8-2012

A secure and efficient discovery service system in EPCglobal network

Jie SHI

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

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

DOI: https://doi.org/10.1016/j.cose.2012.08.005

Citation

SHI, Jie; LI, Yingjiu; and DENG, Robert H.. A secure and efficient discovery service system in EPCglobal network. (2012). *Computers and Security*. 31, (8), 870-885. Research Collection School Of Information Systems.
Available at: https://ink.library.smu.edu.sg/sis_research/1636

A secure and efficient discovery service system in EPCglobal network

Jie Shi, Yingjiu Li, Robert H. Deng

School of Information Systems, Singapore Management University, 80 Stamford Road, Singapore

Published in Computers & Security, Volume 31, Issue 8, November 2012, Pages 870-885.

https://doi.org/10.1016/j.cose.2012.08.005

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. We design, implement SecDS and conduct extensive experiments on it. The results validate the practicality and cost effectiveness of our design and implementations.

Keywords

EPC discovery service, Access control, EPCglobal network, RFID, 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 (Kosmatos et al., 2011), 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 (EPCglobal, 2011a) is an important part of IOT. As a global standard RFID data sharing infrastructure, EPCglobal network is made up of Electronic Product Code (EPC) (EPCglobal, 2011c), EPC Information Services (EPCIS) (EPCglobal, 2011d), EPC Discovery Services (EPCDS) (EPCglobal, 2009), amongst others.

In EPCglobal network, each physical product is associated with an RFID tag, represented by an 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 (EPCglobal, 2011b), and are stored locally at each supply chain partner's location-centric EPCIS (Muller et al., 2010). 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 technology.

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 achieve 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 also handle the product with this EPC). It is thus necessary to provide an efficient approach to specify and enforce these policies.

Our contributions in this work are summarized as follows:

- We provide the requirements of access control for SecDS after analyzing existing literals and standard documents.
- 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.
- An extensive experiment is conducted to validate that SecDS is practical and cost-effective.

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 provides an evaluation of our implementation and finally we introduce related works, conclude the paper, and describe future works.

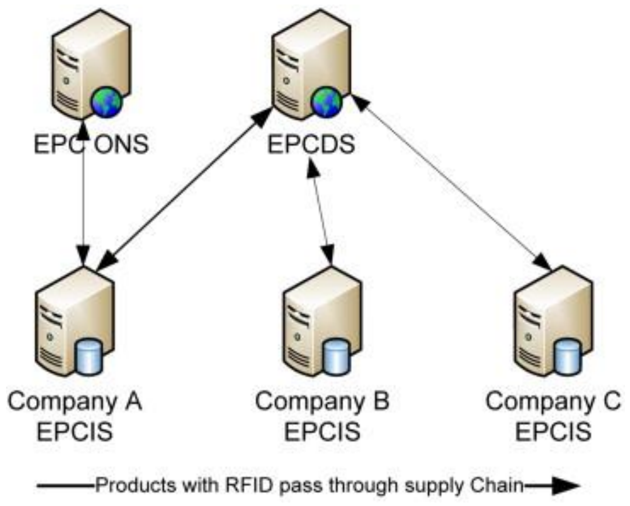
2. Background and motivation

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 (EPCglobal, 2011a). As participants of the EPCglobal Network, companies publish 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 products within supply chains.

The architecture of EPCglobal network is described in the standard document (EPCglobal, 2011a), which is made up of many components, such as EPC Discovery services (EPCDS), EPC Information Services (EPCIS), and EPC Object Naming Services (EPCONS). In order to make this paper self-contained, we provide a simple architecture of EPCglobal network as shown in Fig. 1. At the bottom of Fig. 1, there are several EPCIS servers, typically one per supply chain partner. At the top, there is an EPCDS server and an EPCONS server. While EPCDS provides query service for item-level information, EPCONS provides search service for class-level information (EPCglobal, 2009). An item-level information is related to an object while a class-level information is related to a bunch of objects which are in the same class.

Fig. 1. EPCglobal network architecture.

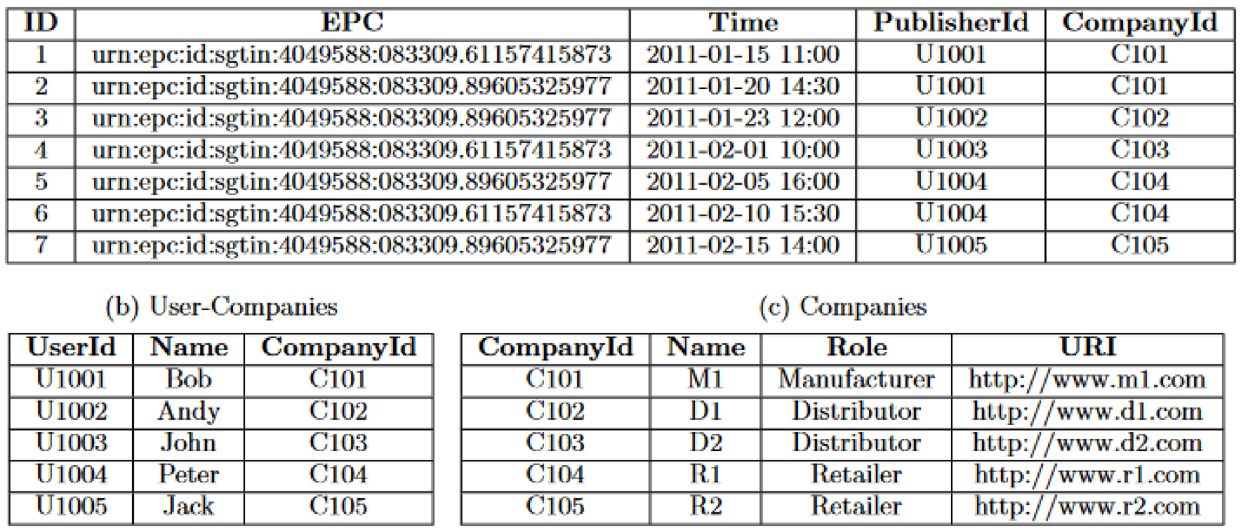


The information in EPCDS is published by supply chain partners and searched by users. When a product with an RFID tag passes through a supply chain, event information is captured by RFID readers in each company and transmitted to its EPCIS. When an event information about an EPC is captured for the first time, this information would be published into the EPCDS. In order to search detailed event information about a product with a given EPC, a user first issues a query to EPCDS to locate EPCISes which stores the detailed event information of the product with the given EPC. Next, the user queries these EPCISes to find the detailed event information. These processes are based on the “Directory Look-up Design” (Kosmatos et al., 2011), which SecDS complies to.

Based on the description above, we are aware that EPCDS mainly stores information about each EPC and the corresponding addresses of EPCISes. When EPCDS uses relational databases as storage engine, the table used to store these information likes Table 1(a), where column Time represents when the product (with an EPC) is handled, PubId column represents the ID of the user who publishes the entry, and the CompanyId column represents the company associated with the record. Table 1(b) and (c) store the information of users and companies. In order to simplify the description, we only illustrate the basic attributes in these tables while other additional attributes, such as record time, business step and disposition etc., used to provide precise query are not considered in this paper.

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

As shown in Table 1(a), EPCDS stores the information of when, where and what products are handled by a company, directly exposing business information of companies. To prevent unauthorized accesses to these sensitive information, a suitable access control mechanism should be supported in the EPCDS, as highlighted in the corresponding standard and related research work (EPCglobal, 2011a; Grummt and Müller, 2008).

In Example 2.1, we list four representative access control policies in EPCDS. These policies will be used throughout this paper.

Example 2.1

Suppose there exist two products Pro1 and Pro2 respectively with EPCs urn:epc:id:sgtin:4049588:083309.611574158731 and urn:epc:id:sgtin:4049588:083309.89605325977. For product Pro1, it comes from manufacturer M1 and is moved to distributor D2 and then shipped to retailer R1. Similar to Pro1, product Pro2 passes through manufacturer M1, distributor D1, and retailers R1 and

R2. All of these companies immediately publish information (shown in Table 1(a)) to EPCDS when they are handling these products.

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:

- pol1 (defined by security administrator of manufacturer M1): For the information about any products handled after 2011-01-01, it is allowed to be accessed by the users of these companies who also handle these products and are distributor companies.
- pol2 (defined by security administrator of distributor D1): For the information about any products whose EPC likes urn:epc:id:sgtin:4049588:083310.* (these products are valuable) it is only allowed to be accessed by the users of manufacturer M1, distributor D1, and retailer R1 who are all partners.
- pol3 (defined by security administrator of distributor D1): For the information about any products whose EPCs do not like urn:epc:id:sgtin:4049588:083310.*, it is allowed to be accessed by the users of these companies who also handle these products.
- pol4 (defined by security administrator of retail R1): For the information about any products handled after 2011-03-01, it is allowed to be accessed by the users of these companies who handle these products before the time when R1 handles.

2.2. Visibility policy

For two different EPCISes a, b and an EPC e, the data in EPCDS represent two possible relationships between a and b: they are both in the supply chain of e or they are not. These relationships are always used to specify access control policies. In Example 2.1, policies pol1, pol3 and pol4 all use these relationships to specify access control policies, which are expressed as “who also handle these products”. Because the access control policies based on these relationships directly affect the visibility of the location and movement of objects within supply chains, we call them visibility policy.

In Kerschbaum (2010), the authors also considered visibility policies. They provided the definition of visibility policy based on the trajectory of objects and enforced visibility policy using cryptographic authentication. Their enforcement requires storing signature information in RFID tag and does not consider the system architecture and data format in EPCglobal network, so their work does not conform to EPCglobal network, and also cannot be used in EPCDS. Following their work, we consider three kinds of visibility policies: whole-stream policy, up-stream policy and down-stream policy.

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

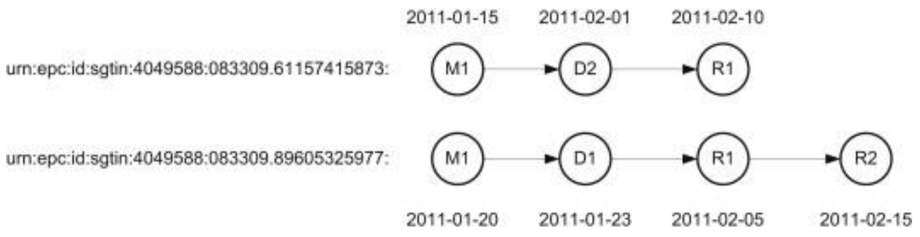
Definition 2.3. (**up-stream policy for EPC*e*defined by company***p*) The event data of EPC *e* published by company *p* is allowed to be accessed by users of any companies who also publish event data of EPC *e* before the time when *p* publishes event data of EPC *e*.

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

For example, there are two supply chains as shown in Fig. 2, which are both constructed from the data in Table 1(a). Consider the supply chain of EPC urn:epc:id:sgtin:4049588:083309.89605325977 in the bottom of Fig. 2. Suppose company R1 defines the above three visibility policies respectively and their meanings are explained as follows.

- whole-stream policy: Users of companies M1, D1, R1 and R2 are all allowed to access the event data of EPC urn:epc:id:sgtin:4049588:083309.89605325977 belonging to company R1, i.e. the fifth record in Table 1(a) is allowed to be accessed by users of companies M1, D1, R1 and R2;
- up-stream policy: Users of companies M1, D1 and R1 are all allowed to access the event data of EPC urn:epc:id:sgtin:4049588:083309.89605325977 belonging to company R1, i.e. the fifth record in Table 1(a) is allowed to be accessed by users of companies M1, D1 and R1;
- down-stream policy: Users of companies R1 and R2 are allowed to access the event data of EPC urn:epc:id:sgtin:4049588:083309.89605325977 belonging to company R1, i.e. the fifth record in Table 1(a) is allowed to be accessed by users of companies R1 and R2;

Fig. 2. Supply chains of Table 1(a).


2011-01-15
2011-02-01
2011-02-10
urn:epc:id:sgtin:4049588:083309.61157415873:
M1
D2
R1
urn:epc:id:sgtin:4049588:083309.89605325977:
M1
D1
R1
R2
2011-01-20
2011-01-23
2011-02-05
2011-02-15

Suppose there are two companies c1 and c2 in the same supply chain network. Based on the schema of Table 1(a), for any records of an EPC e belonging to company c1 which is protected by the whole stream policy, up-stream policy or down-stream policy, any user u2 belonging to company c2 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_1 handles the EPC *e*.

Visibility policy provides rich expressiveness of access control policies for EPCDS, but the enforcement of visibility policy is time-consuming. Visibility policy is based on the relationship of different companies which is reflected by the data in EPCDS, so it is necessary to query the data in EPCDS to check whether different companies satisfy the relationship. However, there is vast amount of data in EPCDS, which makes the cost of access decision of visibility policy too high. Therefore, it is a big challenge to provide an efficient enforcement of access control mechanism for EPCDS, where visibility policy is supported.

2.3. Requirements of access control for SecDS

After analyzing the requirements of access control for SecDS based on existing work and standard documents, we summarize the following four special requirements which are different from other access control systems.

1. Different companies' data must be protected by their own access control policies

The data in SecDS is published from different companies participating in different supply-chains. Different companies have their own security requirements which are reflected by different access control policies. Therefore, besides EPC event data, companies also publish access control policies into SecDS to control accesses to their event data. SecDS must make sure that users' accesses to different companies' data are controlled by the corresponding companies' policies. Meanwhile, SecDS also must make sure that users' accesses to the data of one company are only controlled under this company's policies, i.e. the policies published by one company take no effect on users' accesses to the data belonged to other companies.

2. Information sharing with unknown users in advance

EPCDS in EPCglobal network is designed to enhance the visibility of the movement of objects in supply chains. In nowadays global, complex and fast-changing supply chains, supply chain partners usually do not know the complete supply chain (Beier et al., 2006; Muller et al., 2010), i.e. supply chain partners are often not known in advance. For example, some products in Manufacture M1, which may be in China, are shipped by Distributer D1 to Logistics Provider L1, which may be in Singapore. Then these products are stored in Wholesaler W1 and shipped to Retailer R1. In these supply chains, the Manufacture M1 may not know the Retailer R1 in advance; and Retailer R1 may also not know Distributer D1 in advance. However, in order to achieve the full visibility of these products, which can be used to anti-counterfeiting and product recall (Kerschbaum and Oertel, 2010; Chaves and Kerschbaum, 2008), it is necessary to share information among these five supply chain partners. Note that, this is a very simple example; supply chains in the real world are more dynamic and complex. From another view, some companies may also want to share information to some unknown companies who are not in the same supply chains but may be potential collaborator. Therefore, while designing SecDS, we should consider the requirements — information sharing with unknown users in advance.

As we know, traditional access control (DAC, MAC and RBAC) cannot control accesses of users who is not known in advance; they are based on the users' identities (Yuan and Tong, 2005). These traditional access control models are not suitable for SecDS. However, attribute based access control supports the control of unknown users' accesses. In next section, we will present an extended attribute based access control for SecDS.

3. Visibility policy

We have already introduced visibility policy in detail in last section.

4. Efficiency

As a discovery service, the performance of SecDS is critical.

2.4. Threat model

In SecDS, we aim to prevent the accesses to event data from unauthorized users. That is, we protect event data in SecDS under the access control policies of different companies. We assume that SecDS is trusted and supports a secure authentication mechanism. The adversaries in our threat model mainly include the unauthorized users.

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 specified based on attributes of subjects and objects (Yuan and Tong, 2005).

3.1. Attributes

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

- Subject Attributes: A subject is a user, who takes actions on event data 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 all subjects belonged to the company. The attributes of a company are more important than the attributes of a user, because the event data in SecDS is shared among companies.
- Object Attributes: Objects are event data published from EPCISes. Naturally, object attributes 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 policy specification

Before giving the definition of ABAC policy, we first introduce authorization language. An authorization language (AUL) is used to specify who is allowed to perform what operations on what data. In this paper, we only focus on query operation; therefore AUL does not contain operation information.

The authorization language (AUL) is specified 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 attribute respectively. All these conditions are constructed by the rules shown in Fig. 3.

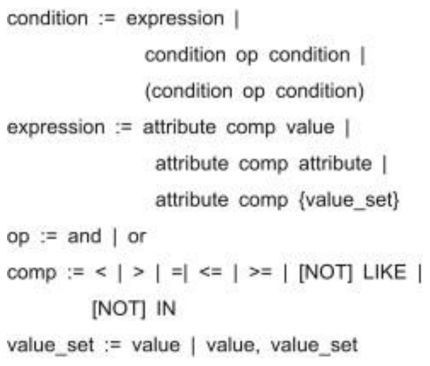
According to the description, each AUL is a function which takes the attribute values of subject and object as input parameters and outputs true or false. Given an AUL *l*, a subject *s* and an object *o*, *l(s, o)* outputs *true* when the attribute values of *s* and *o* satisfy *l*; otherwise, outputs *false*.

The access control policies in SecDS are defined by different companies to protect their own data. Thus, there are two main components in attribute-based access control policies: AUL and company.

Definition 3.1 (**ABAC policy**). Attribute-based access control policy consists of the following components:

- AUL: Authorization language;
- C: Companies who create the policy;

Fig. 3. Condition language.

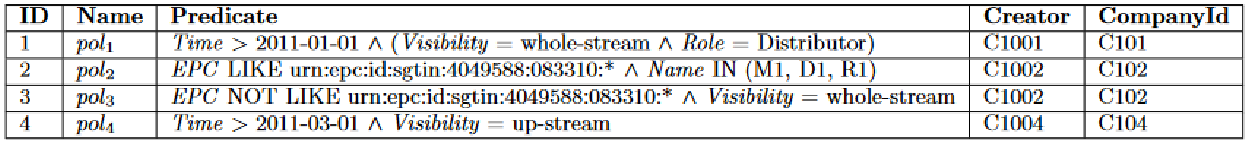


```
condition := expression |
             condition op condition |
             (condition op condition)
expression := attribute comp value |
              attribute comp attribute |
              attribute comp {value_set}
op := and | or
comp := < | > | =| <= | >= | [NOT] LIKE |
        [NOT] IN
value_set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

An ABAC policy *p* = (*l, c*), where c is a company who creates *p* to protect its data, and *l* is an authorization language, which specifies who can access what data.

Example 3.2. The access control policies described in Example 2.1 are specified respectively in Table 2 according to the above authorization language, where predicate column represents authorization language and companyId column represents company.

Table 2. ABAC policy table.



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

Expressiveness: The authorization language of SecDS is semantically richer and more expressive than that of traditional access control model, such as DAC, MAC and RBAC, due to the following reasons. First, as an extension of ABAC, this AUL inherits the expressiveness from ABAC which encompasses the functionalities of identity based access control, such as DAC, MAC and RBAC (Yuan and Tong, 2005). Second, while conforming to the standards of EPCDS, this AUL supports visibility policies which are not supported by other access control models.

3.3. Access decision

Attribute-based access control policies are specified by different companies to protect their data. SecDS enforces these policies by making access decision when receiving an access request. In the following, we first present a concept simple but non-efficient access decision approach, then we provide an equivalently efficient access decision approach.

Definition 3.3. (**Access Decision of single policy**). Given an ABAC policy $p = (l, c)$, a subject s and an object o, s is allowed to access o under the control of policy p if the following condition is true; otherwise the access is prohibited:

$l(s, o) = true \wedge o.owner = c$

where $o.owner$ represents the owner of object o.

In the above definition, $o.owner = c$ is used to guarantee that policy p is only used to protect the data belonged to company c, i.e. policy p has no affect to any accesses to other companies' data.

If there are multiple policies created by a company, we first use the following policy composition approach to combine these policies, then make access decision according to Definition 3.3.

Definition 3.4. (**Policy Composition of a company's policies**). Given ABAC policies $p_i = (l_i, c)$, $i \in [1 \ldots n]$, the combined policy $p = (l, c)$ where $l = (l_1 \vee l_2 \vee_{\ldots} \vee l_n)$.

In Definition 3.4, we use "OR" to combine policies. By using "OR" to combine two policies, it is potential to increase users' privileges while creating new policies (Agrawal et al., 2005). For further restricting users' privilege, a security administrator can revise existing policies instead of writing a new one.

According to Definitions 3.3 and 3.4, we can make access decision for any accesses to an object o from subject s in SecDS, even though there are lots of access control policies which are defined by different companies.

Definition 3.5. (**Access Decision**). Given ABAC policies $p_i = (l_i, c_i)$, $i \in [1 \dots n]$, the access decision of a query to objects $\{o_j\}$, $j \in [1 \dots m]$, from subject s takes the following three steps for each object o_j:

- Search all policies belonged to *oj.owner*;
- Combine these policies according to Definition 3.4;
- Make access decision according to Definition 3.3.

Note that in Definition 3.5, each object only has one owner based on which the access decision are made.

However, this approach is not efficient for users' queries in *SecDS* as there are many records related to each user's query which are belonged to different companies, i.e. there are many objects (records) about an EPC which are belonged to different companies. That is, we need to search policies and combine policies many times for one query. In the following, we provide an approach to combine multiple policies defined by different companies to one authorization language, which will be used in our enforcement to achieve better performance (detail in Section 4).

Definition 3.6. (**Policy composition**). Given policies $p_i = (l_i, c_i)$, $i \in [1 \dots n]$, the combined authorization language $l = (l_1 \wedge o.owner = c_1) \vee \dots \vee (l_n \wedge o.owner = c_n)$.

Definition 3.7. (**Efficient Access Decision**). Given ABAC policies $p_i = (l_i, c_i)$, $i \in [1 \dots n]$, the access decision of the query to objects from subject s takes the following two steps:

Search all policies belonged to the owners of $\{o_j\}$, $j \in [1 \dots m]$;

Get the combined authorization language l according to Definition 3.6;

For each object o_j, check whether $l(s, o_j)$ = true or not; if yes, o_j is allowed to be accessed by s; otherwise, o_j is prohibited to be accessed by s;

Example 3.8. Suppose there are five companies M1, D1, D2, R1 and R2. One product with EPC urn:epc:id:sgtin:4049588:093309.61157415873 passes M1, D2 and R1 as shown at the top of Fig. 2. Companies M1, D2 and R1 publish event data into SecDS respectively, as shown in Table 1 (the first, fourth and sixth records). Each of the five companies defines two policies denoted as

$$\{p_1^{\mathrm{M1}} = (l_1^{\mathrm{M1}}, \mathrm{M1}), p_2^{\mathrm{M1}} = (l_2^{\mathrm{M1}}, \mathrm{M1}), p_1^{\mathrm{D1}} = (l_1^{\mathrm{D1}}, \mathrm{D1}), p_2^{\mathrm{D1}} = (l_2^{\mathrm{D1}}, \mathrm{D1}), p_1^{\mathrm{D2}} = (l_1^{\mathrm{D2}}, \mathrm{D2}), p_2^{\mathrm{D2}} = (l_2^{\mathrm{D2}}, \mathrm{D2}), p_1^{\mathrm{R1}} = (l_1^{\mathrm{R1}}, \mathrm{R1}), p_2^{\mathrm{R1}} = (l_2^{\mathrm{R1}}, \mathrm{R1}), p_1^{\mathrm{R2}} = (l_1^{\mathrm{R2}}, \mathrm{R2}), p_2^{\mathrm{R2}} = (l_2^{\mathrm{R2}}, \mathrm{R2})\}$$

When a user u tries to search event data about this EPC, the first, fourth and sixth records in Table 1 are accessed. According to Definition 3.7, three steps are taken:

1) As these three records are belonged to companies M1, D2 and R1, the policies defined by M1, D2 and R1 are retrieved, i.e. policies $p_1^{\mathrm{M1}}, p_2^{\mathrm{M1}}, p_1^{\mathrm{D2}}, p_2^{\mathrm{D2}}, p_1^{\mathrm{R1}} \text{ and } p_2^{\mathrm{R1}}$.

2) According to Definition 3.6, a combined authorization language *l* is constructed as

$$l = \left(l_1^{\mathrm{M1}} \wedge o.owner = \mathrm{M1}\right) \vee \left(l_2^{\mathrm{M1}} \wedge o.owner = \mathrm{M1}\right) \vee \left(l_1^{\mathrm{D2}} \wedge o.owner = \mathrm{D2}\right)$$
$$\vee \left(l_2^{\mathrm{D2}} \wedge o.owner = \mathrm{D2}\right) \vee \left(l_1^{\mathrm{R1}} \wedge o.owner = \mathrm{R1}\right) \vee \left(l_2^{\mathrm{R1}} \wedge o.owner = \mathrm{R1}\right).$$

3) For each object o_j of the three records, if $l(u, o_j) = true$, the object o_j is allowed to be accessed by user u; otherwise, o_j is prohibited to be accessed.

Obviously, the result of using the approach in Definition 3.7 to make access decision is same as that of using the approach in 3.5. However, there are some advantages to use the approach in Definition 3.7: First, for a query searching event data about an EPC, we only need to retrieve policies and combine policies one time; Second, it makes possible to use query modification approach to achieve better performance (Detail in Section 4.4); Third, the policy composition in Definition 3.7 also guarantees that the access decision of any accesses to an object o is not affected by any ABAC policies created by other users, as each object in SecDS only has one owner.

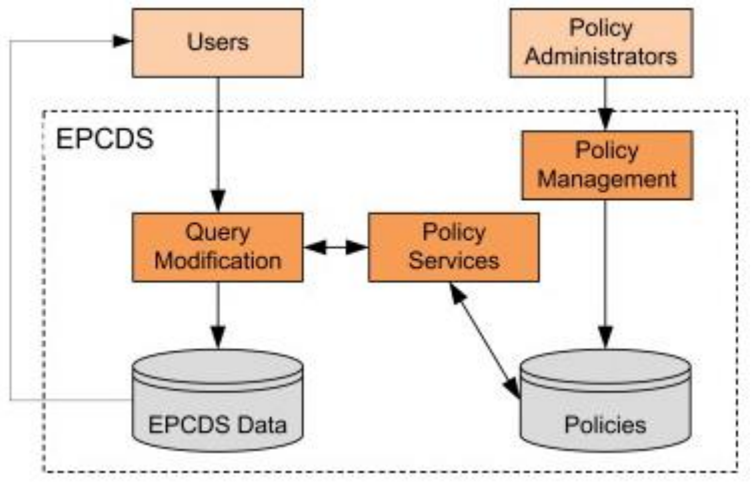
In next section, we will present the implementation of SecDS. In order to achieve efficiency, we mainly take three approaches. First, we transform attribute-based access control policy to fine-grained access control policy. Second, we use the approach in Definition 3.7 to combine policies. Third, we use query modification approach to enforce access control. In next section, all of these three approaches are described in detail.

4. Secure discovery service system

4.1. Architecture of secure discovery service system

Before introducing the detailed implementation, we first provide the architecture of SecDS as shown in Fig. 4.

Fig. 4. Architecture of secure discovery service system.



SecDS comprises the following components: Data Storage Server, Policy Storage Server, Policy Management, Policy Service, and Query Modification. Each of them is briefly presented as follows.

Data Storage Server: Data Storage Server stores event data published by supply chain partners, which is also what users want to access. The access control mechanism in SecDS is used to protect the event data in Data Storage Server from unauthorized users' accesses.

Policy Storage Server: There are two different types of access control policies in SecDS. First is global policy, which is defined by security administrators of SecDS to determine who is allowed to publish information into EPCDS; Second is local policy which is specified by security administrators of each supply chain partners protect their own event data. All policies are stored in the Policy Storage Server. When users access the event data in SecDS, the policies in Policy Storage Sever are selectively obtained by Policy Services Component and sent to Query Modification Component to control users' accesses.

Policy Management: Policy Management Component provides services for policy administrators to manage policies. When policy administrators create a policy, Policy Management Component processes it as follows: syntax analysis, semantic analysis and policy transformation. The syntax analysis and semantic analysis detect the syntax and semantic errors. Policy transformation, in order to improve the efficiency of users' queries, transforms attribute based access control policies into fine-grained access control policies.

Policy Service: Policy Service Component mainly serves the Query Modification Component. When a user issues a query to EPCDS, Query Modification Component sends the user's ID to Policy Service Component, then Policy Service Component finds the corresponding access control policies from Policies Storage Server, thereafter combining them into one policy which is returned to Query Modification Component.

Query Modification: Query Modification Component is the enforcement component of access control mechanism. It receives queries from users, obtains corresponding access control policies from Policy Service Component, and finally modifies users' queries and executes them. The modified queries ensures that information is not returned to unauthorized users.

The overall architecture of SecDS is not affected by the enforcement approaches used to combine access control policies or make access decision. However, the approaches of policy composition may affect the approaches used to make access decision. In SecDS, the query modification technique is used to enforce access control, which requires that the access control policies can be combined into one predicate. We realize that there are lots of policy composition approaches, such as conjunctive and disjunctive etc (Samarati and De Capitani di Vimercati, 2000; Bonatti et al., 2002). Therefore, in the future, we will further consider how to apply and enforce different policy composition approaches in SecDS.

In the following section, we will focus on the components related to access control, i.e. Policy Management, Policy Service and Query Modification.

4.2. Policy management

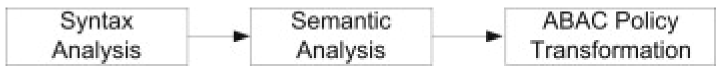
Policy Management Component (PMC) provides services for policy administrators to manage access control policies. There are two different types of policies in SecDS system. First is global policy which is defined by security administrators of SecDS. Second is local policy which is defined by security administrators of each supply chain partner. The local policies are used to control users' queries. Other types of accesses from users, such as publishing data and querying auditing data, are controlled under global policies.

Query is the most important access in SecDS which is under control of local policies defined by security administrators of different supply chain partners to protect their published data. In the following, we mainly focus on such local policies.

For managing local policies, PMC provides services for security administrators of each supply chain partner 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.

As shown in Fig. 5, there are three steps to create a policy 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 policy are correct, which are similar to the corresponding components in most access control systems.

Fig. 5. Process of creating policy in policy management component.



In order to reduce the cost of enforcing users' queries, ABAC policies are transformed into fine-grained access control (FGAC) policies where SQL predicates are used to express users' privilege (Agrawal et al., 2005; LeFevre et al., 2004; Rizvi et al., 2004; Chaudhuri et al., 2007; Wang et al., 2007; Shi and Zhu, 2010). In FGAC policy, it assigns a predicate to a user to specify which data is allowed to be accessed 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 only user's ID.

The transformation from 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 only contains subject attributes, object attributes and visibility attributes, respectively. Second, PMC searches these users in SecDS whose attribute values satisfy the subject predicate, then assigns the object predicate and visibility predicate to these users. The transformation is similar to the technique given in Jahid et al. (2011). In the following, we take an example to further illustrate the ABAC policy transformation, and analyze the merits and drawbacks of the transformation approach.

Example 4.1. Consider the tables in Table 1 and the ABAC policies in Example 3.2. 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.

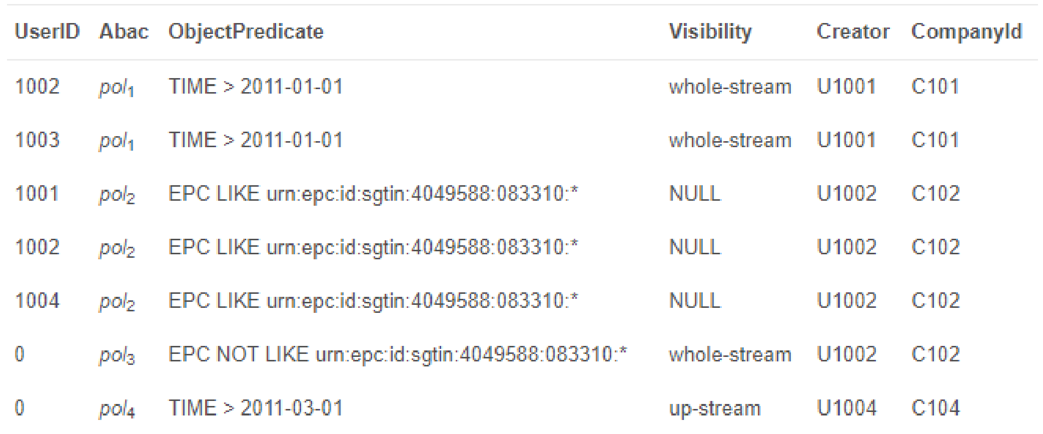
Then, by using subject predicate, we construct the following SQL statement where the subject predicate is appended into the WHERE clause:

```
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} are returned. Then, the first two records in Table 3 are constructed. According to this approach, all policies in Example 3.2 are then transformed into FGAC policies in 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 transformed FGAC policy is 0.

Table 3. FGAC policy table.



UserID	Abac	ObjectPredicate	Visibility	Creator	CompanyId
1002	pol_1	TIME > 2011-01-01	whole-stream	U1001	C101
1003	pol_1	TIME > 2011-01-01	whole-stream	U1001	C101
1001	pol_2	EPC LIKE urn:epc:id:sgtin:4049588:083310:*	NULL	U1002	C102
1002	pol_2	EPC LIKE urn:epc:id:sgtin:4049588:083310:*	NULL	U1002	C102
1004	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

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 Jahid et al. (2011). After transforming ABAC policies into FGAC polices, the cost of determining whether a user satisfies an ABAC policy is relatively low. However, as stated in Jahid et al. (2011), 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 Jahid et al. (2011) 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 with the Query Modification Component (QMC) in SecDS. There are two types of services in PSC: FGAC Policy Searching Service (FPSS) and FGAC Policies Combining Service (FPCS). When a user issues a query, QMC knows which companies' data is requested to be accessed by this query and then sends the current user's ID and these companies' IDs to PSC. PSC first calls FPSS to search FGAC policies which are assigned to this user and are created by these companies. Then, PSC invokes FPCS to combine the returned FGAC policies into a single predicate before returning it to QMC.

The implementation of FPSS is simple. First, when receiving a user's ID and many companies' IDs, FPSS constructs an SQL query to search 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 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 directly 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, and u1 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)

According to Definition 3.6, we combine FGAC policies by the following three steps.

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 only, 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 an 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 to each FGAC policy to ensure that this policy only protects the data belonged to the company who 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 who 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 the 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 belonged to WHERE clause of pd_2 is moved into the WHERE clause of pd_3 with OR connector.
- If one of these predicates pd_2 and pd_3 is an 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 policies created by one company take no effect on any other companies' data. 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: Policy Set: pSet;
Output: Combined policy: cp1: cp = NULL
2: for i = 1 to pSet.Length do
3:     tempply = pSet[i];
4:     get object predicate objp from tempply
5:     get visibility policy visp from tempply
6:     get publisher ID pubId from tempply
7:     combine objp and visp to form predicate pred
8:     form an own predicate using pubId and add the formed own predicate to pred with AND
9:     combine predicates pred and cp with OR and then give the combined predicate to cp
10: end for
11: return cp
```

Example 4.2. Consider these FGAC policies in Table 3. Suppose a user with user ID U1003 submits a query to search for information about an EPC urn:epc:id:sgtin:4049588:083309.89605325977 in Table 1(a). First, QMC gets the set of companies' IDs {C101, C102, C104, C105}, who publish event data about the EPC, and sends the set to FPSS. Then FPSS executes the following query to get FGAC policies which are assigned to this user and created by these companies:

SELECT * From FGAC_TABLE WHERE (UserID = U1003 or

UserId = 0) AND CompanyId IN (C101, C102, C104, C105);

In Table 3, "UserId = 0" means that all users' accesses should be controlled by this policy. The results of the above query 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 an 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 as 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:4049588:0083310:*) 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).

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 = C102) 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 (Wang et al., 2007). Query modification is widely used in databases to enforce fine-grained access control policy which is also demonstrated as a scalable and efficient technique (LeFevre et al., 2004; Rizvi et al., 2004; Wang et al., 2007).

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 by 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.

Algorithm 2. Query modification algorithm

```
Input: Original query: Q; User ID: userId;
Output: Modified Query: Q′
1: form a query to get the set comIdSet of company IDs who have the data queried by Q;
2: send comIdSet and userId to PSC to get the combined predicate pred
3: construct a view using pred and Table EPCDS_records
4: replace Table EPCDS_records with the view in Q to get Q′
5: return Q′
```

Example 4.3. Suppose that a user U1003 queries for the information about EPC urn:epc:id:sgtin:4049588:083309.89605325977. 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 *pred*4 is used to construct a view, which is used to replace the table *EPCDS_records* in the original query Q1 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.

4.5. Security analysis

In this section, we analyze security features and issues of SecDS.

We assume that SecDS is a trusted server where a secure authentication mechanism is implemented. We also assume that the attributes of subjects are correct, i.e. the attributes of subjects provided to SecDS are correct when subjects are registering or modifying their attributes. We note that in the real world, the administrator of SecDS can verify the attributes of subject in some extent. Companies and users should register before participating the EPCglobal network. When companies and users are registering, the attributes of subjects can be verified by administrator of SecDS. Sometimes, such verification is easy, since only the attributes of companies are usually used in policy because the event data is shared among companies, and the attributes of companies are usually static. The modification of companies' and users' attributes should also be under such verification. Although administrator of SecDS can provide such verification in some extent, it is still not an efficient approach. In the future, we are planning to provide an efficient and effective verification mechanism in SecDS. We also assume event data published into EPCDS is correct.

Due to the transformation of attribute-based access control policies into fine-grained access control policies, there is a window of access vulnerability which is the time between the change of privileges as the updating of subject's attributes and next policy transformation. In SecDS, the policy transformation is enforced immediately when the attributes of subject is updated. Thus, this window duration is very small as stated in Jahid et al. (2011).

The policy combination in Section 4.3 follows the policy composition approach in Section 3.3, although the policies are fine-grained access control policies which are still special attribute-based access control policies as mentioned before. Therefore, the policy combination approach can guarantee that the policies defined by one company have no affect to any accesses to other companies' data. That is, the attribute-based access control policies correctly represent the corresponding companies' security requirements.

We use query modification technique to enforce the transformed fine-grained access control policies, which directly determines the security of SecDS. In Wang et al. (2007), 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 LeFevre et al. (2004), which constructs temporary views to replace the tables in user's query, is secure, because the information 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 LeFevre et al. (2004), which supports both row-level and cell-level policies.

In short, the implementation of SecDS guarantee the accesses to companies' data are correctly controlled under their attribute-based access control policies. We will further analyze the efficiency of the implementation approach in next section.

5. Experiments

We implement a prototype of SecDS system. In these experiments, we aim to measure the cost of enforcing attribute-based access control in SecDS using the proposed approaches, i.e. we measure the performance of enforcing attribute-based access control comparing to that of no security mechanism. We also would like to study how the experimental parameters affect the performance of the enforcement. The parameters we consider are:

- *scNum*: The total number of supply chains;
- *epcisNum*: The total number of the EPCISes;
- *maxEcpisNum*: The maximum number of EPCISes in a supply chain;
- *minEcpisNum*: The minimum number of EPCISes in a supply chain;
- *per*: The percentage of the number of fine-grained access control policies. For each company, if there is a fine-grained access control policy for each of company in SecDS, there are total *epcisNum*^{2} access control policies. When the percentage is per, there are *per*epcisNum*^{2} access control policies in SecDS. Why we use per as a parameter? There are mainly two reasons. First, the parameter per directly determines the number of fine-grained access control policies in

SecDS. Second, there is a property in SecDS: the number of fine-grained access control policies is increasing with the increasing of the number of EPCISes when the number of attribute-based access control policies remains. The parameter per preserves this property.

In these experiments, we assume each supply chain partner or company has one EPCIS, so we interchangeably use EPCIS and supply chain partner (or company).

5.1. Experimental setup

In SecDS, a user submits an EPC of interest to query which EPCISes store the detailed event data about the EPC. Therefore, the query to SecDS has the following form:

SELECT * FROM EPCDS_records WHERE EPC = eid;

where the *eid* is the parameter input by users. To test the performance, we use queries following the above form. Besides the parameter *eid*, we also should know the current user's id *uid*. In order to get an accurate result, we should run as many queries as possible with random uid and eid. In our tests, we set the number of *eid* as 200 and the number of *uid* as 30. Therefore, for each result we ran 6000 queries with random *uid* and *eid*, and averaged the results.

There are two types of queries in our experiments:

- Original query: it inherits the form described above;
- Modified query: it is modified from original query by our query modification algorithm. Thus, the runtime of modified query includes the time of searching policies, combining policies, modifying query and the execution of the modified query.

SecDS is implemented based on web service, which is developed in C-Sharp and ASP.NET, using SQL SERVER 2005 as its database. We conduct the experiments on a desktop with 2.53 GHz Intel Core Duo CPU, 3 GB RAM and 240 GB hard drive. We do not take into account the performance of bandwidth.

5.2. Data setup

We generate random synthetic event data and access control policies so that our experiments are reproducible and they are not customized to specific EPC applications. Instead of producing attribute-based access control policies, we directly generate fine-grained access control (FGAC) policies in these experiments due to the following two reasons. First, as the query modification algorithm is based on fine-grained access control policies, we can observe how the percentage of the number of fine-grained access control polices affects the performance of the query modification algorithm by directly generating FGAC policies. Second, the cost of the transformation from ABAC policies to FGAC policies is acceptable which is demonstrated in Jahid et al. (2011), where the same transformation technique is used. Moreover, as SecDS is a search engine in EPCglobal network, the performance of query is critical, which is not determined by the transformation technique as policy transformation is not executed in the execution of users' queries.

Algorithm 3. EPCDS data generation algorithm

```
Input:scNum; EpcisNum; MaxEpcisNum; MinEpcisNum
1: randomly generate EpcisNum companies (EPCISes) and a user for each company
2: for i = 1 to scNum do
3:     randomly choose a number tempEpcisNum between MinEpcisNum and MaxEpcisNum
4:     randomly generate an EPC epc

5:   for each j = 1 to tempEpcisNum do
6:         randomly generate an unique ID id
7:         randomly generate a Time time
8:         randomly choose a user whose ID is userId
9:         randomly choose a company whose ID is companyId
10:        insert (id, epc, time, userId, companyId) in to table EPCDS_records
11:   end for
12: end for
```

In the following, we introduce the algorithms of generating data in table EPCDS_records and FGAC policies in detail.

Algorithm 4. Access control policy generation algorithm

```
Input: EpcisNum; per
1: randomly generate EpcisNum companies (EPCISes) and a user for each company
2: for each EPCIS whose ID is cid and his user ID is cuiddo
3:    for each user whose ID is uiddo
4:         randomly generate an policy name pName
5:         randomly choose 0 from {0,1} to nTemp with percent 1 – per
6:         if nTemp == 0 then
7:             continue;
8:         end if
9:         randomly generate an object predicate objp and a visibility predicate visp
10:        insert (uid, pName, objp, visp, cuid, cid) into FGAC policy table
11:    end for
12: end for
```

The data in table EPCDS_records is randomly generated by Algorithm 3. In this algorithm, there are four parameters: scNum, EPCISNum, maxEPCISNum, and minEPCISNum, which are introduced before. The data generation algorithm is processed as follows. First, EPCISNum companies are generated as well as a user for each company. Second, for all scNum supply chains, the data is generated by the following steps. (1) An EPC is randomly generated to represent this supply chain. (2) A number tempEPCISNum is chosen randomly between minEPCISNum and maxEPCISNum as the number of companies (EPCISes) in this supply chain. (3) For each EPCIS in these tempEPCISNum EPCISes, one record is constructed and inserted into table EPCDS_records. Namely, for each EPCIS, two random values of id and time are randomly generated, and a user and a company are randomly chosen from the previous generated

EPCISNum companies and users. Then a record is constructed and inserted into table EPCDS_records. The complexity of Algorithm 3 is *O*(scNum*(MinEPCISNum + MaxEPCISNum)).

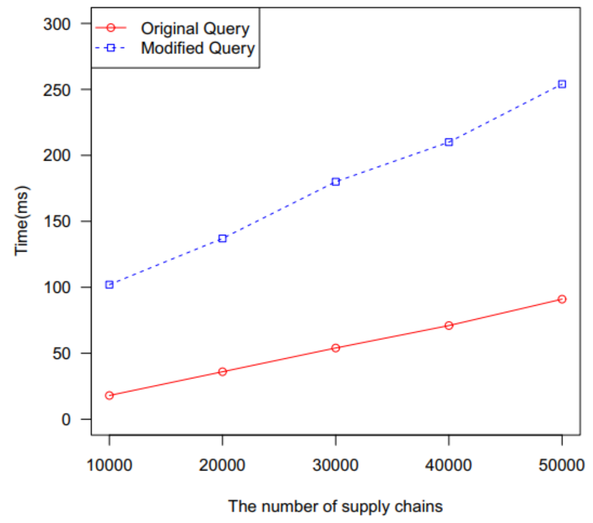
Algorithm 4 is for generating FGAC policies, which has two parameters: EPCISNum and per. If each company creates EPCISNum policies for all companies (one policy for one company), there are total $EPCISNum^2$ access control policies. Moreover, if a percentage per is chosen, there are $per*EPCISNum^2$ access control policies. For generating a FGAC policy, it is important to generate a random predicate, which is made up of many atomic predicates connected with "AND" or "OR". For each atomic predicate, an attribute is randomly chosen from {*EPC*, *Time*, *Visibility*}; a value is also randomly generated; and a comparison operator is randomly chosen to connect the attribute and the value. After generating a random predicate, a record for FGAC policy is constructed and inserted into FGAC policy table. The complexity of Algorithm 4 is $O(EPCISNum^2)$.

5.3. Experimental results and analysis

Our experiments study how the following factors may affect the performance of the query enforcement: the number of supply chains, the number of EPCISes, the number of EPCISes in a supply chain and the percentage of the number of fine-grained access control policies.

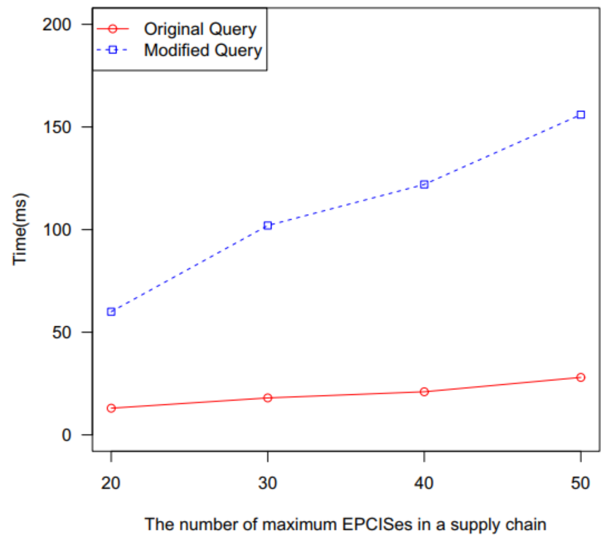
Our first set of experiments study how the number of supply chains may affect the performance of the query enforcement. From Fig. 6, we observe that, as the number of supply chains increases, the runtime of the original query and modified query both grows. The reason is that the total number of records in EPCDS_records (almost equal to (*maxEPCISNum* + *minEPCISNum*)**scNum*/2) grows as the number of supply chains increases. We also observe that, as the number of supply chains increases, the runtime of modified query grows faster than that of the original query. The reason is that the time of executing the modified query is affected more by increasing the number of records in table EPCDS_records, since the modified query may contains EXIST predicate as representation of visibility policy, which may be transformed into join query to be executed.

Fig. 6. The performance of varying maximum number of supply chains. The other parameters: the number of EPCISes: 300; the max number of EPCISes in a supply chain: 30; the percentage of access control policies: 50%.



Our second set of experiments study how the number of EPCISes in one supply chain may affect the performance of the query enforcement. From Fig. 7, we observe that the result is similar to that of increasing the number of supply chains, i.e. the runtime of original query and modified query both grows with the increasing of the number of EPCISes. The reason is that the total number of records in table EPCDS_records grows as the maximum number of EPCISes increases while the minimum number of EPCISes remains. From Fig. 7, we also observe that the runtime of modified query grows faster than that of the original query; and this phenomenon in Fig. 7 is more obvious than that in Fig. 6. The reason is that, the number of access control policies be executed for one query grows with the increasing of the number of EPCISes in one supply chain, i.e. the number of time-consuming visibility policies grows.

Fig. 7. The performance of varying max number of EPCISes in a supply chain. The other parameters: the number of supply chains: 10,000; the number of EPCISes: 300; the percentage of access control policies: 50%.



Our third and fourth sets of experiments illustrate that the performance of query enforcement is affected seriously as the number of access control policies increases. In the third experiment, while the number of EPCISes increases, the number of access control policies (almost equal to per*EPCISNum2) grows too. In the fourth experiment, while the percentage of the number of fine-grained access control policies increases, the number of policies also grows. From Figs. 8 and 9, we observe that the runtime of modified query grows very fast as the number of EPCISes or the percentage of the number of fine-grained access control policies increases. The reason is that, as the number of EPCISes or the percentage of the number of fine-grained access control policies increases, the number of access control policies be executed for each query grows, resulting in more time-consuming visibility policies being executed for each query.

Fig. 8. The performance of varying the number of EPCISes. The other parameters: the number of supply chains: 10,000; the max number of EPCISes in a supply chain: 30; the percentage of access control policies: 50%.

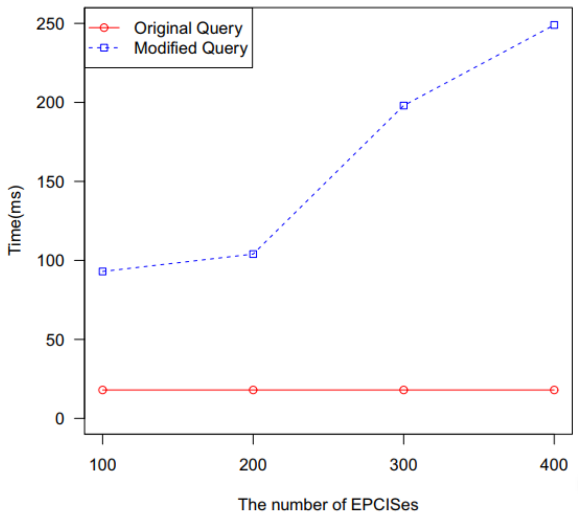
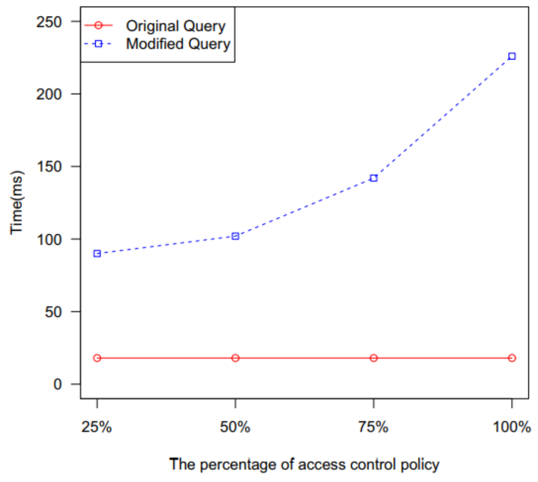


Fig. 9. The performance of varying the percentage of access control policies. The other parameters: the number of supply chains: 10,000; the number of EPCISes: 300; the max number of EPCISes in a supply chain: 30.



In short, while the query modification algorithm incurs reasonable cost, it is still acceptable, because the runtime is only 259 ms, even when there are 50,000 supply chains and 300 EPCISes, the maximum number of EPCISes in a supply chain is 30, the minimum number of EPCISes in one supply chain is 10, and the percentage of access control policies is 50%, as shown in Fig. 6.

6. Related work

For improving supply chain visibility, RFID-enabled supply chain network has drawn considerable attention from research and industrial community in recent years. Many track & trace systems are designed and implemented in the past decade, including IBM Theseos (Cheung et al., 2007), DIALOG (Främling and Nyman, 2009), and the system developed in BRIDGE project (Bridge, 2007). However, the problem of how to efficiently and effectively share supply chain information among different partners is still not solved as intentions of designing and implementing these systems are different. A standard supply chain network is very important to facilitate information sharing among supply chain partners; then EPCglobal network turns out.

As a global standard RFID data sharing infrastructure, EPCglobal network is primarily made up of EPCIS, EPCDS, and EPCONS (EPCglobal, 2011a), where EPCIS and EPCDS play an important role for increasing the visibility of objects in supply chains (EPCglobal, 2011d, 2009). For providing traceability and increasing visibility, there are lots of work about design and implementation of track & trace systems following the standard EPCglobal network (Beier et al., 2006; Evdokimov et al., 2010; Kürschner et al., 2008; Muller et al., 2010; Manzanares-Lopez et al., 2011; Muñoz Gea et al., 2010). However, security problems are not taken into account in these works, which are very important for EPCglobal network, as the information flowing in EPCglobal network is sensitive and valuable.

Recently, many security mechanisms are proposed to protect the information in different components of EPCglobal network, i.e. EPCIS, EPCDS and EPCONS. For protecting the information in EPCIS, a fine-grained access control mechanism is proposed and implemented (Grummt and Müller, 2008), where the security administrators of EPCISes create fine-grained access control policies to authorize different privileges to different users. Query modification technique is used to enforce fine-grained access control policies. However, their work is not suitable for EPCDS due to the following reasons: first, the data in EPCDS is published from different companies which should be protected by different policies while in their work the data in an EPCIS only belongs to one supply chain partner and is protected by the policies defined by the security administrator of the EPCIS; second, the users in EPCDS may be not known in advance for each security administrator while fine-grained access control is used in their work which requires that all users are known in advance; third, visibility policies are not considered in their work. All of these reasons are our challenges and contributions which make our work different from their work. For protecting the information in EPCONS, a peer-to-peer name service architecture is proposed to enhance the data security of EPCONS and the privacy of the users. However, their work is based on peer-to-peer system which make their work not fully conform to the standard EPCglobal network. Following the work of EPCONS, a privacy-enhanced discovery service is proposed to protect the privacy of users in EPCDS (Fabian et al., 2011), which is also based on peer-to-peer system. Yan et al. (2010) considered a 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 the standard EPCglobal network and is not practical in real world applications. Shi et al. (2012) proposed a secure track and trace system which is based on EPCglobal network. In their work, they proposed a new approach to enhance the security of EPC discovery service system based on “relay model”. However, as they mentioned, the EPCDS and EPCIS in their work do not fully conform to the standard EPCglobal network.

Besides the above work, there are some works which focus on security of RFID-enabled supply chain. Li et al. (2010) proposed a semantic access control for RFID-enabled supply chains. In Fu and Li (2011), a role-based authorization framework is presented for RFID-enabled supply chain network. However, these works do not conform to the standard EPCglobal network, and cannot be used for EPCDS.

Rigorous effort has been made in the research community on the security and privacy aspects of RFID systems (e.g. Deng et al., 2010; Juels, 2006; GSI, 2011). However, these works 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 which conforms to the standard EPCglobal network.

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 (Bertino and Sandhu, 2005; Bertino et al., 2001; Samarati and De Capitani di Vimercati, 2000) may be not suitable. Attribute based access control (ABAC) (Lang et al., 2009; Yuan and Tong, 2005) 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.

Acknowledgments

The authors thank the anonymous reviewers for valuable comments and suggestions. This work is supported in part by the Office of Research at Singapore Management University.

References

Agrawal R, Bird P, Grandison T, Kiernan J, Logan S, Rjaibi W. Extending relational database systems to automatically enforce privacy policies. In: Proceedings of the 21st international conference on data engineering. Washington, DC, USA: IEEE Computer Society; 2005. p. 1013-22. ICDE ’05.

Beier S, Grandison T, Kailing K, Rantzau R. Discovery services enabling RFID traceability in EPCglobal networks. In: COMAD; 2006. p. 214-7.

Bertino E, Sandhu R. Database security-concepts, approaches, and challenges. IEEE Transactions on Dependable and Secure Computing 2005; 2:2-19.

Bertino E, Bonatti PA, Ferrari E. Trbac: a temporal role-based access control model. ACM Transactions on Information and System Security 2001; 4:191-233.

Bonatti P, De Capitani di Vimercati S, Samarati P. An algebra for composing access control policies. ACM Transactions on Information and System Security 2002; 5(1):1-35.

Bridge. High level design for discovery services. Technical Report, Bridge Project; 2007.

Chaudhuri S, Dutta T, Sudarshan S. Fine grained authorization through predicated grants. In: ICDE; 2007. p. 1174-83.

Chaves LWF, Kerschbaum F. Industrial privacy in RFID-based batch recalls. In: Proceedings of the 2008 12th enterprise distributed object computing conference workshops. Washington, DC, USA: IEEE Computer Society; 2008. p. 192-8. EDOCW ’08.

Cheung A, Kailing K, Schonauer S. Theseos: a query engine for traceability across sovereign, distributed rfid databases. In: ICDE; 2007. p. 1495-6.

Deng RH, Li Y, Yung M, Zhao Y. A new framework for RFID privacy. In: Proceedings of the 15th European conference on research in computer security. Berlin, Heidelberg: Springer-Verlag; 2010. p. 1-18. ESORICS ’10.

EPCglobal. Data discovery (dd jrg) requirements document; 2009.

EPCglobal, http://www.gs1.org/epcglobal; 2011a.

EPCglobal. Application level events (ale) standard, http://www.gs1.org/gsmp/kc/epcglobal/ale; 2011b.

EPCglobal. Electronic product code, http://www.gs1.org/gsmp/kc/epcglobal/tds; 2011c.

EPCglobal. EPC information services, http://www.gs1.org/gsmp/kc/epcglobal/epcis; 2011d.

Evdokimov S, Fabian B, Kunz S, Schoenemann N. 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. Washington, DC, USA: IEEE Computer Society; 2010. p. 237-44. SUTC ’10.

Fabian B, Ermakova T, Muller C. Shardis: a privacy-enhanced discovery service for RFID-based product information. IEEE Transactions on Industrial Informatics 2011; 8(3):707-18.

Framling K, Nyman J. From tracking with RFID to intelligent products. In: Proceedings of the 14th IEEE international conference on emerging technologies & factory automation. Piscataway, NJ, USA: IEEE Press; 2009. p. 1418-25. ETFA ’09.

Fu G, Li Y. A role-based authorization framework for RFID-enabled supply chain networks. In: The 16th international conference on transformative science, engineering, and business innovation; 2011.

Grummt E, Muller M. Fine-grained access control for EPC information services. In: Proceedings of the 1st international conference on the internet of things. Berlin, Heidelberg: Springer-Verlag; 2008. p. 35-49. IOT ’08.

GSI. RFID security & privacy lounge, http://www.avoine.net/rfid/index.php; 2011.

Jahid S, Gunter CA, Hoque I, Okhravi H. Myabdac: compiling xacml policies for attribute-based database access control. In: Proceedings of the first ACM conference on data and application security and privacy. New York, NY, USA: ACM; 2011. p. 97-108. CODASPY ’11.

Juels A. Rfid security and privacy: a research survey. IEEE Journal on Selected Areas in Communications 2006; 24(2):381-94.

Kerschbaum F. An access control model for mobile physical objects. In: Proceeding of the 15th ACM symposium on access control models and technologies. New York, NY, USA: ACM; 2010. p. 193-202. SACMAT ’10.

Kerschbaum F, Oertel N. Privacy-preserving pattern matching for anomaly detection in RFID anti-counterfeiting. In: Proceedings of the 6th international conference on radio frequency identification: security and privacy issues. Berlin, Heidelberg: Springer-Verlag; 2010. p. 124-37. RFIDSec ’10.

Kosmatos EA, Tselikas ND, Boucouvalas AC. Intergrating RFIDs and smart objects into a unified internet of things architecture. Advances in Internet of Things 2011; 1:5-12.

Kurschner C, Condea C, Kasten O, Thiesse F. Discovery service design in the EPCglobal network: towards full supply chain visibility. In: Proceedings of the 1st international conference on the internet of things. Berlin, Heidelberg: Springer-Verlag; 2008. p. 19e34. IOT ’08.

Lang B, Foster IT, Siebenlist F, Ananthakrishnan R, Freeman T. A flexible attribute based access control method for grid computing. Journal of Grid Computing 2009; 7:169-80.

LeFevre K, Agrawal R, Ercegovac V, Ramakrishnan R, Xu Y, DeWitt D. Limiting disclosure in hippocratic databases. In: Proceedings of the thirtieth international conference on very large data bases, vol. 30. VLDB Endowment; 2004. p. 108-19. VLDB ’04.

Li Z, Chu CH, Yao W. Semantic access control for rfid-enabled supply chains. In: Workshop on RFID security e RFIDSec Asia ’10. Cryptology and information security, vol. 4. Singapore, Republic of Singapore: IOS Press; 2010. p. 95-108.

Manzanares-Lopez P, Munoz Gea JP, Malgosa-Sanahuja J, Sanchez-Aarnoutse JC. An efficient distributed discovery service for EPCglobal network in nested package scenarios. Journal of Network and Computer Applications 2011; 34: 925-37.

Muller J, Oberst J, Wehrmeyer S, Witt J, Zeier A, Plattner H. An aggregating discovery service for the EPCglobal network. In: Proceedings of the 2010 43rd Hawaii international conference on system sciences. Washington, DC, USA: IEEE Computer Society; 2010. p. 1-9. HICSS ’10.

Munoz Gea JP, Malgosa-Sanahuja J, Manzanares-Lopez P, Sanchez-Aarnoutse JC. Implementation of traceability using a distributed RFID-based mechanism. Computers in Industry 2010; 61:480-96.

Rizvi S, Mendelzon A, Sudarshan S, Roy P. Extending query rewriting techniques for fine-grained access control. SIGMOD ’04. In: Proceedings of the 2004 ACM SIGMOD international conference on management of data. New York, NY, USA: ACM; 2004. p. 551-62.

Samarati P, De Capitani di Vimercati S. Access control: policies, models, and mechanisms. In: FOSAD; 2000. p. 137-96.

Shi J, Zhu H. A fine-grained access control model for relational databases. Journal of Zhejiang University - Science C 2010; 11(8):575-86.

Shi J, Li Y, He W, Sim D. Sectts: a secure track & trace system for RFID-enabled supply chains. Computers in Industry 2012; 63(6): 574-85.

Wang Q, Yu T, Li N, Lobo J, Bertino E, Irwin K, et al. On the correctness criteria of fine-grained access control in relational databases. In: Proceedings of the 33rd international conference on very large data bases. VLDB Endowment; 2007. p. 555-66. VLDB ’07.

Yan Q, Deng RH, Yan Z, Li Y, Li T. Pseudonym-based RFID discovery service to mitigate unauthorized tracking in supply chain management. International Symposium on Data, Privacy, and E-Commerce 2010; 0:21-6.

Yuan E, Tong J. Attributed based access control (abac) for web services. IEEE International Conference on Web Services 2005; 0:561-9.

Jie Shi received the B.S. degree from Hefei University of Technology in 2006 and received his Ph.D. degree from Huazhong University of Science and Technology in 2010. He is currently working as a research fellow in Singapore Management University. His current research interests are security and privacy of EPCglobal network, data and applications security.

Yingjiu Li is currently an Associate Professor in the School of Information Systems at Singapore Management University. He received his Ph.D. degree in Information Technology from George Mason University in 2003. His research interests include RFID security, applied cryptography, and data applications security. He has published over 70 technical papers in international conferences and journals. He has served in the program committees for over 50 international conferences and workshops. Yingjiu Li is a senior member of the ACM and a member of the IEEE.

Robert H. Deng is currently a professor, associate dean for Faculty and Research, School of Information Systems at Singapore Management University. He received his Ph.D. degrees from the Illinois Institute of Technology. He has more than 200 technical publications in international conferences and journals in the areas of computer networks, network security, and information security. He has served as general chair, program committee chair, and program committee member of numerous international conferences. He is an Associate Editor of the IEEE Transactions on Dependable and Secure Computing, Associate Editor of Security and Communication Networks Journal (John Wiley).