:
  exist (select 1 from T1 where T1.companyId = c_1 *and T.EPC = T1.EPC)*

- **Up-stream policy**:
  exist (select 1 from T1 where T1.companyId $= c_1$ *and T.EPC = T1.EPC and T1.Time < T.Time)*

- **Down-stream policy**:
  exist (select 1 from T1 where T1.companyId $= c_1$ *and T.EPC = T1.EPC and T1.Time > T.Time)*

There are three steps taken to combine any FGAC policies.

Step 1: Each FGAC policy is transformed into one predicate:

- If an FGAC policy consists of a visibility predicate only, the visibility predicate is transformed into an EXIST SQL predicate as described above.
- If an FGAC policy consists of an object predicate as described above, there is no need of any transformation.
- If an FGAC policy is made up of a visibility predicate and an object predicate, the two predicates are combined into one predicate in the following steps. First, the visibility predicate is transformed into EXIST SQL predicate; then the object predicate is moved into the WHERE clause of the EXIST SQL predicate with connector AND.

Step 2: A predicate is added for each FGAC policy to ensure that this policy only protect the data of the company which creates this policy without affecting any other companies' data. The predicate is "companyId $= C_X$", which we call *own predicate*, where C_X is the ID of the company which creates this policy. Let pd_1 denote the predicate constructed in **step 1**. If pd_1 is an EXIST SQL predicate, the *own predicate* is added to the WHERE clause of pd_1 with connector AND; if pd_1 is just an object predicate, the *own predicate* is combined directly with pd_1 by using AND connector.

Step 3: Assume that after **step 1** and **step 2**, two policies are transformed into predicates pd_2 and pd_3, respectively. A connector OR is used to combine two FGAC policies in three cases:

- When pd_2 and pd_3 are both EXIST SQL predicates , the predicate belonging to WHERE clause of pd_2 is moved into the WHERE clause of pd_3 with OR connector.
- If one of predicates pd_2 and pd_3 is EXIST SQL predicate (assuming pd_2 is an EXIST SQL predicate, and pd_3 is a common predicate), pd_3 is moved into the WHERE clause of pd_2 with connector OR.
- If pd_2 and pd_3 are both common predicates , pd_2 and pd_3 are combined directly with connector OR.

The use of *own predicate* in **step 2** and OR connector in **step 3** ensures that all ABAC policies created by one EPCIS take no effect on any other companies' data. By using OR to combine two FGAC policies, it is potential to increase users' privileges while creating new policies [1]. For further restricting users' privilege, a security administrator can revise existing policies instead of writing a new one. The detailed combination algorithm is shown in Algorithm 1. The complexity of this algorithm is $O(N)$ where N is the number of policies which are combined.

Algorithm 1 Policy combination algorithm

```
Input: $poicySet : pSet$;
Output: $combinedpolicy : cp$
$cp$ = NULL
for $i = 1$ to $pSet.Length$ do
    $tempply = pSet[i]$;
    get object predicate $objp$ from $tempply$
    get visibility policy $visp$ from $tempply$
    get publisher ID $pubId$ from $tempply$
    combine $objp$ and $visp$ to form predicate $pred$
    form an own predicate using $pubId$ and add the formed own predicate to $pred$ with AND
    combine predicates $pred$ and $cp$ with OR and then give the combined predicate to $cp$
end for
return $cp$
```

Example 4.2. *Consider the FGAC policies in Table 3. Suppose the user with userId* $U1003$ *submits a query to search for the information about EPC = urn:epc:id:sgtin:40495-88:083309.89605325977 in Table 1(a). First, QMC gets the set of companies' IDs* $\{C101, C102, C104, C105\}$*, which publish the EPC information , and sends the set to FPSS. Then FPSS executes the following query to get FGAC policies:*

```
SELECT * From FGAC_TABLE WHERE (UserID = U1003 or
UserId = 0) AND CompanyId IN (C101, C102,
C104, C105);
```

In Table 3, UserId = 0 means all users' accesses should be controlled by this policy. The results are the second, sixth and seventh records in Table 3.

Then FPCS combines the three policies into one single predicate. For the second records in Table 3, the object predicate is TIME > 2011-01-01, and visibility is whole-stream. The visibility policy is transformed into EXIST predicate as follows:

```
EXIST (SELECT 1 FROM EPCDS_records T1 WHERE
T1.companyId = C103 and T.EPC = T1.EPC)
```

The object predicate and the own predicate companyId = $C101$ *are both added to the WHERE clause of the above predicate with AND so as to form the predicate item* 1 *shown below. The transformed predicates for the sixth and seventh records in Table 3 are shown in the following items* 2 *and* 3*, respectively. Finally, items* 1, 2 *and* 3 *are combined into the following item* 4 *as the final predicate.*

1. *pred1 = EXIST (SELECT * FROM EPCDS_records T1 WHERE (T1.CompanyId =* $C103$ *AND T.EPC = T1.EPC AND T.TIME > 2011-01-01) AND T.CompanyId =* $C101$*);*
2. *pred2 = EXIST (SELECT* $*$ *FROM EPCDS_records T1 WHERE (T1.CompanyId =* $C103$ *AND T.EPC = T1.EPC AND T.EPC NOT LIKE urn:epc:id:sgtin:404-9588:0083310:*) AND T.CompanyId =* $C102$*).*
3. *pred3 = EXIST (SELECT * FROM EPCDS_records T1 WHERE (T1.CompanyId =* $C103$ *AND T.EPC = T1.EPC AND T1.TIME < T.TIME AND T.TIME > 2011-03-01) AND T.CompanyId =* $C104$*).*

Algorithm 2 Query Modification Algorithm

```
Input: originalQuery : $Q$; userId;
Output: modifiedQuery : $Q'$
form a query to get the set comIdSet of company IDs
    who have the data queried by $Q$;
send comIdSet and userId to PSC to get the combined
    predicate pred
construct a view using pred and Table EPCDS_records
replace Table EPCDS_records with the view in $Q$ to get
    $Q'$
return $Q'$
```

4. *pred4 = EXIST (SELECT * FROM EPCDS_records T1 WHERE ((T1.CompanyId = C103 AND T.EPC = T1.EPC AND T.TIME > 2011-01-01) AND T.Publisher = C101) OR ((T1.CompanyId = C103 AND T.EPC = T1.EPC AND T.EPC NOT LIKE urn:epc:-id:sgtin:4049588:0083310:*) AND T.CompanyId = C-102) OR ((T1.CompanyId = C103 AND T.EPC = T1.EPC AND T1.TIME < T.TIME AND T.TIME > 2011-03-01) AND T.Company = C104)).*

4.4 Query Modification

The basic idea of query modification is that before being processed, user queries are transparently modified to ensure that users can access only what they are authorized to access [28]. Query modification is widely used in databases to enforce fine-grained access control which is also demonstrated as a scalable and efficient technique [22, 25, 28].

As aforementioned, ABAC policies in SecDS are transformed into FGAC policies, so that we can use query modification technique to enforce FGAC policies in SecDS.

Algorithm 2 shows our query modification algorithm. At the beginning, the original query is modified using "CompanyId" to replace the select target in the original query. The modified query is executed to return all company IDs who have the queried data. Then, the set of company IDs and the current user ID are sent to Policy Service Component to get a combined predicate. A temporary view is constructed using the returned predicate and *EPCDS_records*. Finally, the view is used to replace table *EPCDS_records* in the original query to form the finial modified query which will be executed instead of the original query.

EXAMPLE 4.3. *Suppose that user U1003 queries for the information about EPC urn:epc:id:sgtin:4049588:083309.896-05325977. The original query Q1 is constructed and modified into Q2 to get all companies whose data is requested to access. After sending the user's ID and the set of companies' IDs to Policy Services Component, PSC returns the combined predicate pred4 in Example 4.2. Finally, the returned predicate pred4 is used to form a view which replaces the table EPCDS_records to form the final modified query Q3.*

- *Q1: SELECT * FROM EPCDS_record WHERE EPC = urn:epc:id:sgtin:4049588:083309.89605325977;*
- *Q2: SELECT companyId FROM EPCDS_record WHERE EPC = urn:epc:id:sgtin:4049588:083309.89605325977;*
- *Q3: SELECT * FROM (SELECT * FROM EPCDS_record T WHERE pred4) WHERE EPC = urn:epc:id-:sgtin:4049588:083309.89605325977.*

Security: We assume that SecDS is a trusted server where a secure authentication mechanism is implemented. The query modification algorithm determines the security property of SecDS. In [28], a criterion with three properties is proposed for enforcing fine-grained access control policies in databases; one of these properties is *secure*. The *secure* property is defined as having no information leaked to adversaries under access control policy. It is also stated that the algorithm in [22], which constructs temporary views to replace the tables in user's query, is secure, because the information to which the user is not allowed to access, is filtered out in the constructed temporary views. As only row-level policies are used, our query modification algorithm is a special case of the algorithm given in [22], which supports both row-level and cell-level policies.

5. EXPERIMENTS

We implemented a prototype of SecDS system. Rigorous experiments conducted on the prototype show that SecDS is practical. The average query response time is about 260ms in a setting of 50, 000 supply chains, 300 EPCISes, on average 20 EPCISes being involved in each supply chain and on average 10 policies being evaluated for each query. Details of our experiments are given in an extended version of this paper.

6. RELATED WORK

EPCglobal network provides a platform for trading partners to share their product information [10], where EPCIS and EPCDS play an important role for increasing data visibility [11, 9]. There are lots of work focusing on how to implement EPCIS and EPCDS for providing traceability and increasing visibility [2, 13, 14, 20, 24]. However, the information in EPCIS and EPCDS is sensitive which should be protected.

In [15], the authors proposed and implemented a fine grained access control mechanism to protect the information in EPCIS, where query modification technique is used. However, the access control policy for EPCIS is simpler than that of EPCDS. This is because access control policies for an EPCIS are defined by the administrator of this EPCIS only, but the policies for EPCDS are defined by different security administrators of different companies. The access control approach designed for EPCIS cannot be directly applied for EPCDS.

Li el. at. proposed a semantic access control for RFID-enabled supply chains. Since it is not conform to the EPCglobal network, it cannot be used for EPCDS [23].

In [29], Yan et. al. considered an different situation where the EPCDS is an untrusted server, and they proposed a pseudonym-based design to mitigate the adversary's attack. Their work is also not conform to EPCglobal network and is not practical in real world applications.

Rigorous effort has been made in the research community on the security and privacy aspects of RFID systems e.g. [6, 18, 16]. However, these work mainly target at RFID communication systems, rather than EPC discovery services. To the best of our knowledge, SecDS is the first secure and efficient EPC discovery service system with suitable access control mechanism.

Since access control mechanism for EPCDS should support complex policies such as visibility policies, traditional access control models such as DAC, MAC, RBAC and TRBAC [4, 3, 26] may not suitable. Attribute based access control (ABAC) [21, 30] is adapted in this paper to meet the requirements of access control for EPCDS.

7. CONCLUSION

This paper described SecDS, a search engine based on EPCDS for EPCGlobal network. SecDS is not only efficient in processing users' search queries, but also secure and expressive in enforcing various data protection. We analyzed the requirements of access control for EPCDS and proposed an extended attribute based access control model to meet the requirements. In order to maintain efficiency, we proposed an approach of transforming ABAC policies to FGAC policies, and using query modification techniques to implement these FGAC policies. In future, we will consider how to further enhance the performance of SecDS.

8. REFERENCES

[1] R. Agrawal, P. Bird, T. Grandison, J. Kiernan, S. Logan, and W. Rjaibi. Extending relational database systems to automatically enforce privacy policies. In *ICDE*, pages 1013–1022, 2005.

[2] S. Beier, T. Grandison, K. Kailing, and R. Rantzau. Discovery services-enabling rfid traceability in epcglobal networks. In *COMAD*, pages 214–217, 2006.

[3] E. Bertino, P. A. Bonatti, and E. Ferrari. Trbac: A temporal role-based access control model. *ACM Trans. Inf. Syst. Secur.*, 4(3):191–233, August 2001.

[4] E. Bertino and R. Sandhu. Database security-concepts, approaches, and challenges. *IEEE Trans. Dependable Secur. Comput.*, 2:2–19, January 2005.

[5] S. Chaudhuri, T. Dutta, and S. Sudarshan. Fine grained authorization through predicated grants. In *ICDE*, pages 1174–1183, 2007.

[6] R. H. Deng, Y. Li, M. Yung, and Y. Zhao. A new framework for rfid privacy. In *ESORICS*, pages 1–18, 2010.

[7] EPCglobal. http://www.gs1.org/epcglobal.

[8] EPCglobal. Application level events (ale) standard. http://www.gs1.org/gsmp/kc/epcglobal/ale.

[9] EPCglobal. *Data Discovery (DD JRG) Requirements Document, 17 August, 2009, Version 0.0.25.*

[10] EPCglobal. Electronic product code. http://www.gs1.org/gsmp/kc/epcglobal/tds.

[11] EPCglobal. Epc information services. http://www.gs1.org/gsmp/kc/epcglobal/epcis.

[12] A. C. B. Evangelos A. Kosmatos, Nikolaos D. Tselikas. Intergrating rfids and smart objects into a unified internet of things architecture. *Advances in Internet of Things*, 1(1):5–12, 2011.

[13] S. Evdokimov, B. Fabian, S. Kunz, and N. Schoenemann. Comparison of discovery service architectures for the internet of things. In *Proceedings of the 2010 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing*, pages 237–244, 2010.

[14] C. Floerkemeier, M. Lampe, and C. Roduner. Facilitating rfid development with the accada prototyping platform. In *IEEE International Conference on Pervasive Computing and Communications*, 2007.

[15] E. Grummt and M. Müller. Fine-grained access control for epc information services. In *Proceedings of the 1st international conference on The internet of things*, IOT'08, pages 35–49, 2008.

[16] GSI. Rfid security & privacy lounge. http://www.avoine.net/rfid/index.php.

[17] S. Jahid, C. A. Gunter, I. Hoque, and H. Okhravi. Myabdac: compiling xacml policies for attribute-based database access control. In *CODASPY*, pages 97–108, 2011.

[18] A. Juels. Rfid security and privacy: a research survey. *IEEE Journal on Selected Areas in Communications*, 24(2):381–394, 2006.

[19] F. Kerschbaum. An access control model for mobile physical objects. In *SACMAT*, pages 193–202, 2010.

[20] C. Kürschner, C. Condea, O. Kasten, and F. Thiesse. Discovery service design in the epcglobal network: towards full supply chain visibility. In *Proceedings of the 1st international conference on The internet of things*, IOT'08, pages 19–34, 2008.

[21] B. Lang, I. T. Foster, F. Siebenlist, R. Ananthakrishnan, and T. Freeman. A flexible attribute based access control method for grid computing. *Journal of Grid Computing*, 7(2):169–180, 2009.

[22] K. LeFevre, R. Agrawal, V. Ercegovac, R. Ramakrishnan, Y. Xu, and D. DeWitt. Limiting disclosure in hippocratic databases. In *VLDB*, pages 108–119, 2004.

[23] Z. Li, C.-H. Chu, and W. Yao. Semantic Access Control for RFID-enabled Supply Chains. In *Workshop on RFID Security – RFIDSec Asia'10*, February 2010.

[24] J. Muller, J. Oberst, S. Wehrmeyer, J. Witt, A. Zeier, and H. Plattner. An aggregating discovery service for the epcglobal network. In *Proceedings of the 2010 43rd Hawaii International Conference on System Sciences*, pages 1–9, 2010.

[25] S. Rizvi, A. Mendelzon, S. Sudarshan, and P. Roy. Extending query rewriting techniques for fine-grained access control. In *SIGMOD*, pages 551–562, 2004.

[26] P. Samarati and S. D. C. di Vimercati. Access control: Policies, models, and mechanisms. In *FOSAD*, pages 137–196, 2000.

[27] J. Shi and H. Zhu. A fine-grained access control model for relational databases. *Journal of Zhejiang University - Science C*, 11(8):575–586, 2010.

[28] Q. Wang, T. Yu, N. Li, J. Lobo, E. Bertino, K. Irwin, and J.-W. Byun. On the correctness criteria of fine-grained access control in relational databases. In *VLDB*, pages 555–566, 2007.

[29] Q. Yan, R. H. Deng, Z. Yan, Y. Li, and T. Li. Pseudonym-based rfid discovery service to mitigate unauthorized tracking in supply chain management. In *International Symposium on Data, Privacy, and E-Commerce,*, pages 21–26, 2010.

[30] E. Yuan and J. Tong. Attributed based access control (abac) for web services. In *IEEE International Conference on Web Services*, pages 561–569, 2005.