Interoperable Internet-Scale Security Framework for RFID Networks

by

Tingting Mao

B.S., Wuhan University (2002)
S.M., Peking University (2004)
S.M Massachusetts Institute of Technology(2007)

Submitted to the Department of Civil and Environmental Engineering in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in the field of Information Technology

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2008

Signature of Author . . . .
Department of Civil and Environmental Engineering
August 15, 2008

Certified by . . . .
John R. Williams
Associate Professor of Civil and Environmental Engineering
Thesis Supervisor

Certified by . . . .
Steven R. Lerman
Professor of Civil and Environmental Engineering
Thesis Supervisor

Accepted by . . . .
Daniele Veneziano
Chairman, Departmental Committee for Graduate Students

Interoperable Internet-Scale Security Framework for RFID Networks

by

Tingting Mao

Submitted to the Department of Civil and Enviromental Engineering
on August 15, 2008, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

It is estimated that over 3 billion Radio Frequency Identification (RFID) tags have been deployed through 2007. Most tags are used in supply chains where the Electronic Product Code (EPC) and associated business event data are transmitted through RFID networks. Security and privacy issues are critically important in RFID networks because EPC data and their associated business events are valuable assets. Companies need to share these data with restricted business partners and, under some conditions, such as product recall, more widely with regulators and non business partners. At present, no security or privacy framework has been chosen as an EPCglobal standard(industry-driven standards for EPC) due to the difficulty of sharing information between parties who have no direct business relationships and hence no business rules for sharing these data. To date, no security schemes have been deployed that can support data exchange with multiple identity techniques and interchangeable complex business rules, as required by RFID networks.

In this thesis, an Interoperable Internet-Scale Security (IISS) framework for RFID networks is proposed. The IISS framework performs authentication and authorization based on an aggregation of business rules, enterprise information, and RFID tag information. IISS provides a protocol for several authentication schemes and identity techniques. It also provides an engine for reasoning over business rules from different domains. Moreover, the IISS framework is able to resolve provenance information of RFID tags, which can identify the history of a particular piece of EPC data through the supply chain. The IISS framework and the IISS ontologies to model the information in RFID networks are also described, and how the IISS framework can be developed for access control in RFID enabled supply chains is discussed. Finally, the IISS framework's efficiency is tested using a supply chain EPC simulator as the testing platform, which allows optimization of the IISS protocol's performance.

Thesis Supervisor: John R. Williams
Title: Associate Professor

Thesis Supervisor: Steven R. Lerman
Title: Professor

Acknowledgments

First and foremost, I would like to thank my advisors, Professor John R.Williams and Professor Steven R.Lerman. I wish to thank Professor John R. Williams for his significant support and inspiration, for his invaluable insights, his dedication in spending time in discussion with me, and for his advise duing my research. I want to thank Professor Steven R. Lerman for his endless support, generous help, great encouragement, and invaluable direction throughout my four years Ph.D program.

I also want to thank my thesis committee member Research Scienctist Dr. Abel Sanchez for his invaluable insights, for endlessly pushing me to find better ways to tackle the problems in this thesis.

Special thanks go to my parents and my sister for their love, encouragement, especially in the hardest time. I also want to thank them for their great support of my education.

I also want to thank Shane Markstrum for being patient with me over the past two years, and for his help in revising my thesis.

I would like to thank my fellow graduate students Jin Hock Ong, Sergio Herrero, Rahul Bhattacharyya, Miao Sun, Anne Fabo for their friendship, and the discussion we had.

I also would like to thank all my friends at MIT for their friendship.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Since Radio Frequency Identification (RFID) was invented in 1946, RFID technology has had widespread application, including electronic passports, transportation payments, and inventory systems. It is estimated that over 3 billion RFID tags have been deployed through 2007. As RFID deployment continues to increase, its associated applications have become more and more network-centric. Most tags are used in supply chains where the Electronic Product Code (EPC) and associated business event data are transmitted through RFID networks. The design of RFID network thus plays a critical role in successfully deploying large-scale applications. This is especially true when the tagged object is item-level. For example, it is estimated that Wal-Mart's in-store implementation of RFID technology will generate about 7.5 terabytes of RFID data per day [59].

Without adequate security, unauthorized parties can potentially read RFID tags. EPC event data are valuable corporate assets. A company is likely to only share them with certain trading partners under special conditions. According to a study conducted by the Ponemon Institute, data breaches cost businesses an average of $197 per customer record in 2007[12].

Thus, security and privacy issues are critically important for RFID networks. Companies need to share these data with a restricted set of their business partners and, in some cases, such as product recall, more widely with regulators and non-business partners. For example, governments mandate to report and share data.

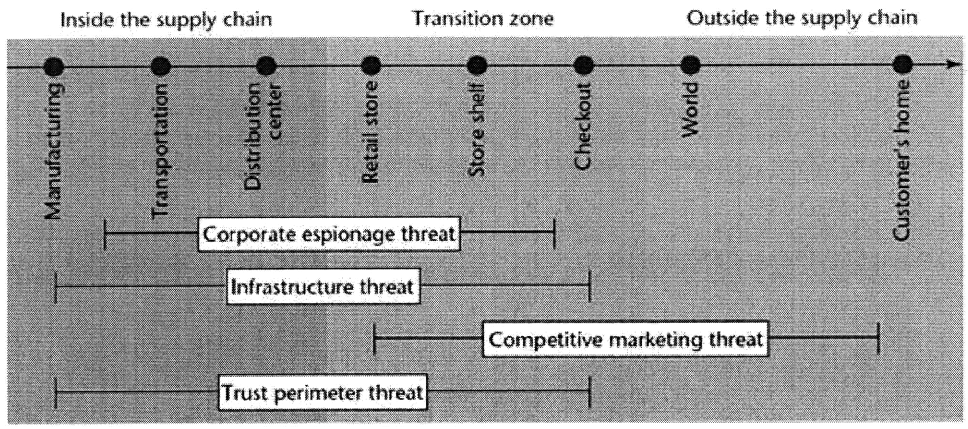

Figure 1-1: Corporate data security threats[45]

However, as Figure 1-1 shows, EPCs pose threats to corporate data security because the same tag can be read by multiple parties. Corporate data security threats mainly consist of corporate espionage, competitive marketing, infrastructure, and trust perimeter breaches[45]. Corporate espionage consists of competitors remotely gathering supply chain data - one of a corporations most closely guarded secrets - through tagged objects in the supply chain. Competitive marketing threats are the threats that competitors can gain unauthorized access to customer preferences and use the data in competitive marketing scenarios. A corporate infrastructure that is dependent on easily jammed radio frequency signals makes organizations susceptible to new kinds of denial-of-service attacks. Sharing large volumes of data electronically among organizations makes a corporation susceptible to a trust perimeter breach.

This evidence points to a need for a security framework for RFID networks. At present, no specific security or privacy framework has been specified as part of the the EPCglobal[1] standard[5]. The lack of agreement on a specific security framework in RFID networks is mainly due to the following three problems. The first problem is that RFID networks, supply chains in particular, are open world-wide systems. For example, in a supply chain, a single RFID tag must be read by manufacturing plants, storage warehouses, carrier cross docks, major distribution centers, ports,

[1]EPCglobal is leading the development of industry-driven standards for the Electronic Product Code (EPC) to support the use of Radio Frequency Identification (RFID) in todays fast-moving, information rich, trading networks.

intermodal terminals, suppliers, transport carriers, third-party logistics providers and retail stores. It is hard to make mandatory adoption of a particular identity system or a certificate authority for all such participants[31].

The second problem in standardizing RFID security is that there are trust problems in RFID networks. For example, a manufacturer sells a product to a distributor, and the distributor sells that product to a retailer. If a manufacturer wants to trace its products and requests particular EPC information from retailers, those retailers must decide whether they should give access permission to the manufacturer even though they do not have a direct business relationship.

The final difficulty in establishing a uniform RFID security framework is that the business rules for protecting access to EPC data can be complex. The power of an EPC code is that it provides access key to data about a product and its business context. For example, the product could be an F15 Engine GE504 destined for Pakistan, and the associated business information could be that the engine was shipped from Boeing on 12/5/2007 and was part of a purchase order (PO) that is stored on http://boeing.SAPsecretsite/secretPlace.aspx. Thus, the resource being accessed is potentially more than just an EPC code. Authentication and authorization decisions depend on all of the associated data: EPC, time, business event, PO number. Further, the user might have authorization to see some of the data but not all of it. An authorization system itself must be able to look up a business rule that is defined by EPC, shipper, date, PO context, and requester identity. Some policies are confidential in themselves due to business reasons, so only parts of them are exchangeable. At present there are no widely accepted standardized business rule languages which can express such complex rules.

In this thesis, an Interoperable Internet-Scale Security (IISS) framework is proposed for RFID networks as the solution to the security problems described previously. IISS performs authentication and authorization based on an aggregation of business rules, enterprise information, and RFID tag information. IISS provides a protocol for several authentication schemes and identity techniques. It also provides an engine for reasoning over business rules across domains. Moreover, IISS is able to resolve provenance information of RFID tags, which can identify the track of a particular piece of EPC data. The IISS framework and the ontologies used to model the information in IISS is described. How the IISS framework can be developed for access control in RFID-enabled supply chains is discussed.

1.1 Approach

In this thesis, several strategies are advocated and justified to solve the security issues associated with RFID networks.

An adequate security framework must avoid centralized identity providers in order to handle the wide variety and huge number of parties involved in RFID networks. In a decentralized network, the practical application of synchronizing any entity to another presents significant logistical challenges. An *identity selecting scheme* in RFID networks is advocated in this thesis to allow relying parties(the parties who own data resources) and querying parties(the parties who make queries) to reach agreement on the selection of an identity provider automatically. Different authentication schemes and identity techniques are supported, so parties using different techniques or platforms remain interoperable. A security protocol is provided, through which relying parties are able to specify security token formats and techniques that querying parties need to support. Querying parties can retrieve and provide security tokens according to the requirements from relying parties.

Provenance challenges is introduced to solve trust issues in RFID networks. An RFID network, by its nature, is a decentralized network. Entities in RFID networks communicate directly with each other without the support of a centralized server, authority, or database. In order to perform authentication and authorization in such an environment, some secrets must be shared by the parties who have common features so that one of these parties can perform authentication based on whether the querying parties have the same feature. Provenance information of EPC data is generated and passed through RFID networks to identify the track of particular EPC data for both traceability and authentication reasons. Because provenance information is the secret shared among all the parties in a particular supply chain, the provenance information can be used as a form of proof for authentication when two parties do not share any identity authorities.

RFID networks are open to parties in a wide variety of domains. It is important to ensure that business rules are interchangeable between agents in different domains. To achieve this goal, business rule vocabularies which are available in a machine understandable format is defined. Moreover, the set of vocabularies can be referred to and shared by agents in different domains. Among current web technologies, the Semantic Web consists of a distributed environment of shared and interoperable ontologies, which have emerged as common formalisms for knowledge representation. Therefore, I chose the Semantic Web format - Resource Description Framework (RDF) [21] - to represent the business rules and vocabularies in RFID networks.

It is important that any solution to the data exchange problem in RFID networks not make simplifying assumptions about authorization policies, but instead be flexible enough to accommodate the parties' desires to retain full control over authorization decisions. A business rule language for RFID networks is defined in this thesis. Resource managers in RFID networks need a business rule language that allows them to describe their complex business rules according to only their requirements. Enterprises can give permission to particular parties by doing inference on facts and some business rules. This business rule language described in this thesis for an RFID security framework allows resource managers to perform access control dynamically instead of searching static identity lists. This thesis also defines a policy structure which allows an enterprise to expose only the authorization requirements of input parameters instead of the entire business criterions of EPC-related authorization.

Based on these strategies, IISS, an interoperable internet-scale security framework for RFID networks is designed and implemented. IISS is a World Wide Web-based framework for an agent in RFID networks to perform authentication with another agent from a different domain by reasoning over queries and related policies described in RDF-s[40] and OWL[20].

1.2 Implementation

IISS's data model is implemented using the Resource Description Framework (RDF), a descriptive logic framework for encoding metadata on the World Wide Web. RDF is a World Wide Web Consortium (W3C) recommendation, and its status as a standard has encouraged the development of a variety of different internet-based metadata systems including the widely-used RDF Site Summary (RSS) and a news syndication format. The key distinguishing feature of RDF is its use of *uniform resource identifiers* (URIs) for identifying objects. These identifiers are globally unique and have well-defined semantics that persist when they are exchanged between systems. As IISS is designed to support interoperation in globally-distributed RFID networks, this feature makes RDF a natural choice for IISS's data model. To develop an approach better suited to sharing information in an open RFID environment, a description logic language for defining and using ontologies on the Web in machine-readable form is provided. In applying the IISS framework, the initial semantic web ontology is extended by defining classes for EPCs, queries, identities, provenance, and policies; and creating appropriate instances.

To decide whether or not to give permissions for a query about a particular EPC, relying parties need three kinds of information related to EPC-bearing objects. The first bit of information – enterprise level information – concerns the enterprises' identity. The second – business rule level information – concerns the rules or policies associated with the EPC-bearing object in a particular business context. The third kind of information – item level information – concerns EPC-bearing objects, such as their provenance information or EPC-related information. In order to performs authentication based on the integration of the information from all of the three levels, the IISS framework is composed of five modules to process the information: (1) an *EPCIS*, which stores EPC data generated by RFID readers; (2) a *Provenance Server*, which stores the provenance information of each RFID tag-bearing object; (3) a *Rule and Policy Engine*, which collects relevant policy information or proofs and generates permissions after reasoning over the collected information; (4) an *Identity Selector*;

(5) and *Identity Providers*, which issue identity tokens.

The modules are implemented in the object-oriented C# language, and using Windows Communication Foundation libraries. Cwm[23], a general purpose data processor written in Python, is used as the reasoning engine for parsing and merging ontologies and for making logic inferences over *Query* instances and *Policy* instances. A web portal is provided to let users generate policies for particular ECP events.

This thesis focus on a supply chain as the standard use case to demonstrate how IISS can be used in RFID applications. In this use case, when a manufacturer starts shipping an item, his IISS generates a provenance challenge to identify the item's trace. When the item passes through a supply chain, the EPC code's provenance challenge will be transmitted confidentially and stored in each agent's provenance server in the supply chain. Consider the following example use case. Manufacturer A initiates a query to retrieve a purchase order about item C with the EPC code 010000A8900016F000169D from a retailer D. However, D does not have a direct business relationship with A. Because C's purchase order is stored in a confidential web page http://boeing.SAPsecretsite/secretePlace.aspx, access permission to C's purchase order can only be given to C's manufacturer if he is one of the government's contractors. The manufacturer needs to present the appropriate X.509 certificate issued by the government. At the same time, the manufacturer has to prove that he knows the provenance challenge of C. The retailer D published the rule for this purchase order. The rule is described in RDF-s[21] and OWL[72], based on the policy ontology which is defined in IISS ontologies. The manufacturer A generates a proof file which contains information to validate its query. According to the result from the reasoning engine, D decides whether A is allowed to access the purchase order that contains C's EPC code after D's reasoning engine processes the proof file from A.

To test the efficiency of the IISS framework's protocol and explore its performance, the IISS framework is built and employed in a supply chain EPC network simulator. The supply chain EPC network simulator was developed by the AUTOID lab at MIT in order to estimate the quantity of message flow in the future EPC network infrastructure. The potential usability of the IISS framework's protocol in the real supply chains of industry is shown by experimenting with IISS using the simulator as a test bed.

1.3 Contribution

The thesis will present six major contributions and demonstrate their applications to supply chains.

First, a solution to security problems in the RFID networks on which multiple partners with different identity schemes can rely - the Interoperable Internet-Scale Security (IISS) Framework is developed. By employing this IISS protocol, parties in RFID networks do not need to give up their original security schema to adapt a unified technique or standard.

Second, IISS is shown to be the first security mechanism that performs authentication based on an aggregation of business context, enterprise information, and RFID tag information. Users take these three kinds of information related to RFID-bearing objects into consideration when they decide whether to give permission to access information to a query in RFID networks. It is very important to provide a mechanism to collect information over these three aspects.

Third, an ontology for describing entities and business rules in RFID networks is developed. The ontology models concepts such as provenance, EPC, query, and policy. Based on the ontology, business rule vocabularies and a business rule language that allow parties to express complex rules with business-specific context in RFID networks are defined. Moreover, business rules defined on a shared business language can be interchangeable and understandable by parties from different domains.

Fourth, it is shown that IISS provides a dynamic mechanism to regulate access control of EPC- related resources. This scheme is an improvement over role-based access control solutions. It gives an enterprise more flexibility when defining its resource access criteria. Moreover, it will save resource managers a significant amount of labor from maintaining the role database and identity list[2].

[2]The role database and identity list store users' identification information and the information

Fifth, it is shown how IISS introduces provenance challenges in order to solve the trust problem in RFID networks. Provenance information can be used as a proof to validate a party's query to distant members of the supply chain who may or may not know about existence of this particular querying party. The provenance challenges are very lightweight and, thus, do not add much traffic to RFID networks.

Sixth, an analysis of the infrastructure of a centralized registration model in supply chains is detailed. Based on testing data, it is proved that centralized registration is not a feasible solution in terms of the security transaction workload. Moreover, merely learning the identity of who has data about a specific EPC from the centralized registry can reveal confidential information. Therefore, discovery information faces similar authorization issues as event data itself. IISS provides a strategy that enables enterprises to trace particular RFID-bearing objects without using a centralized registration service. A broadcasting mechanism may be used by parties to retrieve an RFID-bearing object's historic trace if the IISS framework is employed in RFID networks and a list of potential agents in the supply chain exists.

1.4 Thesis organization

The thesis discusses the principle underlying IISS design, starting by illustrating the basic IISS framework and its deployment infrastructure. I then work towards defining ontologies which conceptualize data flow in the IISS framework and demonstrate the policy structures defined based on these ontologies. Finally, an application of the IISS framework to the supply chain is presented to demonstrate the power of the framework in solving security issues in RFID networks.

Chapter 2 frames the problems discussed in this thesis. At the beginning of this chapter, the layer in RFID networks where the security issues which this thesis solves exist is described. Then data exchange problems of RFID networks are reviewed.

Chapter 3 examines key related work in the literature and explains the limitations of current solutions to the security issues faced by RFID networks, highlighting the

about which group they belong to

differences between these previous approaches and IISS as presented in this thesis.

Chapter 4 presents the semantic network-based IISS framework, which embodies five characteristics of the security scheme in RFID networks. I describe how RDF suits these requirements and how it is employed in IISS. Some basic RDF vocabularies that are used throughout the system and this thesis is introduced. In addition, The provenance information, which is a unique feature of IISS, is described.

Chapter 5 presents the IISS framework's infrastructure and abstracts the idea of storage containers in IISS. The IISS's backend, which consists of a multi-agent environment with a shared RDF-based repository acting as common reference, is characterized. The mechanism for interoperating among the IISS framework's distributed files and modules on the web is also presented and justified.

Chapter 6 presents the ontologies developed for modeling the entities involved in authentication and authorization decision in RFID networks. These entities lay the foundation for representing the information exchanged in the IISS framework. Those entities are *Query, EPC, Identity, Policy*, and *Provenance Information*. How the ontologies are described in the form of RDF is explained. With this foundation, how the IISS framework manages to link and integrate data distributed in widely distributed RFID networks through the IISS ontologies is discussed.

Chapter 7 presents the policy structure of the IISS framework. How the policy files are deployed and published is demonstrated. After that, I discuss how the implementation of the IISS data model can support a basic forward chaining mechanism-which can be used for business rule inference-and how the business rule engine does reasoning over *Query* instances and related policies. In the end, the policy user interface developed for generating EPC-related policy rules in the RDF format via a policy writer is described. With the policy user interface, users not familiar with the RDF format are ensured to be able to use an English-like representation of business methods, entities, conditions and actions.

Chapter 8 demonstrates the application of the IISS framework to the problem of authorization and authentication on EPC-related queries in supply chains. I explore how to use IISS to handle queries from different domains and how to generate and publish business rules which are interoperable between multi-domain agents in supply chains. I analyze latency results obtained by simulating a registry model in supply chain EPC networks to argue that a centralized registry model should be avoided. Then how to avoid a centralized registry model in the data transmission layer by introducing IISS in supply chains is shown.

Chapter 9 introduces a supply chain EPC simulator as the testing platform for the IISS framework. How to integrate the IISS framework with the supply chain EPC simulator is demonstrated. Then the efficiency of the IISS framework in the domain of authorizing and authenticating EPC-related queries in supply chains are shown. This chapter is finished by discussing factors that affect the IISS framework's scalability and provide solutions to improve its performance.

Chapter 10 concludes the thesis and gives some summarizing remarks on the limitations of the IISS protocol and suggestions for future work.

Chapter 2

Problem Space

This chapter set the stage for a discussion of the fundamental principles underlying IISS that follow in the ensuing eight chapters. It is started by describing the layer in RFID networks where the security issues which this thesis solves exist. The data exchange problems of RFID networks are reviewed.

To date, the work of the EPCglobal business and technical communities has laid a solid foundation for exchange of EPC data, including the definition of the Electronic Product Code (EPC) standard, and the EPC Information Services (EPCIS) standard. Despite this foundation, data exchange in a multi-party supply chain setting is not fully addressed by the existing standards or architecture documents.

The EPCglobal Architecture Framework defines the EPC network as a community of trading partners engaged in the capturing, sharing, and discovery of EPC-related data using standard interfaces[5]. The EPC network enables companies to track goods using EPC tags, and to securely exchange information over the network. As shown by Figure 2-1, the EPCglobal Architecture is divided into three layers: the EPC physical object exchange layer, the EPC infrastructure layer, and the EPC data exchange layer. EPC physical object exchange defines the standards for tag data and tag data translation. The Tag Data Standard defines the specification for EPC tag data, including how the data is encoded on the tag and how must be encoded for applications in the information systems layers of the EPC network. The Tag Data Translation Standard defines a machine-readable version of the EPC tag data

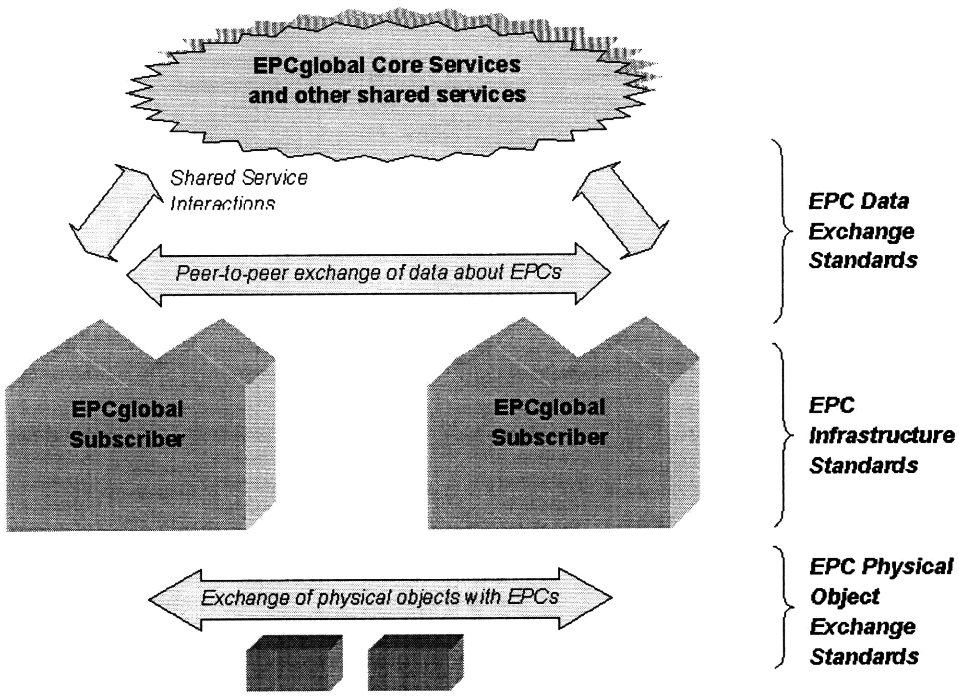

Figure 2-1: EPCglobal architecture framework[38]

specification.

On top of the EPC physical object exchange layer is the EPC Infrastructure layer which defines standardized protocols for capturing and subscribing EPC data. These standards include a low-level reader protocol, a higher-level reader protocol, reader management, and the tag protocol UHF Class1 Gen2.

The topmost layer in the EPC network is the EPC exchange layer which defines protocols for peer-to-peer EPC data exchange among EPCglobal core and other shared services. In this layer, standards for EPC information services, application level events, object name services, discovery services, and Epedigree[1] are defined [5].

RFID technology provides security challenges in all the three layers. This thesis focuses on the security problems in the highest layer-the EPC data exchange layer. The security issues in this layer are the most challenging due to the diversity of stakeholders and minimal standardization at the time of writing. The functional

[1]Epedigree documents are used by pharmaceutical supply chain participants to protect consumers from counterfeit drugs.

requirements of the EPC data exchange layer are the most open-ended portion of the EPC network[38]. The application space itself is large and growing rapidly. The use-cases that the EPC network serves are expanding with this burgeoning adoption. Within any use case, the functions that the EPC network is being asked to handle are also rapidly changing. For example, while the EPC network was always expected to serve large consumer packaged goods (CPG) suppliers and retailers, a number of smaller entities are beginning to use EPC tags to communicate in a more freeform way.

The EPC network must be able to adapt to changing requirements. Companies rely on the EPC data exchange layer to provide services that enable trading partners to interoperate and to take advantage of current and future EPC capabilities. At the same time, the EPCglobal network is a highly distributed information-sharing model. While EPCglobal networks are trying to bring a sharing mechanism and a form of openness to various enterprises, they also expose the associated enterprises to growing dangers and security issues. The new problems are then left to security researchers and specialists in this information transmission layer to handle. For example, if inadequate security is in place, an agent could query a competitor's products sales information from distribution centers and monitor the centers' replenishment dynamics. Because tagged objects are uniquely numbered, it's easy for competitors to unobtrusively gather large volumes of such data. In this chapter, I discuss the three main security issues faced in the data exchange layer of EPC networks. In later chapters I will show how each of these problems has been address in the IISS framework proposed in this thesis.

2.1 Problem one-Which identification scheme should be chosen?

In order to decide whether to give querying party access permission to the requested EPC data, an authentication agent in RFID network inevitably needs to authenticate the querying party. During the authentication phase, the querying party has to provide some identity token that is issued by a third party which can prove the authenticity of the querying party's identity. This third party is what I call an *identity provider*. The kind of identity token issued by an identity provider is decided by what authentication technique the identity provider supports. At the same time, in order to authenticate the identity token provided by the querying party, the authentication agent has to support the same authentication technique as the issuing identity provider.

There are a number of these authentication techniques available to use including as the Public Key Infrastructure (PKI), Kerberos, and SAML. With this variety of authentication schemes, the first issue to address is which kind of identity scheme the authentication agents in RFID networks should adopt? It is vitally important that a relying party be able to recognize the querying party's identity. The relying party is only willing to share data with authorized querying parties by virtue of the relying party's own information sharing policy. Unfortunately, the emergence of a single, simple digital identity solution as a universal panacea is not realistic.

There are several reasons that developing a single identification scheme does not work. First, the RFID network, and the supply chain in particular, is a globally open system. For example, in a supply chain, an RFID tag is read by manufacturing plants, storage warehouses, carrier cross docks, major distribution centers, ports, intermodal terminals, suppliers, transport carriers, third-party logistics providers, and retail stores. In addition, the tags themselves may travel widely, even across country borders. The authentication agents in this case might deal with several identity providers and security schemes. This is due to the fragmentation of the global certificate authority business, with national or regional providers dominating their home markets. Many uses of digital certificates, such as legally binding digital signatures, are linked to local law, regulation, and accreditation schemes. Agents in RFID networks will almost certainly choose their local certificate authorities as their identity provider. Second, most agencies want to keep their own digital authorities. For example, governments have their own digital authorities to distinguish them from other kinds of organizations. Third, enterprises need to maintain relationships with their customers, which may mean sticking with a legacy authentication scheme. It is very difficult for these enterprises to give up their original security scheme in order to adopt new authentication technologies. Fourth, the practical application of synchronizing any entity to another entity presents significant logistical challenges.

2.2 Problem two- Should access permission be - given to unknown entities?

Trust problems are rife in RFID networks. For example, a manufacturer sells products to a distributor, and the distributor sells products to a retailer. When the manufacturer wants to trace his products by querying for particular EPC information from the retailer, the retailer needs to decide whether or not to grant access permission to the manufacturer even though there is no direct business relationship between the two companies.

In the two-party example, it is well understood to build the trust relationship through pre-arrangement. That is, the manufacturer already knows that the retailer was the place to go to for the necessary data because the manufacturer had delivered the products directly to the retailer. The answer to the discovery question thus follows directly from the business relationship between the manufacturer and retailer. However, in most cases in the supply chain, products go through multiple parties in their journey from origin to destination. The data exchange question becomes more complex in these multi-party scenarios, and business relationships continue to play an important role in understanding how data exchange works.

In the scenario shown by Figure 2-2, four products with SKUs 1 to 4, respectively, are shipped out from two different manufacturers. They take different paths through two tiers of distributors before the shipments arrive to retailers. Each retailer has products from both the manufacturers who are competitors with each other. Just like the links in a metal chain, the members of those supply chains may only have

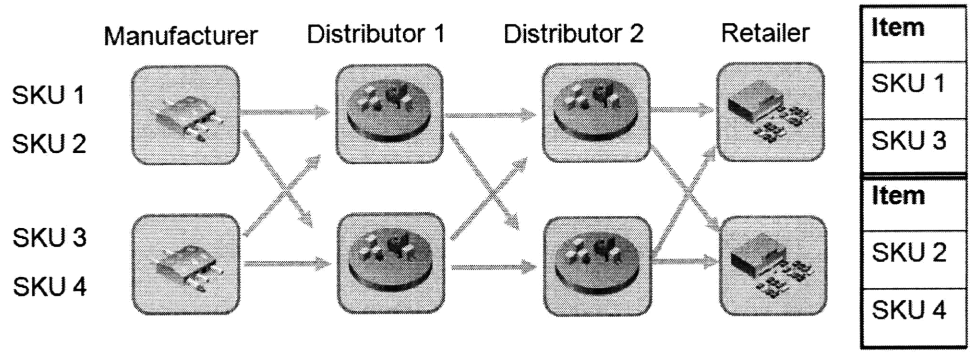

Figure 2-2: Business relationships in a supply chain

business relationships with their immediate neighbors. They may or may not know about more distant members of the chain and even if they are aware of their existence they may not have a business relationship with them. Queries from distant members have to be performed in order to execute some business activities. When those queries happen, trust issues emerge. For example, suppose retailer R1 queries manufacturer M2 about information associated with SKU 3. How does M2 know if SKU 3 was actually shipped to R1? Moreover, how does M2 know the query is not from one of its competitors posing as R1? M2 can try to consult shipping records to see if SKU 3 was actually shipped to R1, but this is only straightforward if manufacturers ship directly to retailers. If there are distributors or other intermediaries between the manufacturer and the retailer, as in this scenario, identifying trusted entities is much more challenging. In this scenario, the information about which retailers received particular EPCs might not be directly available to the manufacturer. In a promotion scenario, manufacturer M1 needs to query retailer R1 about the arrival time of promotional goods with SKU 1 to determine whether the promotional goods reached R1 on time. How does R1 know M1 is the manufacturer of the good with SKU 1? (Again, the manufacturer may not have a business relationship with the retailer). In a shipment confirmation scenario, distributor D1 queries retailer R1 with the following question "Were shipped goods received?" in order to prove delivery occurred. For the same reasons as previously discussed, R1 may not be able to validate the query from distributor D1.

As one can see from the example above, the answer to the data exchange question may include parties with whom the relying party does not have a direct business relationship. In those situations, merely learning who has data may violate confidentiality agreements that other parties have with each other. This means that the data exchange question becomes entangled with authorization, and therefore with authentication and trust.

2.3 Problem three- What information is an entity entitled to access?

Business rules for protecting access to EPC data can be complex. When a querying party actually issues its query to a relying party, the relying party has to authenticate the identity of the querying party before responding to the request. If authentication succeeds, the relying party can then decide how to respond to the query. This requires the relying party to make an authorization decision. The authorization decision is based on the relying party's own data sharing policy. The identity of the querying party, as confirmed through authentication, is one input into this decision, as are other bits of business information, such as the current state of business relationships, and the records of past business transactions.

Business rules can be quite complex, mirroring the complexity of real-world business relationships between companies. It is thus important that any solution to the data exchange problem not make simplifying assumptions about authorization policies. But the question still remains of how to make an authorization scheme flexible enough to accommodate the various parties' desires to retain full control over authorization decisions.

What sorts of rules might a relying party use to decide whether a querying party is entitled to data about a particular EPC? In reality, the rules may vary from simple to complex. In a simple case, if a retailer sells products from different manufacturers, it is likely the retailer will only share data about a particular product with the manufacturer who produces it and not with the manufacturers of other products. In the simple case, the company prefix portion of the EPC code indicates the identity of the manufacturer, which makes it simple to implement this rule. The retailers can just use the company prefix of the EPC in question to validate the incoming query. In complex cases, authorization rules may be subject to contractual arrangements, which may be tailored to individual partners, subject to time limits, etc. For example, a manufacturer may sell to a distributor, who sells to two different retailers. The manufacturer's right to share data with retailers may be limited by the agreements with the distributor. In this case, the manufacturer may be able to exchange data with one retailer but not the other, depending on the agreements between the distributor and those two retailers. That information may not be readily available to the manufacturer, and may vary over time as agreements are changed.

In RFID networks, authorization policies are not limited to controlling queries about EPC data. The authorization policies also control access to EPC related resources, such as purchase orders or shipment records. The power of an EPC code is that it acts as a key for accessing data about a particular product and its business context. For example, the product could be an F15 Engine GE504 destined for Pakistan, and the associated business information could be that the engine was shipped from Boeing on 12/5/2007 and was part of a Purchase Order (PO) that is stored at http://boeing.SAPsecretsite/secretPlace.aspx. The resources being accessed are potentially more than just an EPC Code. Authentication and authorization decisions might depend on any of the following data: EPC, time of transaction, business event, or PO number. Further, the user might have authorization to see some of the data but not all of it. An authorization system itself must be able to look up a business rule that is defined by EPC, shipper, date, PO context, and requester identity. Some policies are confidential in themselves due to business reasons, so only parts of them are exchangeable. For example, some authorization policies can leak secrets of an enterprise such as its business partners or confidential contracts. Therefore, enterprises need the flexibility to publish only the subset of their policies they are willing to publish. At present, there are no standardized business rule languages which can express such complex rules.

Chapter 3

Related Work

In this chapter, some related work about RFID networks and associated security issues will be reviewed. this chapter is started by introducing basic knowledge about RFID, EPC codes, and RFID networks. After that, a general review of current research on security issues in the RFID world is provided. The problem this thesis targets is the data exchange layer in RFID networks, which shares many common features with how data exchange occurs over the World Wide Web. As a result, a comprehensive review of the current authentication and authorization solutions for data exchange through the internet is also provided. This is followed by some proposed solutions by researchers in the EPCglobal community for the security issues specific to RFID networks. It is shown that none of the existing solutions that are discussed in this chapter can address all the issues previously described in chapter two. However, some of the solutions developed the foundation and provided inspiration to the formation of the Interoperable Internet-Scale Security (IISS) framework.

3.1 RFID, EPC, and RFID networks

3.1.1 Radio Frequency Identification (RFID)

Radio frequency identification, or RFID, is a generic term for technologies that use radio waves to automatically identify people or objects. RFID is a proven technology

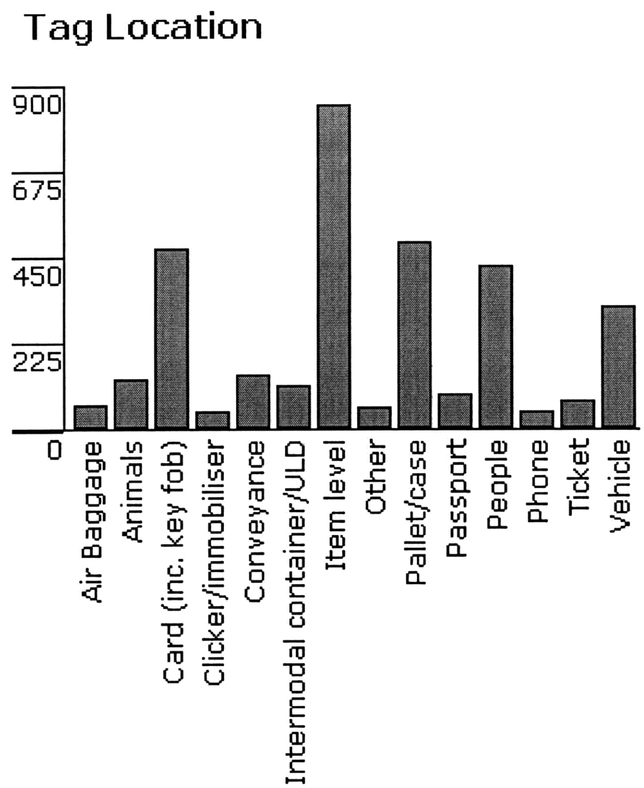

Figure 3-1: RFID distribution[1]

that's been around since at least the 1970s. Up to now, it's been too expensive and too limited to be practical for many commercial applications. An RFID tag stores a serial number that identifies a person or an object, and perhaps other information. An RFID tag has a microchip that is attached to an antenna; the chip and the antenna together are called an RFID transponder or an RFID tag. The antenna enables the chip to transmit identification information to a reader. When an RFID tag reflects radio waves from a reader, the reader converts the radio waves into digital information that can then be passed on to computers that can decode and make use of it.

RFID technology has a broad range of applications. It has been used in inventory management, animal tracking, currency tracking, consumer goods tracking, e-passports, drug counterfeiting, and automatic toll collection, among others. Figure 3-1 shows how RFID tags are distributed in various items based on 3524 RFID case studies in the IDTechEx Knowledgebase[1].

Both RFID and bar codes are used widely in identifying items. The two are, however, different technologies and have different applications, which sometimes overlap. Compared with barcodes, RFID identification has five advantages. First, an RFID reader does not require a direct line of sight to the RFID tags being identified. By contrast, bar codes must be oriented towards readers. Second, an RFID tag can be read at a much greater distance than bar codes. Third, an RFID reader can read an RFID tag much faster. Fourth, RFID tags can be implanted within a product itself, hidden from view. In contrast, the printed bar codes must be exposed on the outside of the products in order to be read. Fifth, an RFID reader has the potential to alter the data stored on an RFID tag, while bar codes have no read/write capability.

The memory of an RFID tag depends on its vendor and its application, but typically a tag carries no more than 2KB of data-enough to store some basic information about the item it is on. Companies are now looking at using a simple "license plate"tag that contains only a 96-bit serial number. According to the function of tags' microchips, RFID tags can be classified as read-write, read-only, and "write once, read many"(WORM). With read-write chips, you can add information to the tag or write over existing information when the tag is within range of a reader. Typically, though, a read-write tag has a serial number that cannot be overwritten. Additional blocks of data can be used to store additional information about the item the tag is attached to. Read-only microchips have information stored on them during the manufacturing process. The information on such chips can never be changed. WORM tags can have a serial number written to them once, and that information cannot be overwritten later.

RFID tags can be classified based on the power source of the tags as active, semi-passive, and passive. An active RFID tag has a transmitter and its own power source. The power source is used to run the microchip's circuitry and to broadcast a signal to a reader. Passive tags draw power from a reader that sends out electromagnetic waves to induce a current in the tag's antenna. Semi-passive tags use batteries to run the chips' circuitries, but communicate by drawing power from readers. Active and semi-passive tags have 100 feet or more read range, while passive tags' read range is less than 20 feet. In general, active tags are useful for tracking high-value goods that need to be scanned over long ranges, such as railway cars on a track. Low-budget-users are focusing on far less expensive passive UHF tags, which cost less than 40 cents today in volumes of 1 million tags or more. The read range of passive tags depends on many factors: the frequency of operation, the power of the reader, interference from other RF devices, etc. Low-frequency tags are usually read from a distance of foot (0.33 meter) or less from the reader. High-frequency tags are read from about three feet (1 meter), and ultra-high frequenc (UHF) tags are read from 10 to 20 feet.

Thousands of companies around the world use RFID today to improve efficiency. For instance, a company could use RFID to track items on a production line to make the production line more effective, or use the information to increase the throughput of containers at its distribution center. However, these investments and their associated improvements are usually seen in closed-loop systems. A company is only able to track goods that never leave its own control. That's because some existing RFID systems use proprietary technology, which means that if a company puts an RFID tag on a product, it can't be read by another company unless they both use the same RFID system as supplied by a common vendor.

3.1.2 Electronic Product Code (EPC)

In order to enable the identification of all physical objects and share this identification globally, the AUTOID Center at MIT[77] conceptualized the Electronic Product Code (EPC). In a supply chain context, this means tracking every manufactured unit on a global scale using unique serialized identifiers. An EPC code is a unique identification scheme consisting of a header and three sets of data partitions[30]. The first partition identifies the manufacturer or the identity responsible for maintaining the object class and serial number. The object class identifies the item, eg, the stock keeping unit (SKU) or consumer unit. The second identifies the Object class by storing an item number, stock keeping unit, or, alternatively, Lot Number. The serial number, which is unique to the item within the Object class, is stored in the third partition. By separating the data into partitions, readers can search for items quickly by reading EPCs and identifying RFID tags with particular manufacture codes, product codes,

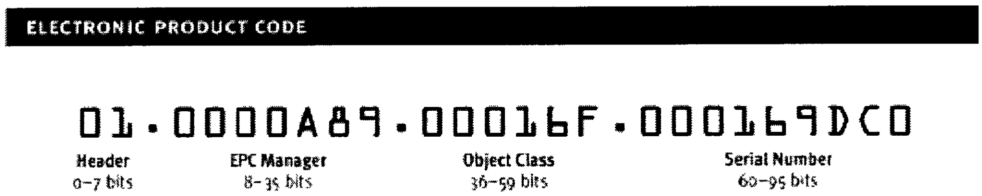

Figure 3-2: An example of a 96-bit EPC code

and serial numbers. This data structure is currently supported in several classes of EPC and is available in more than one length scheme. For example the 64- and 96-bit versions of EPC are fairly common. Other versions include data capacities of differing sizes and types. Wal-Mart has focused on the use of a minimum of 96-bit EPC codes. This size may be increased with the pending Class 1 Generation 2 specification. Figure 3-2 shows an example of a 96-bit EPC code.

3.1.3 RFID Networks

With the goal of creating an internet of physical objects, EPCglobal, a non-profit organization which leads the development standards for EPC to support the use of RFID, defines the EPCglobal network as a community of trading partners engaged in the capture, sharing, and discovery of EPC-related data using standard interfaces. The EPC Network enables companies to track goods using EPC tags, and to securely exchange information over the network as shown below in Figure 3-3.

The EPCglobal network, which is also known as the RFID network, describes components and interfaces for EPC-related information interchange between servers. These servers contain information related to items identified by EPC numbers. In the RFID network, EPC-related information is stored via EPC information services (EPCIS)[13]. By installing EPCIS, disparate applications are able to leverage EPC data both within and across participating enterprises. After a reader reads the EPCs from RFID tags which are associated with the items - such as products, pallets, and containers - EPC-related information is generated and stored via EPCIS through the enterprises' internal EPC middleware. EPC-related information is stored as events

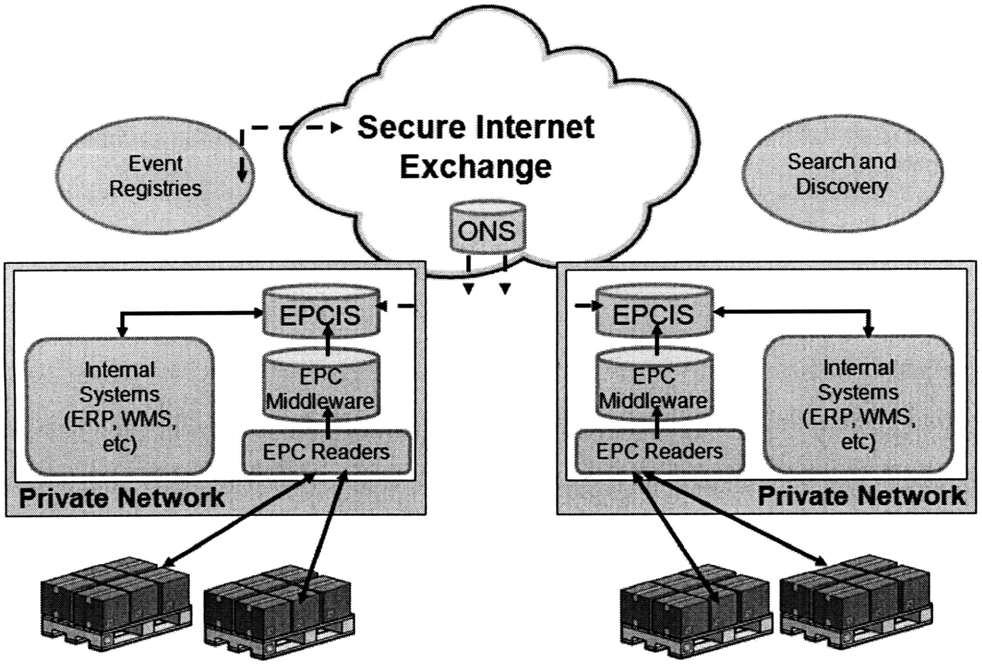

Figure 3-3: RFID networks

which have the descriptive components *what* (product), *when* (time), *where* (location), and *why* (business step and disposition), which together describe the specific occurrences in the supply chain. An example event would be as follows: the EPC urn:epc:id:sgtin:0614141.107340.1 (product) was added (business step) in unsalable condition (disposition) at urn:epcglobal:fmcg:loc:0614141073467.1 (location) at 2006-06-25T00:01:00Z EDT (time). Figure 3-4 shows how this event is stored as a document complying with a standard XML schema in EPCIS for consumption by other business applications.

Another important component of the RFID network is the Object Naming Service (ONS)[6], a counterpart of the Internet Domain Name Service (DNS)[67]. Instead of mapping IP addresses to domain names, ONS takes an EPC as input and produces as output an address pointing to one or more services (in the form of a Uniform Resource Locator (URL)) such as the EPCIS repository designated by the EPC's EPC Manager. It enables users to discover information about products and their related services just from the representative EPC. ONS accomplishes this by using DNS and encoding the IDs in a manner inspired by the Hesiod[44] developed in the

```xml
<ObjectEvent>
  <eventTime>2006-06-25T00:01:00Z</eventTime>
  <epcList>
        <epc>urn:epc:id:sgtin:0614141.107340.1</epc>
  </epcList>
  <action>ADD</action>
  <bizStep>urn:epcglobal:hls:bizstep:commissioning</bizStep>
  <disposition>urn:epcglobal:hls:disp:active</disposition>
  <readPoint>
        <id>urn:epcglobal:fmcg:loc:0614141073467.RP-1</id>
  </readPoint>
  <bizLocation>
        <id>urn:epcglobal:fmcg:loc:0614141073467.1</id>
  </bizLocation>
  <hls:temperature>20</hls:temperature>
  <hls:batchNumber>1</hls:batchNumber>
</ObjectEvent>
```

Figure 3-4: A case commission EPC event stored in EPCIS

1980s at MIT.

EPCglobal is also conceiving an EPC *discovery service*: effectively, a "search engine" for EPC-related data. A discovery service returns locations that contain some data related to a queried EPC. Unlike ONS, a discovery service may contain pointers to entities other than the entity that originally assigned the EPC. It is expected that there will be multiple competitive discovery services, much like there are multiple search engines for finding information on the Web, and that some of them will have limited scope to facilitate better results at a local level. The standards for a discovery service are still under development.

3.2 Security solutions in the RFID world

When considering privacy and security issues, most researchers' main concerns are about information leaking or being counterfeited during data transmission between tags and readers. There is a significant amount of research addressing privacy issues in the physical tag layer. In this section, I review the currently proposed solutions to these issues for the RFID world.

Based on their computing power, RFID can be broken down into two classifications: basic RFID tags, and contactless smart cards[56]. Basic RFID tags cannot execute standard cryptographic operations. However, smart cards are able to securely manage, store, and provide access to data on the card; perform complex functions; and interact intelligently via RF with a contactless reader. Applications using contactless smart cards support security features that ensure the integrity, confidentiality and privacy of information stored and transmitted, such as mutual authentication; strong information security; strong contactless device security; authenticated and authorized information access; support for biometric authentication; and support for information privacy. These additional safety features come at a cost, though: smart cards are more expensive than basic RFID tags. Based on factors of cost and the sensitivity of information, people choose different tags for different applications. Figure 3-5 shows that basic RFID tags are mainly used in supply chains, and track-and-tracing systems. Smart cards are applied to applications with more sensitive information, such as IDs and payment cards.

3.2.1 Smart Card Security

A smart card and a *card accepting device* (CAD) communicate by transmitting small data packets called *application protocol data units* (APDUs). Several characteristics of this interaction make it difficult for third parties to attack the system. First, the transmission is done at a low bit rate (9600 bits per second) using a serial bi-directional transmission line (ISO standard 7816/3). Second, smart cards use a half-duplex mode for sending information (i.e., data only travels in one direction at a

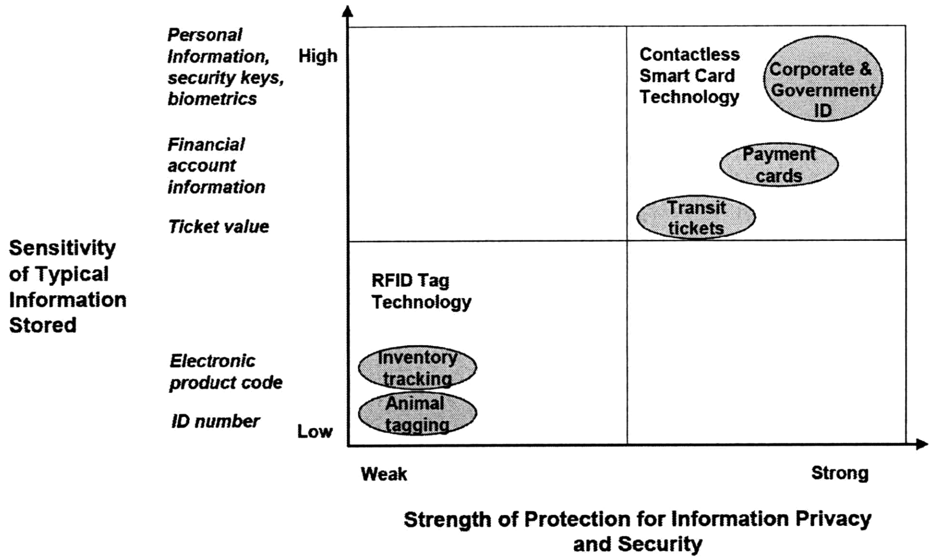

Figure 3-5: RFID application vs Contactless smart cards application[16]

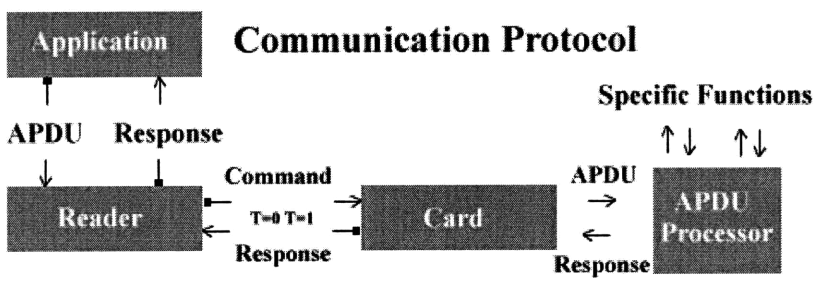

Figure 3-6: Communication protocol between a smart card and a CAD[81]

time). Third, the communication between a smart card and a CAD follows a mutual active authentication protocol which identifies each device to the other. As shown in Figure 3-6, a card first generates a random number and sends it to the CAD. The CAD encrypts the number with a shared encryption key, and then returns it to the card. The card then compares the returned result with its own encryption[53].

Once communication is established, messages between the smart card and CAD are verified with a message authentication code. This is a number that is calculated based on the data itself, an encryption key, and a random number. In this way, if data has been changed for some reason, such as due to a transmission error, the message will be retransmitted. Alternatively, if a chip has sufficient memory and processing power, the data can be verified through use of a digital signature. The most common encryption methods used in smart cards are symmetric data encryption standard(DES), triple DES(3DES), and public key RSA(Rivest-Shamir-Adleman's algorithm), allowing up to 56, 168, and 1024 bit long keys, respectively[81].

Even though smart cards can perform encryption, they are still prone to cloning[27], reverse-engineering, side channels [32], and relay attacks[60, 49]. The central issue of these privacy problems for the symmetric-key-enabled RFID tags lies in the challenge of key management. Because secure authentication of an RFID tag T depends on the symmetric key K shared between tags and readers, a reader must determine which key K it shares with a tag in order to perform any mutually intelligible cryptographic operation. Weis, Sarma, Rivest, and Engels[37] developed the first general approach for key search in RFID-tag identification. They proposed a basic scheme in which a tag emits the value $E = (h(k, R), R)$, where R is a random nonce generated by the tag. To identify a tag, a reader computes $h(k\prime, R)$ for all keys in K until it finds k. Molnar and Wagner (MW)[69] proposed a tree-based approach in which a tag contains multiple keys. These keys are arranged in a hierarchical structure defined by a tree. Molnar, Soppera, and Wagner[68] improve the MW scheme by associating the subtree with individual tags.

The second approach researchers explored is a synchronization-based approach which allows a reader to maintain synchronized state with tags. The basic idea is that tag T_i maintains a counter $P = c_i$ that is incremented with each reader query. When interrogated, the tag outputs $E = f(k_i[c_i])$. A valid reader who knows the approximate current value of the counter of tags in the system can store a searchable table of tag output values. Suppose that for every tag T_i the reader maintains a counter value $c\prime_i$ that does not lag behind c_i by more than d timesteps. Then if the reader maintains the output values $f(k_i[c\prime_i]), f(k_i[c\prime_{i+1}]), ..., f(k_i[c\prime_{i+d}])$ in a table of size $d * n$, it can at any time look up the output E of tag T_i. Ohkubo, Suzuki, and Kinoshita[71], Henrici and Mller[51], Juels[55] and Dimitriou[35] propose various solutions based on this principle. In order to prevent an adversary from recovering secret key k from a cipher text $C = e(k[M])$ on a predetermined message M, Dimitriou in his later scheme[35] proposes that the tag and reader refresh k_i in every time step by hashing it. Thus, it would be infeasible to compute previous keys and outputs from the current key in the short amount of time during each time step.

3.2.2 Basic RFID tag security

Basic RFID tags, such as EPC Gen 2 RFID tags, were designed for supply chain applications and had to be low cost, able to be read from a long distance, and able to support dense tag environments. The EPC Gen 2 Class 1 specification has two basic security features:

- A static 32-bit password that would accompany the kill command. If it received the kill command, the tag would self-destruct.
- An optional static 32-bit password for access-controlled memory in EPC tags. An EPC reader would need to furnish this password to read and write to certain memory locations.

Because the security features of basic RFID are so limited, adversaries can use clandestine scanning for unauthorized tracking and inventorying. Major threats enabled by this impoverished security set include the following:

- EPC tags release their identifiers and product information to any compatible reader, with no ability to authorize that the reader is allowed to access the information prior to releasing the data.
- The static kill and access control passwords may be read.
- EPC tags may be cloned.
- Eavesdropping and/or hotlisting[1] may occur on communication between the RFID tag and the reader.

[1] In the case of hotlisting, the antagonist's intention is to match individuals with known RFID identifiers.

Researchers have provided various solutions to address security issues for basic RFID tags. Some researchers proposed a means to change or re-label a tag's identifiers after certain transactions are completed, such as checking out, to prevent tag tracking[78, 52, 47]. Juels[55] proposed a minimalist cryptography in which tags rotate through a small collection of pseudonyms. Only an authorized reader can correlate different appearances of the same tag. He also proposed to employ a public-key algorithm with a single key pair. Instead of emitting the identifier, the RFID tags emit the cipher text under the public key. In this way, only agents with the private key can read the tags[54]. Golle et al.[46] extended Juels's solution by utilizing the EL Galmal cryptosystem, which allows for the re-encryption of a cipher text without knowledge of the corresponding public key.

Some researchers have proposed some privacy protection devices that consumers can carry to monitor ambient scanning[43, 76, 58]. Fishkin, Roy and Jiang[41] designed a tag which can release different levels of information based on the distance between the tag and the corresponding reader. Another privacy protection method put forward is to use blocker tags to obstruct the information gathered by RFID readers[57]. In this scheme, a modifiable bit called a privacy bit is incorporated into an RFID tag. The privacy bit marks whether the tag can be publicly scanned. A blocker tag is a special tag which prevents unwanted scanning of tags with privacy bit set to be "1". The blocker tags take advantage of the singulation protocol, used to prevent collision during communication between tags and readers. A blocker tag can impede readers by simulating many ordinary RFID tags simultaneously which cause the reader to stall before reading the tags that are marked as private.

3.2.3 Data exchange security in RFID networks

It is clear from the work presented so far that many researchers pay attention to securing transmission between RFID tags and readers. However, in order to enable participants in the EPCglobal network to gain a shared view of EPC-bearing objects within a relevant business context, a security scheme needs to be implemented at the level of information transmission between agents distributed throughout the internet.

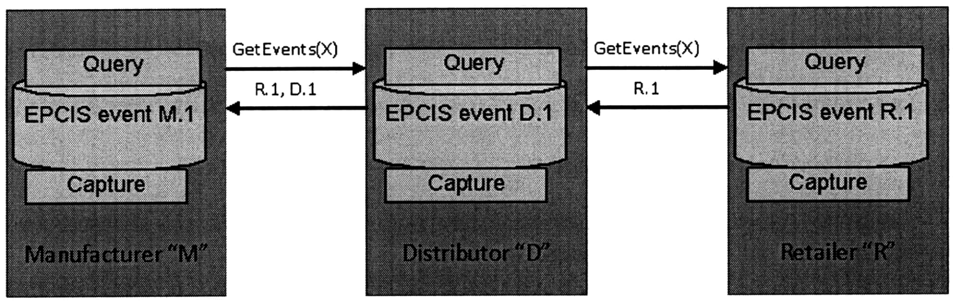

Figure 3-7: Query data via following the chain[38]

To date, I have only found two projects in the research literature that discuss authentication and authorization mechanism in the information layer of RFID networks. Dong Seong Kim[61] proposed a framework in an RFID multi-domain system which mainly solves the trust delegation problem in RFID networks. In this framework, a user can access RFID tags in one domain through another domain that has a trust relationship with the first domain and is able to authenticate the user in the first domain. This is a simple use case in multi-domain data exchange in RFID networks. Kim's framework is limited to fixed network architectures where the two involved parties had previously established a trust relationship. However, as shown in chapter 2, multi-domain data exchange in general RFID networks is much more complicated. The two parties may not even know each other when queries are executed in an RFID application over a supply chain.

Another important piece of work in this area is the "Framework for multi-party data exchange" as described in an EPCglobal white paper[38].The paper presents two solutions: the first is querying data by following the chain, as shown in Figure 3-7, and the second is an open registry approach, as shown in Figure 3-8. To illustrate these approaches, suppose the following scenario is true: manufacturer A sends an item with EPC X to distributor B, and distributor B ships the item to distributor C. In chain-following approach, if manufacturer A wants to discover and retrieve information about an item, manufacture A needs to query B about the EPC X related information. Distributor B will retrieve the EPC X related data from retailer C for

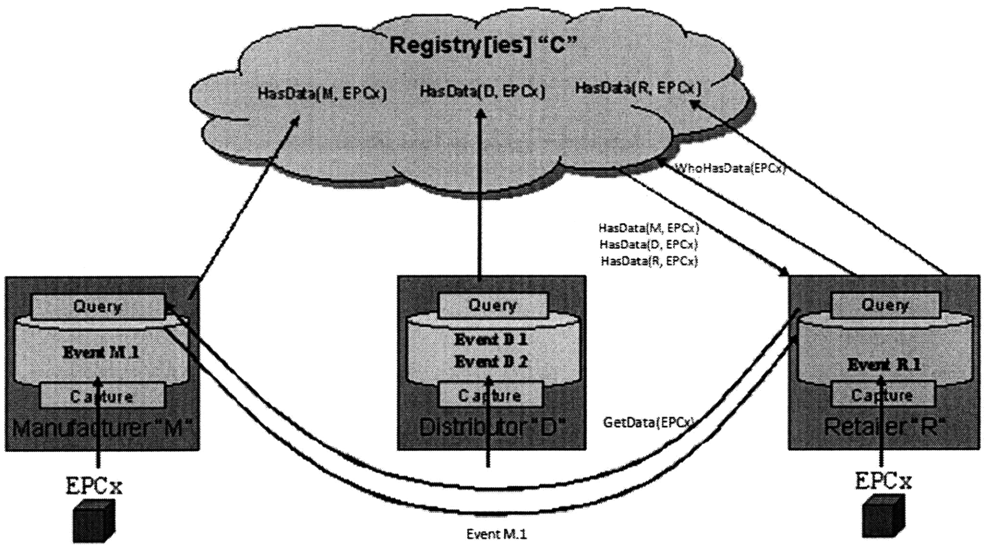

Figure 3-8: Open registry approach[38]

A and forward that information to manufacturer A. The advantage of this solution is that it reduces the complexity of data exchange among three or more parties to the complexity of several rounds of data exchange between two parties. In this way, all the existing authorization and authentication techniques can be applied to each round of communication. However, this scheme exposes itself as naive: why should distributor B carry the cost both in technique and bandwidth from querying information for manufacturer A? How can A trust that B that will reply with full disclosure without further tampering with the data? This approach also shows itself to be unreliable. The entire query process will fail if any party in the chain fails to forward the query or the resulting data. These problems make querying data by following the chain impractical in the real RFID world.

The open registry approach in [38] proposed a centralized registry server which receives reports from all the parties in an RFID network regarding their ownership of items with the EPC X and the associated access permissions for those items. With the centralized registry, a party can quickly discover where a particular EPC-associated item is and further correspond with the party who currently has the desired item. As mentioned in [38], this approach has some advantages because parties can keep their own data and do not need to forward data for other parties.

However, the open registry scheme also has obvious problems. This scheme only provides a solution to find the location of an EPC-related item. But queries in RFID networks should be more powerful than simple queries about who owns a particular item. A query could be any EPC code-related information such as purchase orders, shipment records, or customer information. The registry can lead a querying party to the entity which has the EPC data, but it does not provide a solution for authentication and authorization between the querying party and the relying party.

Moreover, the registry has its own security problems. First, not all the companies want to expose their EPC ownership information because the information itself may be a business secret. For example, with full compliance to this scheme, a manufacturer can check a rivals' sale information in a certain area by querying for the number of EPCs with the rivals' identifiers which the local retailers own. Government can also take advantage of the registry server to monitor an enterprise's business information. Second, the registry solution suggests that the registry play the role of information broker, collecting information from replying parties and forwarding it to querying parties. But the registry cannot make authorization decisions for a relying party because it has no role in the business relationship between the two involved parties. Third, making all parties in an RFID network agree to trust an independent agent acting as the registry server is a huge challenge. Even setting up a centralized registry server for global distributed RFID networks is difficult due to the bandwidth and computational limitations of current server technology. This problem will be more apparent in a later section of this thesis, as I will show that the estimated workload of a registry server in a pharmaceutical supply chain renders the centralized registry scheme infeasible given today's technology.

3.3 Solutions for data exchange on the web

Most data exchange in RFID networks occurs over the internet and the authentication and authorization problems of RFID networks have some common features with the trust problems in decentralized network communication. In this section, I review the current solutions to trust issues over the internet in order to explore the fundamental techniques and inspirations for solving data exchange issues in RFID networks. For readability, I have broken this work into two subsection: identification techniques and authorization policy languages. During this discussion, I point out which of the solutions are adaptable to the domain of data exchange in RFID networks.

3.3.1 Identity technologies

Over the internet, entities need to show their digital identities or credentials to service providers in order to access resources. The most common identification scheme is the username, which people register on a resource website or through some other means such as email. To verify an entity's identification, the authenticating party can require the use of passwords, pin numbers, or codes generated by a mathematical algorithm. However, these methods are static in nature and, as a result, are susceptible to passwords being stolen, guessed, decoded, or reverse-engineered. Additionally, a central server and database are required to verify usernames against their associated authenticating codes.

There are a number of other digital credentials which a service provider can use to verify an entity's identification, such as X.509 certificates[79], Kerberos tickets[70], IP addresses, and SAML tokens[66].

Figure 3-9 shows a decoded X.509 certificate for www.freesort.org issued by Thawte. In this example, the X.509 certificate is shown to include a large amount of data like the encryption algorithm name, issuer, and a pair of public keys. X.509 certificates are used in the socket security layer (SSL) protocol[83] and the Public Key Infrastructure (PKI)[15]. When a server receives a client's digital certificate, the server first verifies that the certificate has not expired. Then the server validates the credential of the client by verifying the signature of the issuing Certificate Authority (CA). To do this, the server must have the CA's certificate, and the CA that issued the client's X.509 certificate must be trusted by both the client and service. The server then uses the RSA public key from the CA certificate to decode the signature on the client's certificate to obtain an MD5 hash[36], which must match an actual MD5 hash

```
Certificate:
   Data:
       Version: 1 (0x0)
       Serial Number: 7829 (0x1e95)
       Signature Algorithm: md5WithRSAEncryption
       Issuer: C=ZA, ST=Western Cape, L=Cape Town, O=Thawte Consulting cc,
               OU=Certification Services Division,
               CN=Thawte Server CA/emailAddress=server-certs@thawte.com
       Validity
           Not Before: Jul  9 16:04:02 1998 GMT
           Not After : Jul  9 16:04:02 1999 GMT
       Subject: C=US, ST=Maryland, L=Pasadena, O=Brent Baccala,
                OU=FreeSoft, CN=www.freesoft.org/emailAddress=baccala@freesoft.org
       Subject Public Key Info:
           Public Key Algorithm: rsaEncryption
           RSA Public Key: (1024 bit)
               Modulus (1024 bit):
                   00:b4:31:98:0a:c4:bc:62:c1:88:aa:dc:b0:c8:bb:
                   33:35:19:d5:0c:64:b9:3d:41:b2:96:fc:f3:31:e1:
                   66:36:d0:8e:56:12:44:ba:75:eb:e8:1c:9c:5b:66:
                   70:33:52:14:c9:ec:4f:91:51:70:39:de:53:85:17:
                   16:94:6e:ee:f4:d5:6f:d5:ca:b3:47:5e:1b:0c:7b:
                   c5:cc:2b:6b:c1:90:c3:16:31:0d:bf:7a:c7:47:77:
                   8f:a0:21:c7:4c:d0:16:65:00:c1:0f:d7:b8:80:e3:
                   d2:75:6b:c1:ea:9e:5c:5c:ea:7d:c1:a1:10:bc:b8:
                   e8:35:1c:9e:27:52:7e:41:8f
               Exponent: 65537 (0x10001)
   Signature Algorithm: md5WithRSAEncryption
       93:5f:8f:5f:c5:af:bf:0a:ab:a5:6d:fb:24:5f:b6:59:5d:9d:
       92:2e:4a:1b:8b:ac:7d:99:17:5d:cd:19:f6:ad:ef:63:2f:92:
       ab:2f:4b:cf:0a:13:90:ee:2c:0e:43:03:be:f6:ea:8e:9c:67:
       d0:a2:40:03:f7:ef:6a:15:09:79:a9:46:ed:b7:16:1b:41:72:
       0d:19:aa:ad:dd:9a:df:ab:97:50:65:f5:5e:85:a6:ef:19:d1:
       5a:de:9d:ea:63:cd:cb:cc:6d:5d:01:85:b5:6d:c8:f3:d9:f7:
       8f:0e:fc:ba:1f:34:e9:96:6e:6c:cf:f2:ef:9b:bf:de:b5:22:
       68:9f
```

Figure 3-9: X.509 certificate

computed over the rest of the certificate.

Kerberos tickets are the short-lived digital credentials used in Kerberos protocols. The Kerberos access control system was developed at MIT. The Kerberos system uses a trusted third party and a key distribution center (KDC). The KDC consists of two logically separate parts: an authentication server (AS) and a ticket granting server (TGS). The KDC maintains a database of secret keys for each entity on the network. By presenting a long term shared secret key to the AS, a client proves its identity and receives a ticket from the TGS. Later the client can use this ticket to get additional tickets with a session key for a service provider to authenticate it without using the shared secret. The session key can be used to secure the interaction between a client and the desired service provider[70].

IP addresses are used to identify users in an OpenID system, a lightweight decentralized identity system designed around the concept of uniform resource identifier

(URI)-based identity. OpenID was initially designed to enable lightweight control over blog commenting, thus protecting the blog from comment spam and misattribution. Since OpenID is decentralized, any website can employ OpenID software as a way for users to sign in. In the OpenID protocol, a user needs to register a URI in an opened provider such as John.openID.com. When a user logs in to the website, he provides the URI to a webpage or a resource provider to initiate an exchange between the resource provider and the identity provider, John.openID.com in this case. Then, the resource provider redirects the user to the identity provider for authentication. The user can authenticate himself by inputting the password he registered with this opened provider. At the same time, the user can decide what information the OpenID provider is allowed to give to the service provider. OpenID's key distinctions are its simplicity, its lightweight default trust model, and its ease of integration into scripted Web platforms (e.g. Drupal, WordPress, etc). OpenID solves the authentication problem without relying on a centralized website for confirmation of digital identities[70].

SAML stands for Security Assertions Markup Language (SAML), an XML-based language for the communication of security-focused identity information. SAML tokens are used as digital credentials in the SAML protocol, and were standardized under the OASIS Security Services Technical Committee (SSTC). As shown in Figure 3-10, a SAML token is designed for expressing three types of security data: authentication, authorization, and attributes. Authentication statements assert to a service provider that the principal did indeed authenticate with an identity provider at a particular time using a particular method of authentication. Authorization decision statements provide limited assertions about what actions a subject is permitted to perform on a resource given a particular set of evidence. An attribute statement asserts that a subject is associated with certain attributes. An attribute is simply a name-value pair, which can be used by queried parties to make access control decisions. The SAML token can be bound to typical message transport mechanisms which tie all of the above into interoperable patterns for common use cases (e.g. Browser Single Sign On, Web Services Security, etc.). SAML provides interoperability of secu-

```xml
<saml:Assertion
   Version="2.0"
   ID="_34234se72"
   IssueInstant="2005-04-01T16:58:33.173Z">

 <saml:Issuer>http://authority.example.com/</saml:Issuer>
 <ds:Signature>...</ds:Signature>
 <saml:Subject>
   <saml:NameID format="urn:oasis:names:tc:SAML:2.0:nameid-format:persistent">
     jygH5F90l
   </saml:NameID>
 </saml:Subject>

  <saml:AuthnStatement
    AuthnInstant="2005-04-01T16:57:30.000Z">
      <saml:AuthnContext>
        <saml:AuthnContextClassRef>
          urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport
        </saml:AuthnContextClassRef>
      </saml:AuthnContext>
  </saml:AuthnStatement>
</saml:Assertion>
```

Figure 3-10: SAML

rity related statements between multiple vendor implementations through the Liberty Alliance's Conformance Program[66].

In order to support heterogeneous identity technologies, Microsoft proposed the Identity Metasystem[31, 73]. For Microsoft, the metasystem is underpinned by the WS-* Web services standards including WS-Security[18], WS-Trust[17], WS-Metadata-Exchange[19], and WS-SecurityPolicy[34]. CardSpace, as a key component of the *Identity Metasystem*, is integrated with Windows to provide a consistent and secure component for identity interactions. When a Web application requires user identity information, it activates CardSpace. CardSpace examines the identity sought by the Web application, compares the identity request to a set of InfoCards stored in its identity repository and determines if any existing InfoCards contain information suitable to meet the request from the Web application. After choosing an appropriate InfoCard, CardSpace forwards the request from the Web application to the identity provider that created the InfoCard. The identity provider then generates a signed and encrypted security token containing the required information (e.g. name and address, employer's name and address, or credit limit). If the user approves the release of this information, then the token is sent on to the Web application. The Web application can then process the token and extract the identity information. In this way, CardSpace can eliminate the need for users to register for a new account via repetitive and cumbersome completion of forms. CardSpace's key distinctions are the consistent visual metaphor made possible by its privileged position on the Windows desktop and its ability to eliminate the user's burden of password management for authentication.

It is possible for parties in RFID networks to use one or more of the credential schemes mentioned in this section to establish the identity of a querying party. None of them is likely to dominate all authentication in RFID networks, though, due to the logistical challenge of creating a globally-unified identity scheme. Each party has its own identification scheme preference which it is difficult to persuade them to drop. Thus, in my opinion, the security framework of an RFID network must have the ability to support different kinds of identity tokens from various identity providers. CardSpace provides an example of how an authentication system can support different identity providers. However, CardSpace's information cards focus on personal profile attributes such as name, address and email, which are not enough to satisfy the representation of entities in RFID network. The entities in RFID business context have more profile attributes which must be available to the identity system. For example, a company's business relationship with a third party can be an important part of its credentials when asking for access privileges. Another setback to deploying CardSpace in RFID networks is that all communications in the CardSpace identity system are supported only by Windows applications.

The solutions presented here show that no propsed identification scheme truely supports the heterogeneous environment expected of parties and devices in RFID networks. Even the solution with the most flexible identification scheme, CardSpace, does not support extended attribute data such as business policies, which are essential to determining data access rights in RFID networks. However, the identity selecting scheme from CardSpace is a worthwhile idea which I adapted to fit in the IISS framework.

3.3.2 Policy languages

A security policy is a set of rules that govern a user's role in an RFID network and the associated data access rights. The rules are used for authorization, access control, and defining trust relationships. Security policies are defined using standard policy languages and structures. Some work has been done for defining security policy languages in the context of existing Web standards such as Privacy Preferences (P3P)[10], and XML markup languages such as EPAL[11] and XACML[64]. This section provides a brief overview of these languages.

The Enterprise Privacy Authorization Language (EPAL) technical specification was developed by the World Wide Web Consortium (W3C). EPAL is a formal language for writing enterprise privacy policies regarding data handling practices in IT systems. It concentrates on core privacy authorization while abstracting data models and user authentication over deployment details. An EPAL policy is essentially a list of privacy rules that are ordered in descending precedence – if an earlier rule applies, subsequent rules are ignored. A rule is a statement that includes a ruling, a user category, an action, a data category and a purpose. It may also contain conditions and obligations. User categories define different user categories and their hierarchies (e.g., marketing manager). Data categories define how data should be handled differently from a privacy perspective (e.g., medical record vs. contact data). Purposes model the intended service for which data is used (e.g., processing a travel expense reimbursement or auditing purposes). For EPAL to satisfy the demands of each use case, inference can be done of particular requirements. Figure 3-11 shows a policy expressed in EPAL[11].

The eXtensible Access Control Markup Language (XACML) is a declarative access control language implemented in XML. It is an extension of SAML by the OASIS standard organization meant to support a richer set of data types and functions. It supports the basic types defined by the XML Schema with additional support for data types representing email addresses and directory distinguished names. People can use XACML to express "who can do what to what". Figure 3-12 shows a policy rule defined by XACM that says "Permit John to open the door"[64].

The Platform for Privacy Preference Project (P3P) is a protocol developed by W3C. It allows web sites to present their data collection practices in a standardized, machine-readable and easy-to-locate manner. Web users are able to understand what data will be collected by the sites they visit, how that data will be used, and what data/uses they may opt-out of or opt-in to. The P3P protocol includes a standard schema for data that a web site wants to collect and a standard set of uses, recipients, data categories, and other privacy disclosures. It also includes an XML format for expressing privacy policies, a means of associating privacy policies with Web pages and cookies, and a mechanism for transporting P3P policies over HTTP. The following information as expressed here is captured by XML in Figure 3-13[10].

"This is the web site for the book Web Privacy with P3P by Lorrie Faith Cranor. We do not currently collect any information from visitors to this site except the information contained in standard web server logs (your IP address, referer, information about your web browser, information about your HTTP requests, etc.). The information in these logs will be used only by us and the server administrators for website and system administration, and for improving this site. It will not be disclosed unless required by law. We may retain these log files indefinitely. Please direct questions about this privacy policy to privacy@p3pbook.com.

This is a P3P-enabled web site. If you are using a P3P-enabled web browser you should be able to fetch the P3P policy automatically. If not, here is the P3P policy and the P3P policy reference file."

All of these policy languages define policies based on role access control. However, RFID networks need additional business context based access control. Business information, such as the current state of business relationships and records of past business transactions, will affect how a queried party determines authorization. For example, a user may want to define the following RFID network business rule: only the distribution centers in the EPC A related supply chain are able to retrieve the sales information associated with EPC A from retailer B. Thus, a querying party that is identified only as a distribution center is not allowed to access the resource.

Business rules can be quite complex, mirroring the complexity of real-world business relationships. A policy language for an RFID network must have a vocabulary rich enough to handle these complexities. This is not surprising, though, as specific application areas tend to need specialized security policy languages. For example, EPAL is designed for enterprise data management and is widely used in that community. Likewise, P3P is designed for web page data management and has had similar acceptance. However, none of this research is targeted at developing policy descriptions specific to RFID networks. There are also no tools for helping users in RFID networks evaluate queries by referring to policy rules, especially when various rulesets interact.

In conclusion, it is clear that current identity systems and policies languages are not suitable for addressing the issues of authorization and access in RFID networks. This body of research is still of high value and has inspired much of the development of the IISS framework.

```
<epal-policy default-ruling="allow" global-condition="NCName"
 version="1.2" xmlns="http://www.research.ibm.com/privacy/epal"
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.research.ibm.com/privacy/epal
epal.xsd
 http://www.w3.org/2001/XMLSchema xs-dummy.xsd ">
  <policy-information id="NCName">
    <short-description language="en">short-description</short-
description>
    <long-description language="en">long-description</long-
description>
    <property id="NCName">
      <value>value</value>
    </property>
    <issuer>
      <name>name</name>
      <organization>organization</organization>
      <e-mail>e-mail</e-mail>
      <address>address</address>
      <country>country</country>
    </issuer>
    <location>http://www.ibm.com</location>
    <version-info end-date="2001-12-31T12:00:00"
       last-modified="2001-12-31T12:00:00"
       revision-number="" start-date="2001-12-31T12:00:00"
test="false"/>
  </policy-information>
  <epal-vocabulary-ref id="NCName" location="http://www.ibm.com"
     revision-number="NCName"/>
  <condition id="NCName">...</condition>
  <rule id="NCName" ruling="allow">
      <short-description language="en">short-description</short-
description>
      <long-description language="en">long-description</long-
description>
      <property id="NCName">
        <value>value</value>
      </property>
      <user-category refid="NCName"/>
      <data-category refid="NCName"/>
      <purpose refid="NCName"/>
      <action refid="NCName"/>
      <condition refid="NCName"/>
      <obligation refid="NCName">
        <parameter refid="NCName">
          <value>value</value>
        </parameter>
      </obligation>
  </rule>
</epal-policy>
```

Figure 3-11: EPAL[11]

```xml
<Rule
      RuleId=""
      Effect="Permit">
    <Description>
        John can open the door.
    </Description>
    <Target>
        <Subjects>
            <Subject>
                <SubjectMatch
                      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue
                 DataType="http://www.w3.org/2001/XMLSchema#string">John</AttributeValue>
                    <SubjectAttributeDesignator
                          AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
                          DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </SubjectMatch>
            </Subject>
        </Subjects>
        <Resources>
            <Resource>
                <ResourceMatch
                      MatchId="urn:oasis:names:tc:xacml:1.0:function:anyURI-equal">
                    <AttributeValue
                 DataType="http://www.w3.org/2001/XMLSchema#anyURI">door</AttributeValue>
                    <ResourceAttributeDesignator
                          AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
                          DataType="http://www.w3.org/2001/XMLSchema#anyURI"/>
                </ResourceMatch>
            </Resource>
        </Resources>
        <Actions>
            <Action>
                <ActionMatch
                      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue
                 DataType="http://www.w3.org/2001/XMLSchema#string">open</AttributeValue>
                    <ActionAttributeDesignator
                          AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
                          DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </ActionMatch>
            </Action>
        </Actions>
</Target>
</Rule>
```

Figure 3-12: XACML[2]

```
- <POLICIES xmlns="http://www.w3.org/2002/01/P3Pv1">
  - <POLICY discuri="http://p3pbook.com/privacy.html" name="policy">
    - <ENTITY>
      - <DATA-GROUP>
          <DATA ref="#business.contact-info.online.email">privacy@p3pbook.com</DATA>
          <DATA ref="#business.contact-info.online.uri">http://p3pbook.com/</DATA>
          <DATA ref="#business.name">Web Privacy With P3P</DATA>
        </DATA-GROUP>
      </ENTITY>
    - <ACCESS>
        <nonident />
      </ACCESS>
    - <STATEMENT>
        <CONSEQUENCE>Our Web server collects access logs containing this information.</CONSEQUENCE>
      - <PURPOSE>
          <admin />
          <current />
          <develop />
        </PURPOSE>
      - <RECIPIENT>
          <ours />
        </RECIPIENT>
      - <RETENTION>
          <indefinitely />
        </RETENTION>
      - <DATA-GROUP>
          <DATA ref="#dynamic.clickstream" />
          <DATA ref="#dynamic.http" />
        </DATA-GROUP>
      </STATEMENT>
    </POLICY>
  </POLICIES>
```


Figure 3-13: P3P policy[3]

Chapter 4

Interoperable Internet Scale Security Framework

The Interoperable Internet-Scale Security (IISS) Framework is a World Wide Web-based framework for an agent in RFID networks to perform authentication with another agent from a different domain by reasoning over queries and related policies described in RDF-s[28] and OWL Web Ontology Language(OWL)[20]. The IISS framework embodies five characteristics of the security scheme in RFID networks. In designing IISS, I was inspired by some of the solutions discussed in the previous chapter and wanted to extend and expand these ideas in the context of RFID security. In addition, the implementation of the IISS framework is built upon the Semantic Web, the future Web technology. In this chapter, how RDF suits the security requirements of RFID networks and how it is employed in IISS are described. Some basic RDF vocabularies that are used throughout the system are introduced. In addition, the provenance information which is a unique feature to IISS is described.

4.1 IISS features

In this section, a number of features of the Interoperable Internet Scale Security(IISS) Framework, for RFID Networks, which solve the problems with RFID network security as described in chapter 2, are discussed.

To solve the identification scheme problem, the IISS framework need to have two features:

- Decentralized identity authorities: Centralized identity providers are avoided in the IISS framework in order to handle the wide variety and large number of parties involved in RFID networks. I borrowed the identity selecting scheme from Card Space to allow relying parties and querying parties in RFID networks to reach agreement on the selection of an identity provider automatically.

- Multiple authentication scheme support: Different authentication schemes and identity techniques including digital certificates, OpenID and Kerberos tickets must be supported so that parties using different techniques or platforms remain interoperable.

To solve the problem of deciding whether or not to give access permission to unknown entities, the provenance information of Electronic Product Code(EPC) data is passed through RFID networks to identify the track of the desired EPC data for both traceability and authentication reasons. Provenance information can be used as a form of proof for authentication when two parties do not share any identity authorities.

For addressing the authorization problem, the IISS framework has the following features:

- Flexible policy language: IISS has its own business rule language so that business rules may be exchanged and inferred between agents in different domains. The language is based on the same definition of the business rule vocabularies in RFID networks.

- Dynamic access control: Resource managers are able to perform access control dynamically instead of searching static identity lists.

- Rule confidentiality: The policy structure in IISS allows enterprises to keep sensitive parts of their business rules confidential.

4.2 IISS framework

Before I describe the IISS framework, I define two terms:

1. *Relying party*: A relying party is the party that owns resource information. The relying party receives queries and performs authentication and authorization on these queries. After the authentication and authorization, the relying party can decide whether or not to give the party that makes the query access permission.

2. *Querying party*: A querying party is the party that makes queries about EPC data.

In an RFID network, a relying party needs three kinds of information related to EPC-bearing objects to decide whether or not to give permission to a querying party to access particular EPC-related information:enterprise level, business rule level, and item level. The first, enterprise level information, concerns the participating enterprises' identities. This kind of information includes an enterprise's name, an enterprise's address, and business description(e.g. a manufacturer of a particular RFID-bearing item, a shipper of a particular RFID-bearing item, a government legal authority). Business rule level information concerns rules or policies that are associated with an EPC-bearing object in a particular business context. In other words, a relying party needs to process EPC-associated policy information which determines which entities are qualified to access particular information of an EPC bearing object. The third kind of information, item level information, is about an EPC-bearing object itself, such as its provenance information or EPC-related information. The EPC related information includes what the EPC is; and when where, and at which business step the EPC has been read. An EPC's properties play an important role in relying parties' decision process. IISS is the first scheme in RFID networks that performs authentication and authorization based on the integration of the information from all of the three levels.

In order to perform authentication based on the integration of the information from all three of the levels, the IISS framework is composed of five modules which

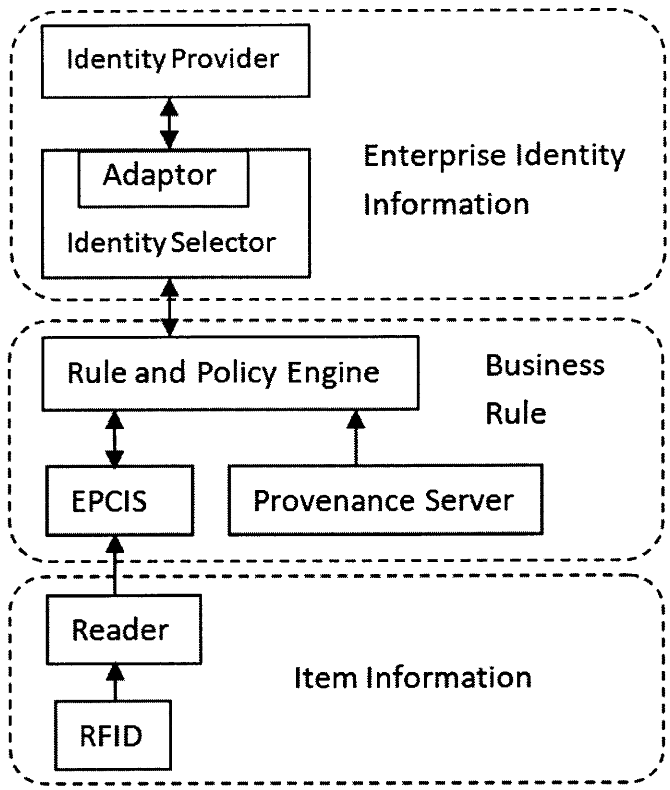

Figure 4-1: IISS framework

process different portions of the information: (1) an *EPCIS*, which stores EPC data generated by RFID readers; (2) a *Provenance Server*, which stores the provenance information of each RFID tag-bearing object; (3) a *Rule and Policy Engine*, which collects relevant policy information or proofs and generates permissions after reasoning over the collected information; (4) an *Identity Selector*; (5) and *Identity Providers*, which issue identity tokens. The *EPCIS* and the *Provenance Server* store and process item-level information. The *Rule and Policy Engine* stores and processes business policy related information. The *Identity Provider* and *Identity Selector* work together to provide enterprise identity information. Figure 4-1 illustrates the interactions between these modules.

The core component of IISS is the *Rule and Policy Engine* which processes resource access policies. Additional inputs are used by a relying party to determine a specific level of access to information that a querying party is authorized to receive. These inputs can be based on existing business relationships between the relying party and the querying party, established through a contract or other legal means. These business relationships are formalized in the *Rule and Policy Engine* by a resource administrator who defines access privileges for specific EPC related data elements controlled by his enterprise.

Authorization decisions require a wide variety of inputs, including participants' identities as determined by the authentication process. This requires that both the relying party and the querying party be involved in the authorization process. The querying party needs to collect evidence to validate its query, while the relying party needs to collect facts and policies to verify the evidence from the querying party. The *Rule and Policy Engine* is used by both parties in this process. A querying party generates evidence which is used to support its requests throug the *Rule and Policy Engine*.

A relying party uses the *Rule and Policy Engine* to validate proof packages from a querying party and control accesses to its resources, including EPCIS or other EPC-related business events information systems. The *Rule and Policy Engine* functionality for use by a relying party includes generating and storing EPC-related resource-control policies, and authenticating queries by reasoning over a *Query* instance and a *Verification Policy* instance based on the IISS ontologies. The functionality for use by a querying party include parsing rules and policies provided by a relying party, collecting relevant information from an *Identity Selector* and a *Provenance Engine* within its enterprise's domain, and generating a *Query* instance and a *Proof* instance according to received polices. Policies transmitted in IISS are described in RDF-s[28] and OWL[20] based on the *Policy ontology* defined in later chapters. The *Rule and Policy Engine* includes a reasoning component, the reasoner, which can consume data in the form of RDF-s and OWL. It accepts as inputs *Query* instances. The engine processes a *Query* instance by retrieving a policy that associates with the requested EPC and reasoning about *Query* instance over the policy via the RDF reasoner. It then checks the output from the reasoner to decide which resources the querying party is permitted to access.

The *Provenance Server* is used to store, transmit, and provide provenance information of RFID-bearing objects. In the IISS framework, provenance information is the identification of the trace of a particular RFID-bearing object. When a manufacturer ships an RFID-bearing object, the provenance server in that manufacturer's domain creates a random number to identify the object's trace. This number will be transmitted confidentially to the next party to which the object is shipped. In this way, only the parties who have processed the object know the random number. The random number can be used as a provenance challenge to verify that a company is a transition point in the RFID object's trace. If a relying party does not share a direct trust relationship, and does not have any common certificate authority with a querying party, the provenance information provided by the querying party can be used by the relying party to authenticate the query. To prevent eavesdropping and replay attacks, an EPC-related provenance challenge number can only be used for authentication once by a querying party.

The *Identity Provider* is the site or service that issues security tokens and identifies users to which it has issued token. Since the IISS framework is used in a wide-open RFID network, it is designed to support multiple kinds of identity providers including OpenID[42], SAML[66], Kerberos[70], and X.509[79] providers.

The *Identity Selector* (IS) enables a querying party to control and dispatch digital identities from different identity providers according to security demands established by the *Rule and Policy Engine*. In order to support multiple kinds of *Identity Providers*, the IS contains an adaptor that can transform various security token formats into a unified interchangeable RDF[21] format complying with the Identity Ontology of the IISS framework.

4.3 Foundations for IISS

4.3.1 Semantic Web

The Semantic Web project at W3C is targeting the data model problem from the viewpoint of improving data's interchangeability[24]. The focus of the Semantic Web effort is to popularize RDF-formatted metadata throughout the internet in much the same way that HTML has proliferated as the result of the popularity of web browsers. Data from multiple agents can be combined and exchanged by building agents that capable of consuming RDF metadata. IISS is designed to work within the framework of the Semantic Web. However, our focus is on building internet-scale interoperable security scheme. In this section, I discuss the role that the Semantic Web can play in addressing security issues faced by RFID networks such as business rules management, EPC event data sharing, distributed trusts management, and communication overhead and scalability problems caused by the traditional centralized access control scheme.

The Semantic Web is an extension of the current World Wide Web designed to represent information in a machine-readable format.The inclusion of this semantic information enables automated reasoners to be used as core components in a wide variety of Web applications and services. The World Wide Web has evolved over the last fifteen years to incorporate technologies that make it easy for humans to understand the information being presented. However, this evolution has also made it more difficult for computers to correctly identify and categorize the content of the Web. The Semantic Web addresses this problem by introducing knowledge representation languages based on XML in addition to the markup languages that have been used to develop most current web pages. The Resource Description Framework (RDF)[21], RDF Schema (RDFs)[28], and the Web Ontology Language (OWL)[20] are the Semantic Web languages used for writing ontologies that describe data models and for encoding data itself.

The Semantic Web is an ideal environment for distributed data sharing and knowledge representation. It also has advantages in the areas of policy management, trust delegation, and information access control on the Web. One of the key distinguishing feature of RDF is its use of Uniform Resource Identifiers (URIs) for identifying objects. These identifiers are globally unique and have well-defined semantics that persist when they are exchanged between systems. As the IISS framework is designed to support interoperation in globally-distributed RFID networks, the features above make RDF a natural choice for IISS's data model. To develop an approach better suited to sharing information in an open RFID environment, I define EPC, query, identity, provenance and policy ontologies in RFID networks using RDF.

Figure 4-2 shows the infrastructure of the Semantic Web. Data can be distributed in various formats on the Web. After they are translated into RDF form by some script language based on ontologies of objects, the RDF data can be stored in the RDF triple store[1]. The RDF data can be published and accessed through HTTP protocol or SparQL[75] services, counter parts of current Web services. Because RDF data are defined based on shared ontologies, agents consuming RDF are able to understand RDF data published by agents from different domains. Applications such as aggregation, visualization, browsing, inferences, and business rules can be developed using RDF data.

In the Semantic Web, everything has a URI. A URI is simply an identifier, like strings starting with "http:"or "ftp:"that you often find on the World Wide Web. Anyone can create URIs and use them for documents and concepts. URIs are the basic elements of RDF statements. An RDF is a triple statement which expresses a simple assertional logic. An RDF statement contains three parts: a subject, a predicate, and an object. A subject is the resource being described. A predicate describes the relationship between a subject and a object. An object is a resource or literal whose interpretation depends on related-predicate. Each component in RDF is referred to by a URI. For example, "a is a class"can be represented in RDF as follows.

```
<ex:a>
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
```

[1]An RDF is a triple statement, so a repository for RDFis called a RDF triple store

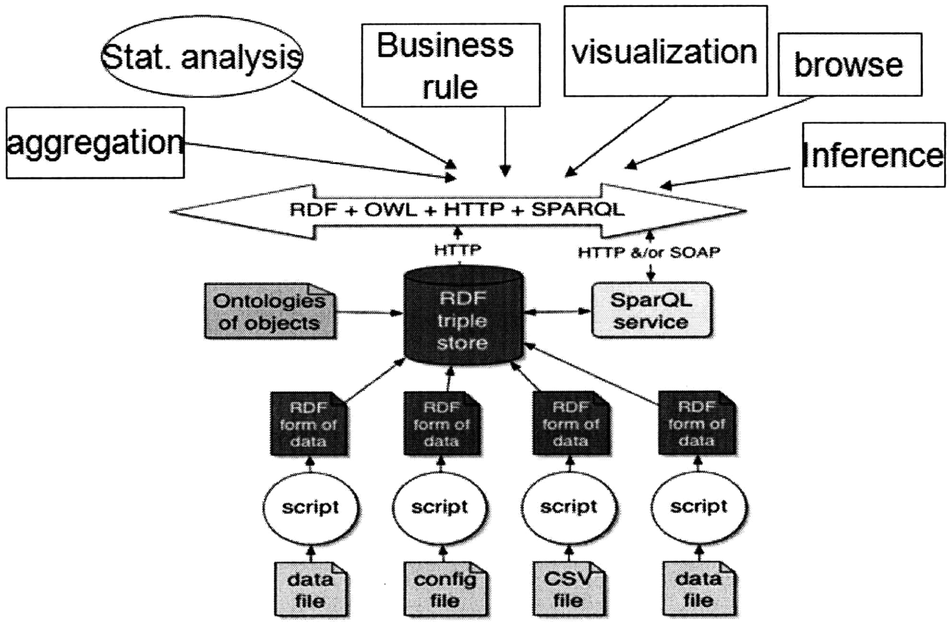

Figure 4-2: Semantic Web infrastructure

`<http://www.w3.org/2000/01/rdf-schema#Class>`

The abbreviation of the statement is

`<ex:a><rdf:type><rdfs:Class>`

The predicate "Is a"is defined at URI http://www.w3.org/1999/02/22-rdf-syntax-ns#type, which resolves to an RDF predicate. The object "class"is defined at the URI http://www.w3.org/2000/01/rdf-schema#Class, which resolves to a vocabulary defined in RDF Schema (RDFS)[28]. Any RDF-consuming agent is able to understand this statement by referring to its components' definition at those URIs.

Most basic vocabularies in the Semantic Web are defined in either RDF with the namespace http://www.w3.org/1999/02/22-rdf-syntax-ns or RDFS with the namespace http://www.w3.org/2000/01/rdf-schema. The RDF vocabulary is a set of URI references in the rdf: namespace. It is intended for representing meta data about resources in the World Wide Web, such as such as the title, author, and modification date of a Web page. The RDF vocabulary includs rdf:type, rdf:Property, rdf:XMLLiteral, rdf:nil, rdf:List, rdf:Statement, rdf:subject, rdf:predicate, rdf:object, rdf:first, rdf:rest, rdf:Seq, rdf:Bag, rdf:Alt, rdf:_1, rdf:_2 ...and rdf:value[50].

RDFS is a semantic extension of RDF. Its vocabularies include semantics of classes

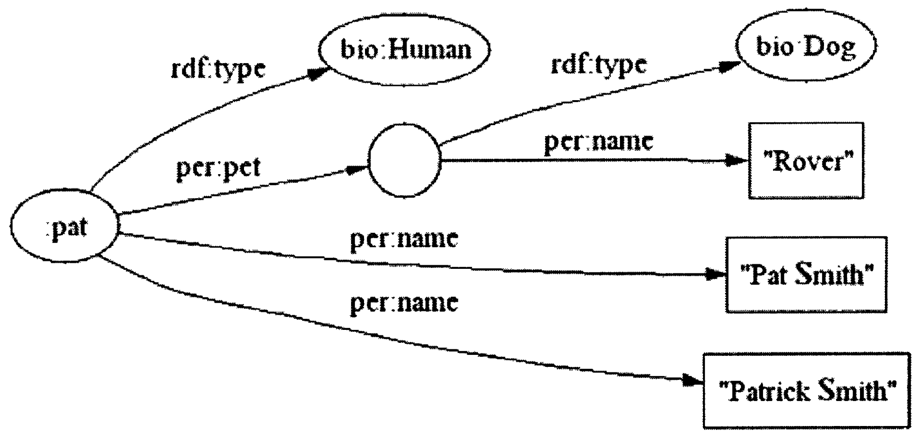

Figure 4-3: RDF graph[4]

and properties. These vocabularies are used to describe how to use RDF to describe RDF vocabularies. The class vocabularies include rdfs:Resource, rdfs:Class, rdfs:Literal, rdfs:Datatype, rdf:XMLLiteral and rdf:Property. The property vocabularies include rdfs:range, rdfs:domain, rdf:type, rdfs:subClassOf, rdfs:subPropertyOf, rdfs:label, and rdfs:comment[29].

More expressive languages such as DAML+OIL [DAML] and OWL [OWL] contain vocabularies for more specialized uses of RDF. For example, OWL defines the predicates for intersection, union and complement[72].

In the Semantic Web, an RDF graphs can be used to represent all of the defined relations among entities. An RDF graph can be represented by a set of RDF triples. The originating nodes in an RDF graph are subjects of RDF statements. Arcs can be represented by the predicates of RDF statements. And target nodes are represented by objects in RDF statements. For example, the RDF graph in Figure 4-3 can be presented by the RDF statements as follows.

```
:pat rdf:type bio:Human;
:pat per:pet [ ];
:pat per:name \Pat Smith";
:pat per:name \Patrick Smith";
[ ] rdf:type bio:Dog;
[ ] per:name \Rover".
```

This example shows how simple relationship information in the Semantic Web is formally represented in the Semantic Web by RDF statements based on shared concepts and vocabularies. Thus, information is machine-readable and interchangeable. Based on shared concepts, people can define rules in RDF form over which Web agents should be capable of inference and reasoning. In some ways, the Semantic Web can be seen as a way of allowing Web agents to become more intelligent as relevant information is provided to the agents in a ready-to-use format.

4.3.2 Provenance information

Provenance information is the data that traverse the same pathways as products or business relationships. There are two kinds of provenance information scheme that have been proposed. The first of these is the E-Pedigree scheme[14]. As shown in Figure 4-4, in the E-Pedigree scheme, when a manufacturer makes a shipment of batched items, a pedigree document will accompany the shipment listing the manufacturer, the date of shipment, the type of product, and the EPC codes of all the items. A hash of this document is then computed and signed with the manufacturer's private key. Upon receiving the goods, the wholesaler will add details of the receipt to the e-Pedigree and will again hash and sign the document. The wholesaler will then add further shipment information to the document. This process is repeated at every receipt and shipment event until the goods the the enterprise which sells directly to the general consumer.

The other kind of provenance information is a product authentication code[82, 62]. As shown in Figure 4-5, a product authentication code is a security marker stored with EPC in an RFID tag. In contrast with an EPC code, the security marker is less subject to inference by adversaries. Only parties that processed the EPC-associated RFID tag can provide the security marker for the EPC for authentication purpose.

Provenance information can help to solve trust issues in RFID networks. In some scenarios, a manufacturer may seek evidence that company X was in physical possession of the EPC in question. So company X must present the provenance information when sending a query to the manufacturer. If the provenance information is transmit-

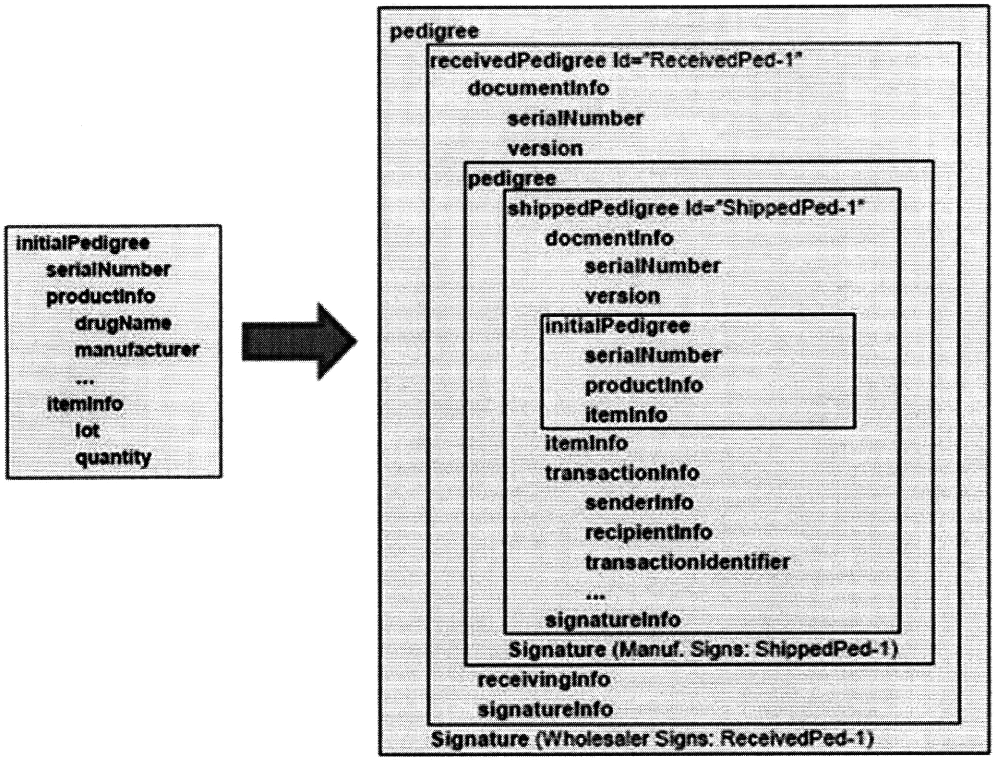

Figure 4-4: E-pedigree

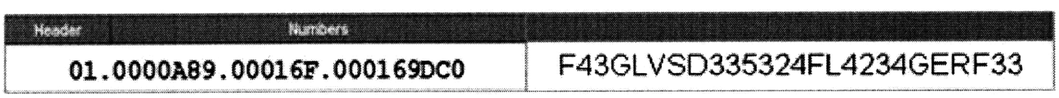

Figure 4-5: Product Authentication Code

ted secretly or bearing signatures along a multi-party supply chain, a relying party can validate the provenance information without requiring access to third-party business records.

In this thesis, a third kind of provenance information called *Provenance Challenge*, a random number which acts as a kind of light-weight provenance information, is introduced. Unlike the E-Pedigree documentation scheme or secret numbers stored in RFID tags, provenance challenges are transmitted confidentially through provenance servers according to a product's shipping plans. If a product is physically diverted to an unauthorized party, that party will not gain access to EPCIS data because the party's provenance service is not able to receive the provenance challenge unless it is in the product's shipping plan.

An example of how provenance challenges are passed through shipping plans is shown in Figure 4-6, where a batch of products are shipped out from a manufacturer. The products will go to different retailers through different distribution centers. When the manufacturer ships an RFID-bearing object, the provenance server in that manufacturer's domain initiates a random number to identify the object's trace. This number will be transmitted confidentially to the next party to which the object is shipped. In this way, only the parties that are authorized to process the object know the random number. The provenance challenge can verify that a company is a transition point in the RFID object's trace or a company has made an observation of the RFID object in facilities under its control. If a relying party does not share a direct trust relationship and does not have a common certificate authority with a querying party, the provenance challenge can be used by the relying party to authenticate the querying party.

In comparison with E-Pedigrees, provenance challenges have the advantage that they are easily generated and very lightweight. As a result, provenance challenges will consume less system's resource such as bandwidth, computing power, and memory than E-Pedigrees. In comparison with product authentication codes, provenance challenges can prevent unauthorized parties in a diverted track from reading and abusing EPC-related secret markings.

The IISS framework is shown to be the first security mechanism that performs authentication based on an aggregation of business context, enterprise information, and RFID tag information. The IISS framework provides users the flexibilty to adopt various identity technologies and consume different provenance information.

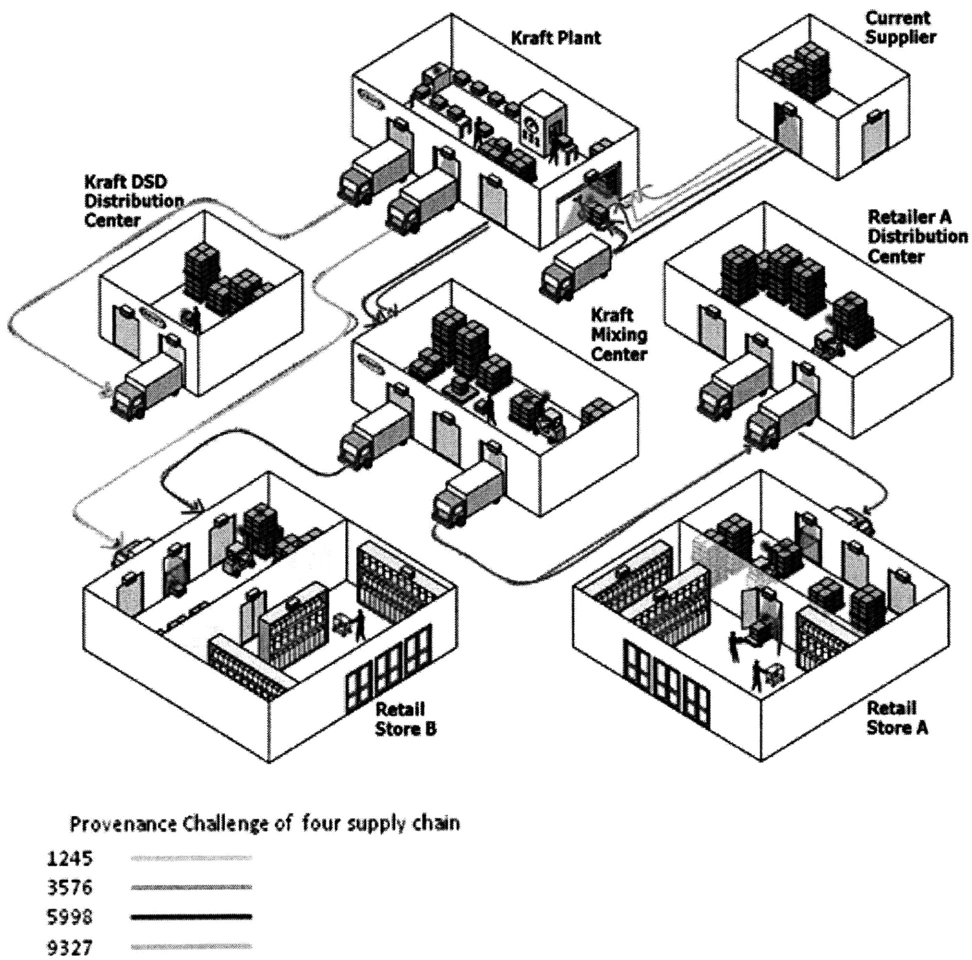

Figure 4-6: Product challenges

Chapter 5

IISS Infrastructure

This chapter presents the IISS framework's infrastructure and abstracts the idea of storage containers in IISS. The IISS's backend, which consists of a multi-agent environment with a shared RDF-based repository acting as common reference, is characterized. The mechanism for interoperating among the IISS framework's distributed files and modules on the web is also presented and justified.

5.1 IISS stores and components

Figure 5-1 represents the infrastructure of the IISS system. The infrastructure is composed by a data storage layer, a data processing layer, and a data resource layer.

At the bottom of IISS infrastructure sits the *IISS ontology store.* It provides general vocabularies describing information in RFID networks. Because all the agents in the IISS framework share an underlying ontology store, they can interoperate with each other by treating the store as a "blackboard". Above the *IISS ontology store*, there are six RDF stores serving as the foundation of the IISS system that utilizes the IISS data model. The six RDF stores are *identity store, query store, proof store, requirement policy store, verification policy store*, and *EPC store.* Those stores' functionalities are as follows:

- EPC Store: Stores EPC Instances. Each EPC instance contains an EPC, the EPC-related provenance information, and its policy files.

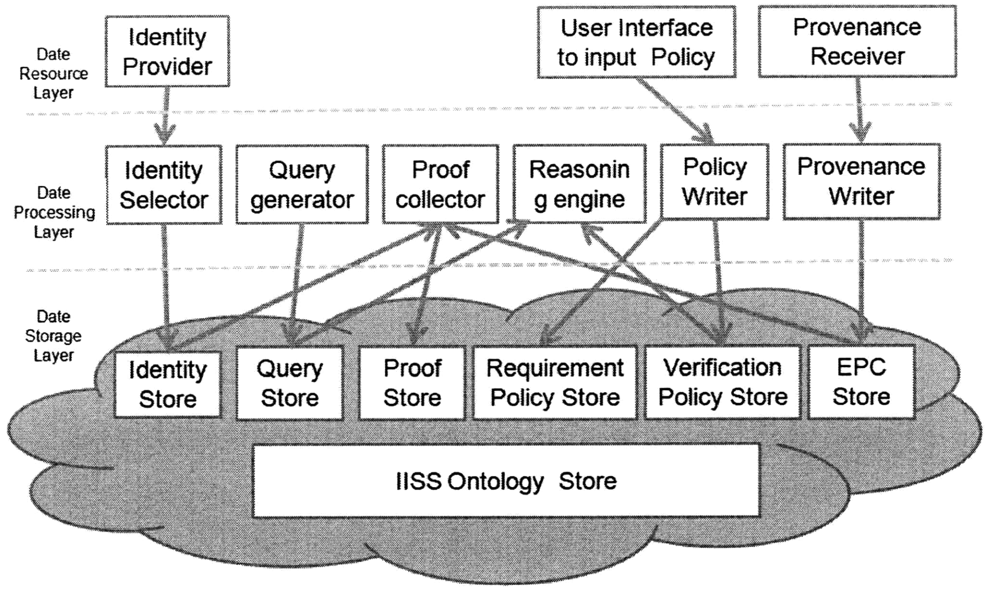

Figure 5-1: IISS infrastructure

- Verification Policy Store: Stores policies that are used to verify incoming queries.
- Requirement Policy Store: Stores policies that describe what information querying parties are required to collect to prove the validation of their queries.
- Query Store: Stores Query instances. Each Query instance contains its targeted EPC and its related proof file.
- Proof Store: Stores collected proof information that is used to validate a query for a particular EPC.
- Identity Store: Stores enterprises' identity information. An identity store is able to contain identity tokens from multiple identity providers.

On top of these RDF stores sit components that process information residing in the RDF stores. Those components are *identity selector, query generator, proof collector, reasoning engine, policy writer*, and *provenance writer*. These components are simply resources in IISS that store their RDF statements in respective RDF stores and have pieces of executable code called methods that can be invoked to the input and generate the statements in the form of RDF. The *policy writer* is responsible for generating an EPC-related policy pair (a requirement policy and a verification policy) according to inputs from resource managers and storing these policies to right policy stores. The *provenance writer* is responsible for receiving provenance information from its previous provenance server in an EPC-associated supply chain and adding provenance information to its *EPC store.* The *query generator* initiates query statements which describe queried EPCs and their supporting proof files. After a query statement is generated, it is stored in the query store by the *query generator.* The *identity selector* retrieves its enterprise's identity tokens from *identity providers* and stores those identity tokens into its *identity store.* The *proof collector* retrieves provenance statements from the EPC store and identity token information from the identity store and combines them into proof files that are used to support queries.

Sitting at the core of the IISS infrastructure is the *reasoning engine.* The *reasoning engine* retrieves query statements from querying parties and reasons them over verification policies. The *reasoning engine* makes final decisions on whether or not to give incoming queries access permissions.

On top of these middle tier components is the data resource layer. The data resource layer includes servers which provide information in the IISS framework. They are *identity providers, policy input proxies* and *provenance servers. Identity providers* can be any authentication authorities on the Web. The *policy input proxy* is a user interface tool allowing resource managers who are not familiar with RDF formats to input English-like representation of business methods, entities, conditions and actions. *Provenance servers* transmit, receive and provide EPC-related provenance information.

5.2 Functionalities of agents

Agents in IISS play two roles. One is as a relying party; the other is as a querying party. As a relying party, an agent uses the policy input proxy, the provenance server, the reasoning engine, the policy writer and the provenance writer in IISS to perform the following four functions.

1. Store EPCs and their provenance information when receiving RFID bearing

objects.

2. Specify the requirement policy and the verification policy for a given EPC.

3. Reason over a verification policy and a proof provided by a querying party using Cwm[23], a forward-chaining reasoner used in the Semantic Web project to decide whether or not a query is valid.

4. Take actions based on results of validation: either provide queried information or deny a query.

As a querying party, an agent uses the identity selector, the reasoning engine, the proof collector, and the provenance writer in the IISS framework to perform the following three functions.

1. Store EPCs and their provenance information when receiving an RFID bearing object.

2. Retrieve proof information from its *identity store* and *provenance store* and generate proof statements according to requirement policies for targeted EPCs provided by relying parties.

3. Use its *query generator* to generate queries which contain statements about targeted EPCs and their associated proof files.

5.3 File Distribution

IISS is developed within the Semantic Web Framework. In the IISS framework, all the RDF statements are stored in the form of Notation 3 (N3)[22] files. Notation 3 is a compact and readable alternative to RDF's XML syntax, and is extended to allow greater expressiveness. Each N3 file has a suffix of "n3". One feature of the Semantic Web is that RDF statements can be distributed anywhere on the Web as long as there is an URI to identify the RDF statements. As shown in the following figure, all the N3 files in the IISS framework are deployed throughout the Web.

The distribution of statements in IISS vis the Web should follow four rules:

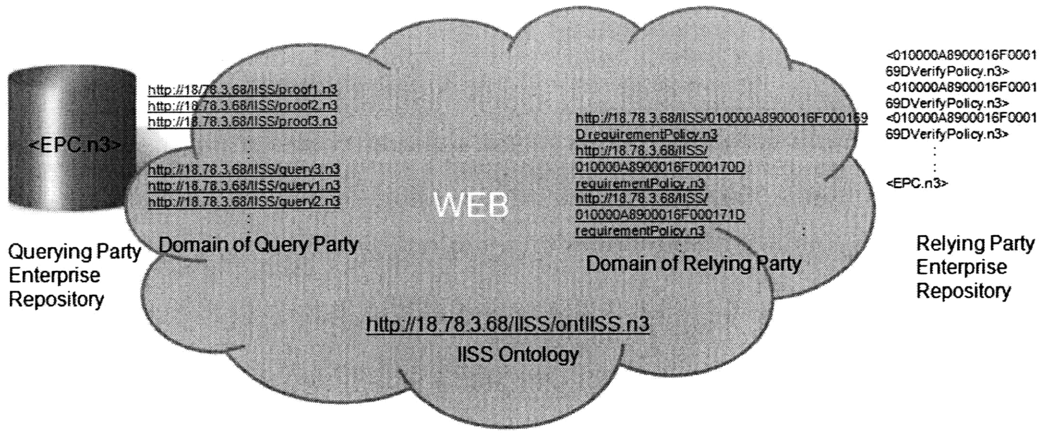

Figure 5-2: File distribution in the IISS framework

1. Information and instances that are common to all the agents should be defined as ontologies which are accessible for all the agents throughout RFID networks.

2. Information or instances unique to a certain enterprise should be defined within the enterprise's domain.

3. Files with sensitive information, such as N3 files in *verification policy stores* and the *EPC stores*, are kept within enterprises' repositories. These files are not published on the Web.

4. Files without sensitive information, such as N3 files in *requirement policy stores, query stores* and *proof stores*, are published with URIs at which other agents are able to retrieve RDF statements in these files to do inference.

According to these four rules, all the files in IISS are deployed as follows. An N3 file in the IISS *ontology store* that defines the IISS ontologies is published on the Web with the URI address, http://18.78.3.68/IISS/ontIISS.n3. All the agents in the IISS framework share this file to generate their own RDF statements based on the common vocabularies defined in the IISS ontology file. Besides the IISS ontologies, some instances, including identity issuers and identity formats, are also defined in the ontIISS.n3 files because those instances are common knowledge and will be referred by both relying parties and querying parties.

Each enterprise has a *requirement policy store.* In a *requirement policy store*, each EPC of which an enterprise owns information has a file describing *requirement policies* for accessing the EPC-related resources, such as PO and EPC events. Because a requirement policy only describes what kinds of proof a querying party needs to provide, it does not contain any confidential information, it is published with an EPC embedded URI, such as http://18.78.3.68/IISS/010000A8900016F000169DCDrequirementPolicy.n3.

Each enterprise has a *verification policy store.* In a *verification policy store*, each EPC about which an enterprise owns information has a file describing verification policies for validating queries for the EPC-related resources, such as POs or EPC events. Because *verification policy* files contain confidential information, they are not published on the Web. They are stored in an enterprise intra-repository with an EPC embedded file name, such as c://010000A8900016F000169DverificationPolicy.n3.

Each enterprise has a *query store.* After being generated by a *query generator*, each query file in a *query store* is published with a URI, such as http://18.78.3.68/IISS/-query1.n3. A query file can be reused if some other relying parties provide the same *requirement policy* for a same EPC-related query. Query files can be deleted right after the queries are finished for security reasons.

Each enterprise has a *proof store.* After being generated by a proof collector, each proof file in a *proof store* is published with a URI, such as http://18.78.3.68/IISS/-proof1.n3. A proof file can be reused if some other relying parties provide the same *requirement policy* for a same EPC-related query. Proof files can be deleted right after the queries are finished for security reasons.

Each enterprise in RFID networks has an *EPC store.* An N3 file in an *EPC store* stays in an enterprise's intra-repository, such as c://EPC.n3. The EPC.n3 file describes two things. First, it matches EPC codes processed by an enterprise with their *verification policy* files. Second, it matches EPC codes with their provenance information. Because provenance information is a secret, the EPC.n3 file should not be published on the Web. In addition, enterprises do not want to publish the EPCs they have processed because this information can leak some confidential business

File	*FileName*	*Example*
Requirement Policy	Domain Name + EPC + "RequirPolicy"+ ".n3"	http://18.78.3.68/IISS/010000A8900-016F000169DCDrequirementPolicy.n3
Verification Policy	Domain Name + EPC + "VerifyPolicy"+ ".n3"	010000A8900016F000169Dverification-Policy.n3
EPC file	EPC.n3	EPC.n3
Query	Domain Name + "query"+ queryID + ".n3"	http://18.78.3.68/IISS/query1.n3
Proof	Domain Name + "proof"+ proofID + ".n3"	http://18.78.3.68/IISS/proof1.n3

Table 5.1: File name convention in the IISS framework

relations. For these two reasons, EPC.n3 files are kept within enterprises' repositories. A *provenance server* in an enterprise can input provenance information of an EPC by invoking an input function in a *provenance writer* when provenance information is received from an upstream provenance server. At the same time, the *provenance writer* writes an entry which associates the EPC with a verification policy file value into the EPC.n3 file. When a proof collector is required to retrieves provenances, it retrieves provenances from EPC.n3 files in its own domain by parsing it.

To achieve interoperability among all the agents in RFID networks and match EPCs with policies within a domain easily, the IISS framework assign URIs and file names to all of the files in the IISS RDF stores based on the name convention as shown in Table 5.1.

The depolyment of those files in distributed RDF stores needs to make RDF statements referrble and keep them secret at the same time. In order to solve this issue, the IISS protocol allows relying parties expose file names of confidential RDF files. For example, a relying party sends a file name of an EPC N3 file together with a URI of an EPC-related requirement policy file to a querying party. By referring the file name, a querying party is able to refer the RDF statements in that EPC N3 file. As long as files' URIs are not published, intruders cannot access those files even though their file names are published.

Chapter 6

Ontologies in IISS

As is the case whenever logic is to be used, care must be taken to precisely define concepts and relationships. In order to realize automatic access control based on rules and policies in RFID networks, a set of ontologies describing concepts and relationships of entities in the security domain and the EPC domain in RFID networks must be defined. In this chapter, the ontologies developed for modeling the entities involved in authentication and authorization decision in RFID networks are presented. These entities lay the foundation for representing the information exchanged in the IISS framework. Those entities are *Query, EPC, Identity, Policy*, and *Provenance Information.* How the ontologies are described in the form of RDF are also explained. With this foundation, how the IISS framework manages to link and integrate data distributed in widely distributed RFID networks via the IISS ontologies is discussed.

6.1 Introduction to Ontologies

In recent years, ontologies have become a topic of interest in computer science. As defined by Tom Gruber[48], *"An ontology is an explicit specification of a conceptualization. The term is borrowed from philosophy, where an Ontology is a systematic account of Existence. For AI systems, what 'exists'is that which can be represented."* A "conceptualization"means abstraction of some phenomenon in the world by identifying relevant concept of that phenomenon. "Explicit"means that types of concepts used and constraints on their use are explicitly defined. This definition is often extended by three additional conditions: *"An ontology is an explicit, formal specification of a shared conceptualization of a domain of interest."* "Formal"reflects the notion that an ontology should be described in machine readable way. "Shared"refer to the fact that an ontology captures consensual knowledge which is not private to some individual, but accepted by a group. "The domain of interest"indicates that domain ontologies are only interested in modeling the parts which are relevant to the application. In computer science, an ontology is a measure to make an explicit commitment to share common knowledge among an interested community. Anyone can use ontologies to describe their own data. Therefore, an ontology provides a scheme for general purposes which can be reused by different applications[80, 74].

At present, methods and tools already exist to support engineering of ontologies (e.g.[80, 74]). There are several existing ontology markup and representation languages worth mentioning: RDF[21], RDF Schema[28], and OWL[20]. Besides those languages, some other languages such as DAML+OIL[33], EER[65], UML[7], Topic Maps[26], MOF[8], and XML Schemas[39] are used to describe ontologies as well.

6.2 Concept and relations - Ontologies in the IISS framework

To model rules as mechanism for declaratively specifying business logic and authentication and authorization policies in RFID networks, some ontologies, namely concepts, relations, and instances about RFID and security are introduced in the IISS framework. The IISS ontologies reflect RFID domain-specific entities and allow for description of security logic supporting RFID business processes in the form of rules. The IISS ontologies enable agents to describe declaratively their domain-specific security properties in a machine readable manner. Thus, by using an inference engine such as Cwm[23], these security properties can be applied automatically during runtime.

To model data flow in the IISS framework, the IISS ontologies include the *Prove-*

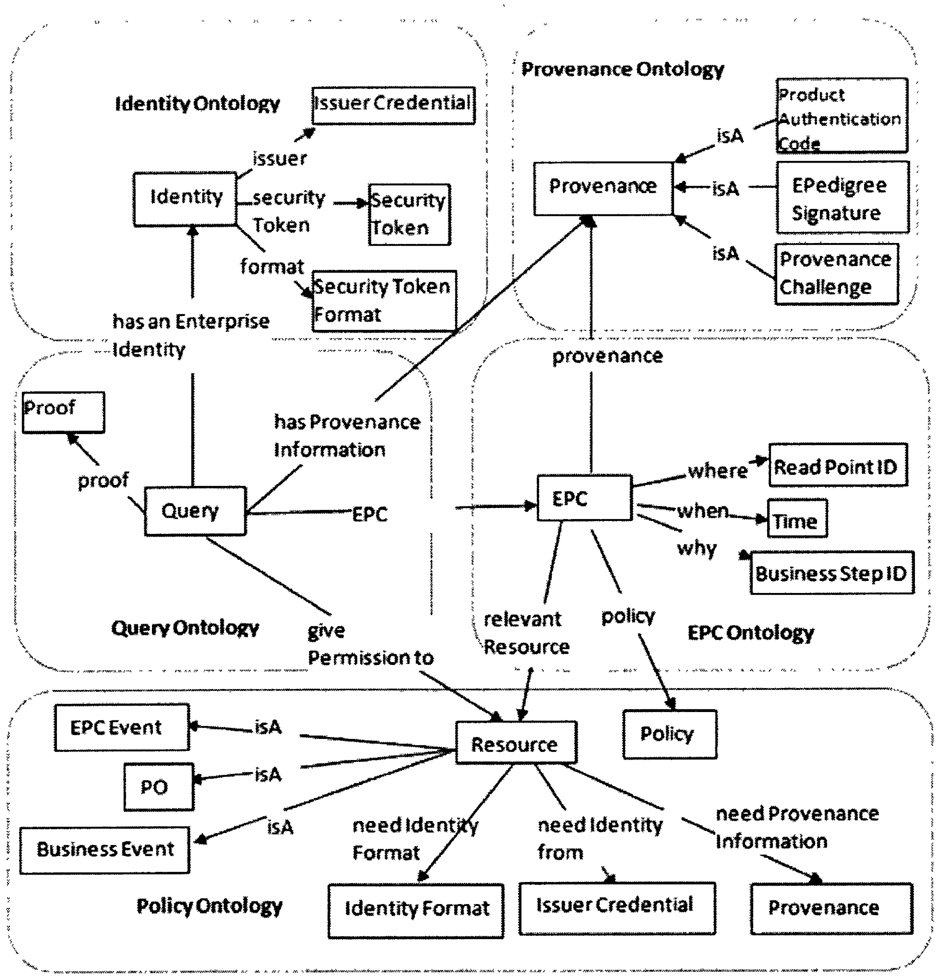

Figure 6-1: IISS ontologies

nance Ontology, Identity Ontology, EPC Ontology, Query Ontology and *Policy Ontology.* These ontologies are illustrated in Figure 6-1. All agents in IISS can describe their own data as instances of the classes in those onotologies. The data are connected to each other via the properties defined in the ontologies. Given the ontologies and the semantics relations among them, the IISS framework can make logical inferences about the consistency between evidences and policy rules.

The IISS *Query* class is an abstraction of a query for a particular EPC in IISS. The *Query* class has five properties that link the *Query* to the five other ontologies. The *EPC* property defines the EPC to be queried. The relation that the property represents between a *Query* and an EPC code can be modeled as EPC(Query, EPC). The *has Provenance Information* property links the *Queries* to the querying parties' provenance information of the EPCs being queried. The relation that the property represents between a *Query* and provenance information can be modeled as hasProvenanceInformation(Query, Provenance). The *has-an-Enterprise-Identity* property is set to the querying party's identity information being requested. The relation that the property represents between a *Query* and an *Identity* can be modeled as has-an-Enterprise-Identity(Query, Identity).

An enterprise may own multiple identities issued by different identity authorities. Only the identity satisfying the relying party's policy will be referred to. The objects of both the two properties are instances of the classes-*Identity Format* and *Identity Credential*-in the *Identity Ontology.* The *proof* property links a *Query* to a proof file that is used to support the validation of the query. The relation that the property represents between a *Query* and a proof file can be modeled as proof(Query, Proof). Proof files are defined as instances of the *Proof* class in the *Query Ontology.* The *give-permission-to* property associates a *Query* to the resources that a querying party is permitted to access. The resources can be purchase orders, EPC events or business events. This semantics is used in *verification policies* to describe a relying party's action after it validates a *Query.*

The *Policy Ontology* in the IISS provides vocabularies describing business rules in RFID networks. It formulizes a business rule language specific to RFID networks, based on which users can define various interchangeable rule sets. The *Policy Ontology* consists of four properties: *need-Identity-from, need-Identity-Format, need-Provenance-Information*, and *give-Permission-to* and two classes: *Resource* and *Policy.* The *need-Identity-from* property specifies from which identity issuer an identity is required to fulfill a particular resource's access control policy. The relation that the property represents between a *Resource* and an *Identity Issuer* in a *requirement policy* can be modeled as need-Identity-from(Resource, Identity Issuer). The *need-Identity-format* property specifies what kind of identity token is required to access a particular resource. The relation that the property represents between a *Resource* and an *Identity Format* in a *requirement policy* can be modeled as need-Identity-Format(Resource, Identity Format). The *give Permission to* property links *Query* to the resource that the querying party is permitted to access. The relation that the property represents between a *Query* and a resource in a policy can be modeled as givePermissionto(Query, Resource). The *Resource* class defines EPC relevant resources. The EPC relevant resources can be purchase orders, EPC events in EPCIS, or business events. They are defined in the *Policy Ontology* as subclasses of the *Resource* class. The *need-Provenance-Information* refers to the provenance information required by business rules to make an authentication decision. The properties *need-Identity-from, need-Identity-format*, and *need-Provenance-Information* are used for describing requirement policies in the IISS framework to specify criteria that a querying party needs to satisfy to access particular resource. The *Resource* class is used in both requirement policies and verification policies. In *requirement policies*, the *Resource* class is used as subjects in requirement statements. In *verification policies*, the *Resource* class is used as objects of the predicate-*give-Permission-to*-in action statements. The *Policy* class defines policy files which contain authentication and authorization policy statements.

The *EPC Ontology* uses the semantics defined in EPCIS standards. The assertions of EPC events in EPCIS include (1) the object(s) or other entities that are the subject of the event; (2) the date and time; (3) the location at which the event occurred; (4) and the business context[43]. Therefore, the properties for describing the properties about EPC are *where, when* and *why*. Those properties describe the relationships between an EPC and its business contexts when the EPC is read, which are where(EPC, Read Point ID), when(EPC, time) and why(ECP, Business Step ID). In addition to those properties, the EPC class also includes a *policy* property and *relevant Resource* property. The *relevant Resource* property links an EPC to its related resources. The relation it represents between an EPC and a *Resource* can be modeled as relevant-Resource(EPC, Resource).The *policy* property is used for associating EPCs with the access control verification policies for their relevant resources. The relation it represents between an EPC and a *policy* can be modeled as policy(EPC, Policy). Because an EPC code is the key to various information such as EPC events, POs or other business events, all the queries in RFID network will be

EPC-related. That is the reason why EPC is also the key to the IISS policies. In this way, querying parties and relying parties can retrieve the policies of their interested resources via EPCs.

The *Identity* class is made up of three properties that are used to describe the identity information. The *issuer* property is set to either the credential or the URI of the issuer who authorizes the credential to be what a querying party claims to be. The *security Token* property is used to refer to a security token that contains credential information. The *format* property describes the format of the security token. Its value can be SAML[66], Kerberos[70], X.509[79] or other credential formats. The relationships that these properties represent can be modeled as issuer(Identity, Issuer Credential), security-Token(Identity, Security Token) and format(Identity, Security Token Format).

The *Provenance* class has three subclasses. One is the *provenance challenge* as described in chapter 4; the other is the *Epedigree Signature.* I define the *Epedigree Signature* class in order to take advantage of Epedigrees[14] as one kind of provenance information if Epedigrees are generally adopted by the industry. The third subclass in *Product Authentication Code* class which is used to represent product authentication code in[82, 62].

Of course, the definition of ontologies in RFID networks is not limited by the definition of the IISS ontologies here. The IISS ontologies can be enriched and extended when more business relationships or security requirements need to be accounted for during data exchange in RFID networks.

6.3 Ontology implementation

Some specifications defined in the Semantic Web-RDF[21] and RDF Schema[28]- are used to represent IISS ontologies. RDF is in the form of a triple: (subject, predicate, object). The object data in ontologies can be described by the subject of an RDF. The property data in ontologies can be described by the predicate of an RDF. The value for a property or the connection with another object by the property can be described by the object of an RDF.

The following two specifications define the prefix "rdf" to refer to the RDF namespace http://www.w3.org/1999/02/22-rdf-syntax-ns#, and the prefix "rdfs" to refer to the URI-reference of RDF Schema http://www.w3.org/2000/01/rdf-schema# where the core vocabulary of RDF semantics is defined.

```
@prefix rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix rdfs:  <http://www.w3.org/2000/01/rdf-schema#>.
```

By putting the two specifications on top of the IISSont.n3 file, vocabularies defined in these two namespaces can be referred to by putting "rdf:" or "rdfs:" in front of the vocabularies. The vocabularies used to define IISS ontologies are rdf:type, rdf:Class, rdf:Property, rdfs:subClassOf, rdfs:domain, and rdfs:range. rdf:type is used to state that a resource is an instance of a class. rdf:Class means a class of resources. rdf:Property means an RDF property. rdf:subClassOf is used to represent the relationship that all the instances of one class are instances of another. rdfs:domain is used to state that any resource that has a given property is an instance of one or more classes. rdfs:range is used to state that the values of a property are instances of one or more classes.

Here are some examples of definitions of the IISS classes. In these examples, the *Identity* class, the *Security Token* class, the *Query* class, the *Resource* class, the *PO* class, and the *Business Event* class are defined. At the same time, the inheriting relationship between *PO* and *Resource* and the inheriting relationship between *Business Event* and *Resource* are defined.

```
:Identity a rdf:Class.
:SecurityTokenFormat a rdf:Class.
:Query a rdf:Class.
:Resource a rdf:Class.
:PO a rdf:Class; rdfs:subClassOf :Resource.
:BusinessEvent a rdf:Class; rdfs:subClassOf :Resource.
```

Here are some examples of definitions of the IISS properties. In these examples, the *has an Enterprise Identity* property, the *has Provenance Information* property and the *EPC* property are defined. The subjects and objects of these properties are specified in the definitions via the words: rdfs:domain and rdfs:rang.

```
:phasanenterpriseidentity a rdf:Property;
      rdfs:domain :Query;
      rdfs:range :Identity.
:phasProvenanceInformation a rdf:Property;
      rdfs:domain :Query;
      rdfs:range :Provenance.
:pepc a rdf:Property;
      rdfs:domain :Query;
      rdfs:range :EPC.
```

Here are two exemplar instances of the IISS ontologies. The first instance is *OpenID* which is a security token format. The second instance is *Gevernment* which is an identity issuer. These two instances are defined in the IISSont.n3 because these two instances are common knowledge that can be shared by any agent in RFID networks.

```
:OpenID a :SecurityTokenFormat.
:Government a :IssuerCredential.
```

6.4 Implementation of the IISS data model

The IISS ontologies provides users with vocabularies to represent information transmitted in the IISS framework. This section demonstrates how to link the data and files in RFID networks based on IISS ontologies in the semantic web framework by explain an example set of linked data in the IISS system.

In Figure 6-2, there is a sample query as a instance of *Query* in the IISS ontologies. The query's target EPC is 010000A8900016F169D which is defined in EPC.n3 file as an instance of *EPC* class in the IISS ontologies. The proof file to support this query can be retrieved at the web adddress http://18.78.3.68/IISS/proof89.n3 . This information about the sample query is stored at the web site http://18.78.3.68/-

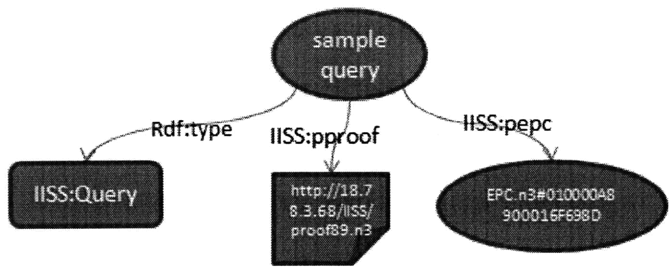

Figure 6-2: RDF graph of a *Query* instance

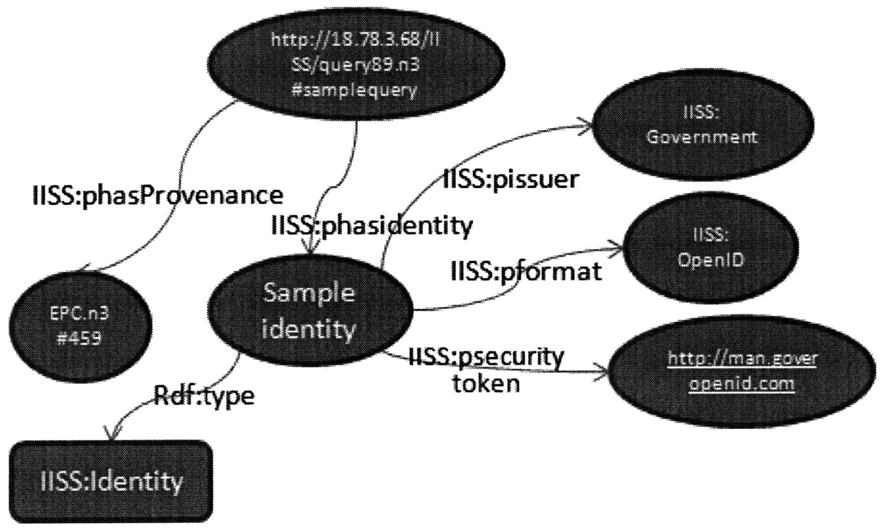

Figure 6-3: RDF graph in a proof file

IISS/query89.n3.

Figure 6-3 represents the data stored in the proof file for the sample query. In the proof file, the sample query is referred by the URI http://18.78.3.68/IISS/query-89.n3#samplequery . The proof file describes that a querying party knows the provenance challenge of EPC 010000A8900016F169D which is 459. The provenance challenge is defined in EPC.n3 file with an URI-EPC.n3#459. And the querying party has an identity which is defined as an instance of *Identity* in the IISS ontologies. The identity is issued by the *Government*, and its format is *OpenID*. And the identity token is the URI http://man.goveropenid.com . The *Government* and *OpenID* is defined as common knowledge in the IISSont.n3 file.

Figure 6-4 represents relevant information of EPC 010000A8900016F169D described in EPC.n3 file. The EPC 010000A8900016F169D is defined as a instance of

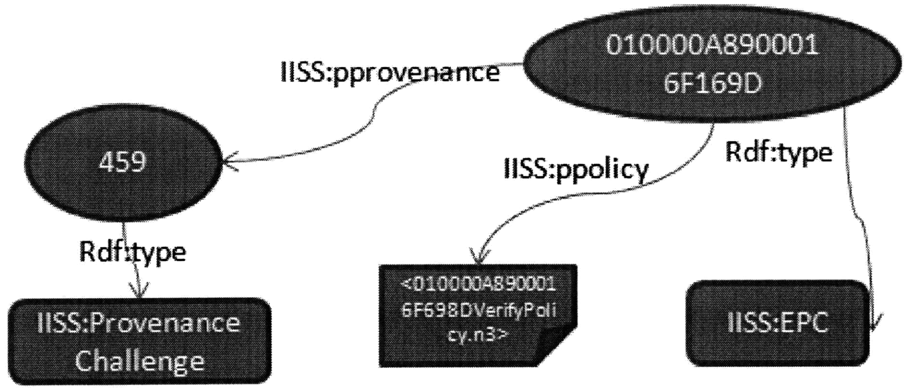

Figure 6-4: RDF graph of an *EPC* instance

the *EPC* class in the IISS onotologies. The provenance challenge of the EPC is 459 which is an instance of the *Provenance Challenge* class in the IISS ontologies. The verification policies of the EPC 010000A8900016F169D related resources are stored in the file 010000A8900016F698DVerifyPolicy.n3.

The discussion above highlights that all the data and files in the IISS framework are linked by the properties defined in the IISS ontologies. The IISS framework adopts the RDF as a means for describing information according to IISS onotologies. The adoption of RDF enables IISS to take advantage of some formats for information encoded in RDF at present and the inference tools developed in the Semantic Web project.

The following three specifications define the prefix "IISS"as a reference to the IISS ontologies' namespace http://18.78.3.68/IISS/ontIISS.n3# , the prefix "EPC"as a reference to the EPC's namespace EPC.n3# where relying parties define the EPCs they own, and the prefix "query"as the URI-Reference http://18.78.3.68/IISS/query89-.n3# where the sample query is defined.

```
@prefix IISS: <http://18.78.3.68/IISS/ontIISS.n3#>.
@prefix EPC: <EPC.n3#>.
@prefix query:  <http://18.78.3.68/IISS/query89.n3#>.
```

By putting the three specifications on top of the N3 files in the IISS framework, instances defined in the three namespace can be referred to by putting "IISS:", "EPC:", http://18.78.3.68/IISS/query89.n3

```
:samplequery rdf:type IISS:Query ;
             IISS:pproof <http://18.78.3.68/IISS/proof89.n3> ;
             IISS:pepc <EPC.n3#010000A8900016F698D> .
```

http://18.78.3.68/IISS/proof89.n3

```
query:samplequery IISS:phasProvenanceInformation <EPC.n3#459> .
:sampleidentity rdf:type IISS:Identity ;
             IISS:pissuer IISS:Government ;
             IISS:pformat IISS:OpenID ;
             IISS:psecuritytoken <http://man.goveropenid.com>.
query:samplequery IISS:phasanenterpriseidentity :sampleidentity .
```

an EPC entry in <EPC.n3>

```
:459 rdf:type IISS:ProvenanceChallenge .
:010000A8900016F698D rdf:type IISS:EPC ;
             IISS:ppolicy <010000A8900016F698DVerifyPolicy.n3> ;
             IISS:pprovenance :459 .
```

Figure 6-5: The IISS instances represented by RDF statements

and “query”in front of the instances. Then I come up with the RDF representation of the data described above as shown in Figure 6-5.

Chapter 7

Policy Structure, Rule Engine and Policy Input Proxy in IISS

In this chapter, the policy structure of the Interoperable Internet-Scale Security(IISS) framework is presented. The means by which the policy files are deployed and published is demonstrated. After that, I discuss how the implementation of the IISS data model supports a basic forward chaining reasoning mechanism, which can be used for business rule inference, and how a business rule engine reasons over *Query* and related policies. In the end, the policy input proxy developed to generate EPC-related policy rules in RDF format via a *policy writer* is described. With the policy input proxy, users who are not familiar with the RDF format are ensured to be able to use the English-like representation of business methods, entities, conditions, and actions.

7.1 Policy structure in the IISS framework

It is important that any solution to the data exchange problem in RFID networks does not make unrealistic simplifying assumptions about authorization policies used by the participating enterprises. A worthwhile solution should, instead, be able to accommodate a participant's desire to retain full control over authorization decisions regarding their own data. Business authorization policies and their associated rules in RFID networks can be quite complex, mirroring the complexity of real-world business relationships. The IISS framework provides a dynamic mechanism for performing access control of EPC-related resources, providing the necessary per-enterprise access authorization flexibility which traditional role based access control solutions do not. The IISS solution also relieves a resource manager of the intensive labor that maintaining a role database and identity list demands.

The IISS framework provides an intermediary business rule language for RFID networks. Resource managers in RFID networks need such a language in order to allow them to describe their complex business rules. Each enterprise has the flexibility to grant access permission to set of parties based on inferences on their business rules. The IISS rule language allows resource managers to perform access control dynamically instead of searching static identity lists. The language also employs a policy structure which allows the enterprise to publish only an abstract interface composed of the necessary input parameters for satisfying the business rule instead of publishing the entire business criteria for EPC-related authorization.

The IISS framework employs ontologies that include not just role hierarchies, but any properties and constraints expressed in an RDF-based language, including elements of description logic and declarative rules. For example, a rule could specify that any party in an EPC associated supply chain is allowed to access the purchase order of the EPC-bearing product. In this way, rights can be assigned dynamically without creating new roles for the specific parties.

The IISS ontologies provide the vocabulary for describing entities and business rules. Each ontology models concepts such as provenance, EPCs, queries, and policies as discussed in Chapter 6. Using these ontologies, parties can express complex rules within business contexts. Because of the shared grammatical and ontological structure of the policy language, business rules are exchangeable between domains, thus understandable to the varied set of enterprises participating in the supply chain.

To describe an access control rule for a resource in the IISS framework, two policy entries are required: a *requirement policy* and a *verification policy.* These two policies are defined independently of each other. The *requirement policy* describes what kind of proofs a querying party needs to collect to validate a query on a particular

EPC-related resource. The *verification policy* is an inference rule which describes what actions a relying party should take given a specified set of antecedents. Only the requirement policies are accessible to querying parties, acting as an abstract protocol interface to the policy collection. The *verification policies* are kept confidential by relying parties. In this way, enterprises are able to publish the minimal amount of information necessary about their access control policies in order for users to receive access privileges. This strategy is necessary because access control policies in themselves may contain sensitive, or confidential, business information that enterprises do not want to or can not publish. For example, enterprises should not publish EPC-related provenance challenges.

Once a querying party provides proofs to the relying party, a relying party is able to do forward chaining[63] inference based on relevant verification policies. Because each verification policy is itself an inference rule, the resulting information generated by satisfying the inference rule may be applied to future verification policies in addition to the information provided initially by the querying party.

The IISS framework utilizes Notation3(N3) language to represent the policies in the IISS framework. The N3 language has the required expressivity and is convenient to read and write. To demonstrate this, an example policy from a relying party is described here. How that policy would be translated into the N3 language is also shown. In order to access the a page about the purchase order related to EPC 010000A8900016F000169D, the querying party must prove that he has an OpenID authenticated by the *Government* and can pass the provenance challenge. The business relationship in this policy can be represented by the IISS ontologies as shown in Figure 7-1.

The policy above can be interpreted by a *Requirement Policy* and a *Verification Policy* as follows. The *Requirement Policy* shows that the querying party needs to provide identity from the *Governmnet*, the identity format should be OpenID, and the querying party needs to provide the EPC's provenance challenge to access the EPC-related purchase order. The *Verification Policy* shows that if the querying party can provide the same EPC-realted provenance as the one that the relying party knows,

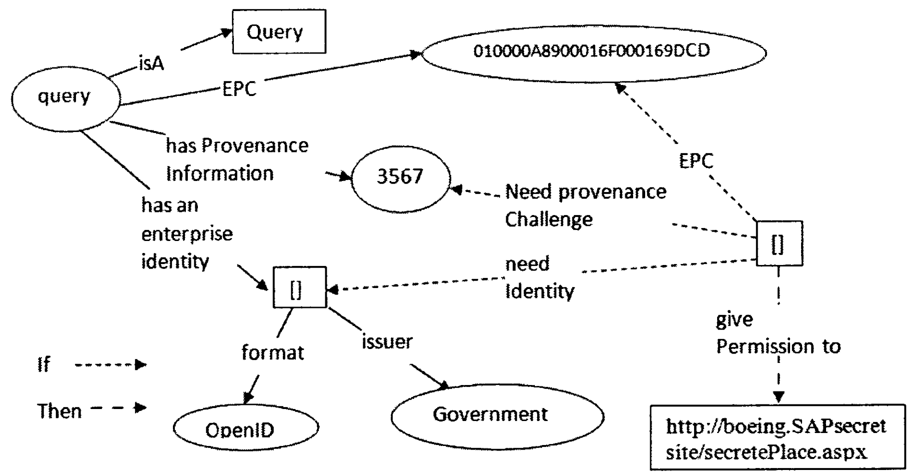

Figure 7-1: An example of how a policy is represented via the IISS ontologies

and an OpenID from the *Government* as http://main.goveropenid.com, the querying party is allowed to access the EPC-related purchase order. The logic behinds this policy can be represented as the following logical formula:

Requirement Policy:

$need\text{–}identity\text{–}from(PO, Government) \wedge$

$need\text{–}identity\text{–}format(PO, OpenID) \wedge$

$need\text{–}provenance\text{–}challenge(PO, ProvenanceChallenge)$

Verification Policy:

$\forall x, \forall pro, x \in Query, pro \in Provenance, \exists iden \in Identity$

$\{provennace(010000A8900016F000169D, pro) \wedge has\text{–}provennace\text{–}information(x,$

$pro) \wedge format(iden, OpenID) \wedge securityToken(iden, http://main.goveropenid$

$.com) \wedge issuer(iden, Government) \wedge has\text{–}enterprise\text{–}identity(x, iden)\}$

$\Rightarrow give\text{–}permission\text{–}to(x, PO)$

All the relationships and instances in the policy above can be represented by the IISS ontologies. To write the policy, the user may also take advantage of the RDF semantics and the logic defined by the World Wide Web Consortium at http://www.w3-.org/2000/10/swap/log. The user might require references to the EPC.n3 file, too. To enable all of the above, the user must include a certain set of prefix statements to their N3 document. The five necessary specifications which define the above vocabularies

```
<http://boeing.SAPsecretsite/secretePlace.aspx> a IISS:PO.
<http://boeing.SAPsecretsite/secretePlace.aspx>
                IISS:pneedidentityfrom IISS:Government;
                IISS:pneedprovenancechallenge [].
```

Figure 7-2: An example of a *requirement policy* represented by RDF statements

```
:SAPPO a IISS:PO.
 @forAll :x, :pro.
{
<EPC.n3>.log:semantics log:includes
{EPC:010000A8900016F000169D IISS:pprovenance :pro}.
<EPC.n3>.log:semantics log:includes {:pro a
IISS:ProvenanceChallenge}.
      :x a IISS:Query;
          IISS:phasProvenanceInformation :pro;
          IISS:phasanenterpriseidentity [
          IISS:pissuer IISS:Government;
          IISS:pformat IISS:OpenID;
          IISS:psecuritytoken <http://man.goveropenid.com> ];
        IISS:pepc EPC:010000A8900016F000169D.
    } => {:x IISS:pgivepermissionto
<http://boeing.SAPsecretsite/secretPlace.aspx>. } .
```

Figure 7-3: An example of a *verification policy* represented by RDF statements

and ontologies are:

```
@prefix rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix rdfs:  <http://www.w3.org/2000/01/rdf-schema#>.
@prefix log:  <http://www.w3.org/2000/10/swap/log#>.
@prefix IISS: <http://18.78.3.68/IISS/ontIISS.n3#>.
@prefix EPC: <EPC.n3#>.
```

The *requirement policy* of the example can be represented in RDF statements is shown in Figure 7-2.The *verification policy* of the example can be represented in RDF statemen is shown in Figure 7-3.

7.2 Reasoning Engine in the IISS framework

The IISS framework's policy reasoning engine accepts a query about a resource in an RFID network as a *Query* instance. It collects relevant information from the IISS framework for fielding the query, and replies to the querying party in regards to access rights to the data. It is used by relying parties to controlling access to resources based on the policy rules described in the previous section. As is the case when any logical statement must be proved true, care must be taken to precisely define the concepts and relationships of the kinds of information available in an RFID network. The IISS framework uses ontologies for this purpose: defining objects, relationships among objects, and inference rules. Since the ontologies are shared, new rules can be developed without preventing inference over existing data and rules. In other words, the ontologies allow the universe of policy rules to be dynamic, as opposed to a set of static rules which cannot deal with new entities in the dynamic networks.

In the IISS framework, the reasoning engine is implemented with software written in C#, Python, Cwm[23], and N3[24] rules. The IISS framework relies on N3 semantics and Cwm functionality to integrate and reason over the IISS access policies. N3 rules allow policies to access the Web at runtime and dynamically check the contents and inference rules of documents over the internet. Cwm contains a number of built-in functions supporting cryptography, string manipulation, and math formulas, all of which have proved useful for specifying policies.

The IISS ontologies are defined in RDF-S.The core of the reasoning engine is made up of an abstract model, as a set of definitions and inference rules, written in N3 and stored in the engine.n3 file. Cwm[23], as a general purpose data processor, is used as the reasoning engine for the N3 rule language(one of the features of the processor). The reasoning engine accepts a *Query* instance as input and retrieves the logic representing the relationship between the requested resource and relevant policies. The input can be serialized either in RDF/XML[21] or N3[22], both being understandable to Cwm.

Upon receiving some input, the reasoning engine parses it to retrieve the attributes provided by the querying party and the information regarding the requested resource. When it has retrieved the necessary data, the engine processes the files which are defined in N3[22]. The results of the reasoning indicate what actions the relying party should take, and the outputs is serialized in RDF/XML or N3. The relying parties can then use these results to decide which resources to which the query parties should be granted access.

One requirement of the IISS framework regarding policy definitions is that relying parties must maintain a mapping of which policies apply to the resources they control. In the implementation, this is done by parsing the EPC.n3 file which contains this mapping. In order to connect an EPC code with its *verification policy*, the *policy* property links an EPC to the namespace where the EPC-related *verification policy* file exists. The verification policy file then contains a definition of the list of all resources related with a particular EPC.

In the case where the query and associated data satisfy the antecedents of an inference rules in the EPC's verification policy file, the reasoning engine will always return the consequences of the satisfied inference rules. These results describe the action the relying party will take regarding the query or the querying party. One such action would be giving permission to access the requested resource.

Formally, the reasoning engine can be modeled as follows.

$\forall que, \forall policy, \forall pro, \forall epc, \forall res,$

$\{epc(que, epc) \wedge proof(que, pro) \wedge policy(epc, policy);$

$F = (policy \vee pro \vee que);$

$F \Rightarrow give\text{–}permission\text{–}to(que, res)\} \Rightarrow give\text{–}permission\text{–}to(que, res)$

When the formalism above is translate into N3 rules, it uses the log:conjunction function defined in the namespace http://www.w3.org/2000/10/swap/log# to perform the merger of the query, proof and *verification policy* files. The log:conclusion function defined in the same namespace is used to create the implication that determines whether the merged formulas from the *Query* instance and *Proof* instance include the same relationship as the antecedents defined by the *verification policy.* When the necessary prefixes are included that provide access to function namespaces,

```
@forAll :que, :epc,:policy, :pro, :A, :F, :G, :res, :iden.
{ :que IISS:pepc :epc.
 :que IISS:pproof :pro.
 <EPC.n3>.log:semantics log:includes {:epc IISS:ppolicy :policy}.
(:policy.log:semantics
 :pro.log:semantics
 :que.log:semantics) log:conjunction :F.
 :F log:conclusion :G.
 :G log:includes { :que IISS:pgivepermissionto :res}
 } =>{ :que IISS:pgivepermissionto :res}.
```

Figure 7-4: Reasoning engine represented in RDF

the reasoning engine can be defined as a set of N3 rules as in Figure 7-4

A reasoning engine can be accessed by a relying party or a resource manager through the Cwm command-line interface or in Python via its Application Program Interface (API).

7.3 Advantages of adoping the Semantic Web Framework in RFID networks

There are three important advantages of applying the semantic web framework in RFID networks. First, using the Semantic Web, parties can retrieve data stored at other domains by referring to the URIs of the data, an essential tactic for a dynamic policy system which requires uniform access across multiple platforms such as the widely distributed, heterogeneous network that comprises a typical RFID network. Instead of exchanging policies and information among different parties, each party stores the information in their respective repositories and only transmits the URI to other parties when requested. Moreover, because access policies are enforced individually, enterprises are free to define access policies based on sensitive information without unintentionally leaking it to competitors: URIs to the sensitive information are simply never released to the public.

Second, the semantic web provides a means through which common concepts or objects in the form of ontologies can be shared among all agents in an RFID network. The shared ontologies guarantee interoperability among agents because all the data transmitted in the IISS framework is defined based on these public, Web-accessible ontologies. Further, a party that can process RDF files is able to understand data that is defined based on the IISS ontologies, such as *Proof* and *Query* objects, from other parties in the network, even if no relation existed between them before. Providing vocabularies targeted at RFID business relationships increases unity among network entities with respect to the format of access control in RFID networks. Agents perform access control dynamically and flexibly by defining their own business rules without having to sacrifice ultimate control over authentication.

Third, the Semantic Web framework already has a rich set of logic vocabularies and reasoning tools available to applications. This set continues to evolve, ensuring that RFID networks using the IISS framework can take advantage of the latest advances and trends.

7.4 Policy Input Proxy

In order to provide an easy way for users of the IISS framework, many of whom will probably be unfamiliar with the RDF syntax-to generate business rules complying with the IISS ontologies, the IISS framework includes a policy input writer application. This small application includes a policy input user interface and a policy writer. Resource managers in RFID networks can use it to generate EPC-related policies in the form of an RDF file and store these policies in the *Requirement Policy Store* and the *Verification Policy Store* respectively. The UI provides the an English-like representation of the business methods, entities, conditions and actions. When the user has entered the required information, the UI displays the translated RDF code in the policy boxes. Figure 7-5 shows the user interface along with the two policies generated by the input. After users check some properties of their polices and click

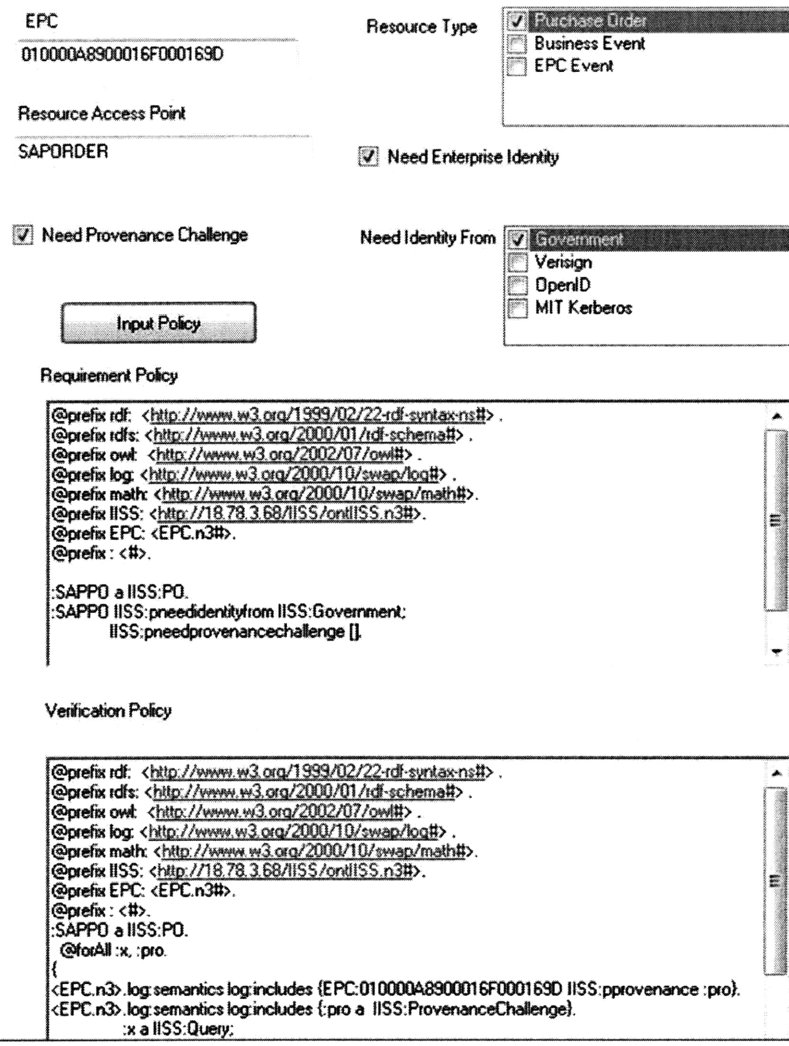

Figure 7-5: Policy input proxy

the Input Plicy botton, the *Requirement Policies* and the *Verification Policies* are generated automatically. The application can be accessed through a web browser or by installing the IISS framework tool kits.

Chapter 8

Application of the IISS Framework in Supply Chains

In this chapter, I take a supply chain as a use case to demonstrate how IISS can be used in RFID applications. I demonstrate the application of the IISS framework to the problem of authorization and authentication on EPC related queries in supply chains. I explore how to use IISS to handle queries from different domains and how to generate and publish business rules which are interoperable between multi-domain agents in supply chains. I argue that a centralized registry model should be avoided by analyzing latency results obtained by simulating a registry model in supply chain EPC networks. Then I show how to avoid a centralized registry model in the data transmission layer by introducing IISS in supply chains.

8.1 Work flow in IISS

In this section, I illustrate the work flow in the IISS framework when it is applied to a supply chain use case to show how to use the IISS framework in RFID networks.

Figure 8-1 illustrates the data flow in the IISS framework when it is applied to supply chains. When a manufacturer starts shipping an item, the manufacturer's provenance server generates a provenance challenge to identify the item's trace. At the same time, his provenance writer puts the provenance challenge as the EPC's

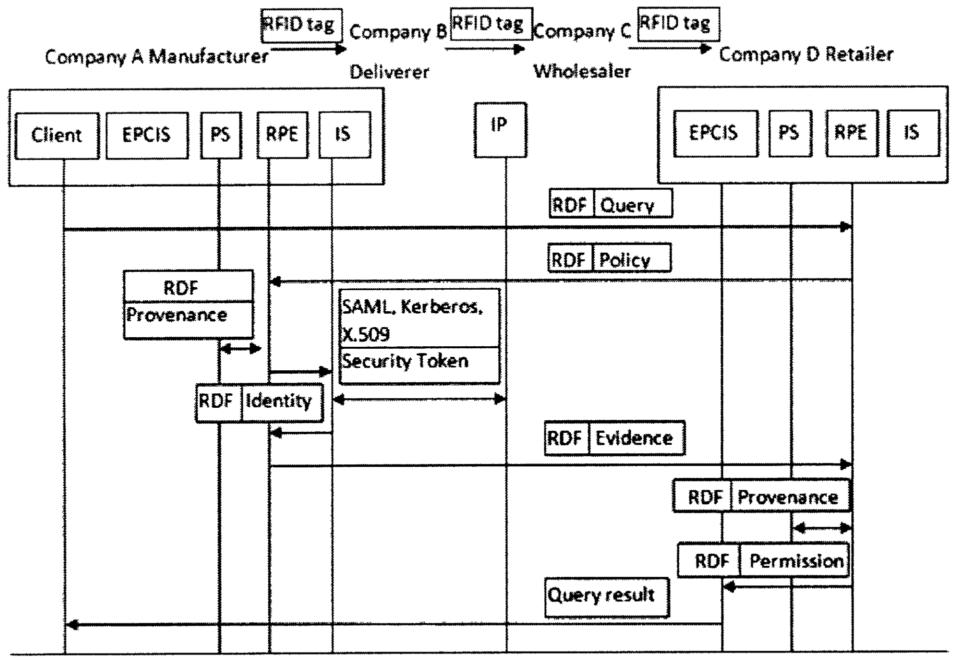

Figure 8-1: Data flow in the IISS framework in a supply chain

provenance value in his EPC.n3 file. The province writer also puts an entry in the EPC.n3 file to associate the EPC with its *verification policy* file. When the item passes through a supply chain, the EPC related provenance challenge will be transmitted confidentially through *provenance servers* and stored in the EPC store in each agent's domain through *provenance writers*. At the same time, *provenance writers* write the entry into the EPC.n3 file, which associates the EPC with the *verification policy* file name.

As shown in the figure, a manufacturer A initiates a query to retrieve a purchase order about an item J with the EPC 010000A8900016F000169D from a retailer D. However, D does not have a direct business relationship with A. Because J's purchase order is stored in a confidential web page http://boeing.SAPsecretsite/secretePlace.aspx, access permission to J's purchase order can only be given to J's manufacturer if he is one of the government's contractors. The manufacturer needs to present the X.509 certificate issued by the government. At the same time, the manufacturer has to prove that he knows J's provenance challenge. The policy about this purchase order has been input by D's resource manager through its policy input proxy. The *requirement policy* and the *verification policy* about this purchase order are expressed using the business rule vocabularies defined in the IISS framework and stored in the 010000A8900016F000169D's *requirement policy* file(http://18.78.3.68/IISS/010000A-8900016F000169DCDrequirementPolicy.n3) and 010000A8900016F000169D's *verification policy* file (c://010000A8900016F000169DverificationPolicy.n3) in D's domain.

On receiving the query from A, D's *Rule and Policy Engine* sends A the URI (http://18.78.3.68/IISS/010000A8900016F000169DCDrequirementPolicy.n3), through which A can retrieve the *requirement policies* on the EPC related resources. D also sends A the namespace of its EPC file (EPC.n3)

When A receives the package from D, A's *Query Generator* creates a query instance for this query referred by a URI (http://18.78.3.60/IISS/query1.n3). In the instance, J's EPC is associated to this query. Also, a URI (http://18.78.3.60/IISS/-proof1.n3) is assigned to this query's proof file and recorded as the value of the *proof* property in the *Query* instance. Then the *Proof Collector* in A retrieves the *requirement policy* file from D and extracts the *requirement policy* for the purchase order by parsing the file. According to the description of the purchase order's *requirement policy*, the *Proof Collector* in A generates the proof file (http://18.78.3.60/IISS/-proof1.n3) to support A's query. In this case, the *Proof Collector* retrieves an X.509 certificate from the government if A is one of its contractors through D's identity selector. The X.509 certificate will be transformed to an instance of the *Identity* class in RDF format through the adaptor. At the same time, A's *Proof Collector* retrieves the provenance challenge about J's EPC from A's EPC store and adds this evidence to the proof file as the value of *has Provenance Information* property of the *Query* instance. Then A's *Query Generator* sends the *Query*'s URI (http://18.78.3.60/IISS/query1.n3) to D's *Reasoning Engine.* After the query is finished, the related proof file and query file are deleted just for security reasons.

Once receiving the *Query*'s URI from A, D's *Reasoning Engine* will reason over the *Query*, the *proof* and the EPC-010000A8900016F000169DCD's *verification policy.* Because D has a trust relationship with the government and has the public key of the government's certificate authority, D can verify the X.509 certificate issued by the government. After reasoning, D's *Reasoning Engine* returns the actions in form of RDF statements that D can perform. According to the result from reasoning, D can either give the permission to A to access the purchase order or not.

8.2 Centralized Registry in Supply Chain

The open registry approach in [61] proposed a centralized registry server to receive reports from all the parties in RFID networks about who has the item with EPC X and who can share this entry of data. With the centralized registry, a party can discover where the item associated with EPC X is and make a further query with the party who currently has the item. This is the only proposal about tracking RFID bearing items in RFID networks so far. However, the centralized registry has some disadvantages as described in Chapter 3. Besides the logistic challenges, security challenges and trust challenges the registry model introduces, the registry model is infeasible in supply chains due to the computational workload of the centralized registry.

The AutoID lab at MIT developed a supply chain EPC simulator in order to obtain the quantity of the message flow in future EPC network infrastructures. It provides a good platform for testing and analyzing performance and scalabilities of various software architectures and models applied to EPC networks[84]. The AutoID lab at MIT simulates the supply chain networks with the registry model. In the simulation, authentication based on querying parties' identities is considered. Digital signatures are applied to authenticate the query from the next tier facility as shown in Figure 8-2.

In order to transmit data securely in a supply chain, security techniques such as digital certificates, digital signatures and encryption are used to guarantee confidentiality and integrity. In the registry model, the encryption layer between the supply chain partners and the decryption layer in the centralized registry will guarantee the confidentiality of data transmission. The registry and a supply chain partner share a symmetric key for encryption and decryption. The Public Key Infrastructure is used to guarantee the data transmission between the facilities in different tiers. A

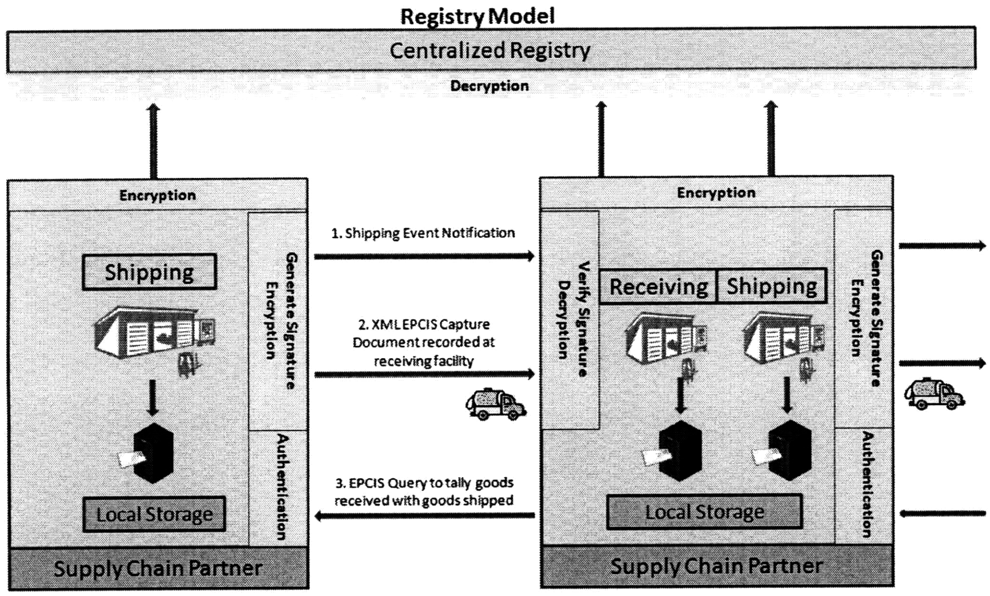

Figure 8-2: Security scheme in open registry approach

centralized certificate authority is required in this scheme. The central registry can have its own certificate authority or using some business certificate authorities, such as VeriSign.

As shown in Figure 8-3, the centralized authority issues an x.509 certificate for each facility, which can be trusted by all the other facilities. Before a facility starts sending EPC events to a facility in the next tier, an X.509 certificate with the facility's public key is transmitted. Using its private key, the facility encrypts the message, and generates a signature of the message and sends it to a facility in the next tier. When the facility in the next tier receives the message, it decrypts the message, and generates a digest of the decrypted message with the public key for the facility in the upper tier. After comparing the digest with the signature received, the facility in the next tier can verify and authenticate the received message. In this way, the message transmitted between facilities can be sent with confidentiality and integrity.

In the evaluation of the centralized registry model, the supply chain topology is assumed as in Table 8.1. These values are based on the pharmaceutics industry in US. Based on these assumptions, the AutoID lab at MIT simulation estimates that the message decryption time to process all the messages generated in EPC networks

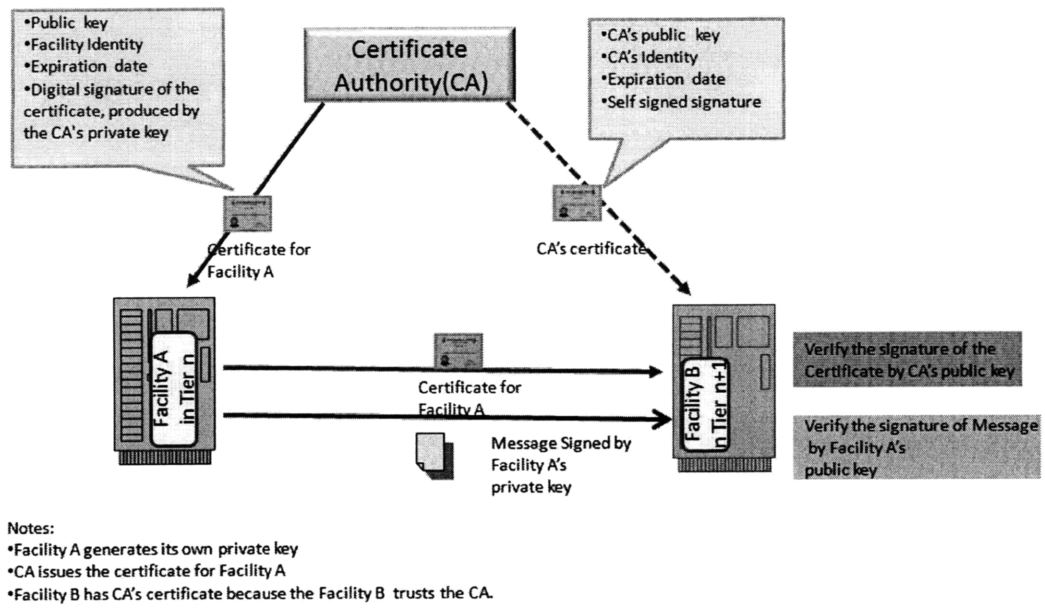

Figure 8-3: PKI in centralized registry model

Component	*Value*
Number of Manufacturer Facilities	400
Number of tiers in the chain	10
Number of Distribution Centers	200 (among 8 tiers)
Number of Retail Outlets	800
Number of Orders processed per day	300,000
Number of supply chain partners	10
Size of EPCIS message (100 items)	100Kb
Size of Purchase Order Message (100 items)	50Kb
Size of E-Pedigree Message (100 items at Manufacturer Tier)	119Kb

Table 8.1: Assumption of a supply chain topology

per day at the registry is 278769.60 seconds which is longer than one full day. More machines in a parallel computing environment are needed to accomplish the tasks of the registry service[25]. If there are more distribution centers involved in supply chains, the work load of the registry will be even heavier. Considering supply chains which cover the whole industry globally, the workload will be much more than the limit a centralized register could realistically handle.

8.3 How to avoid centralized server

Once the IISS framework is implemented in supply chains, a centralized registry server is not the only way to track items. Here is the workflow describing how an enterprise can find a particular EPC bearing product in supply chains without a centralized registry.

First, a querying party has a list of addresses of distribution centers in the supply chain network, or it can retrieve the list from some information center. Then the querying party broadcasts its query to all the parties in the list. When the parties who may have transacted the RFID tag receive the query from the querying party, they either respond to the query with the URIs of their requirement policies of the EPC or ignore the query if they do not want to publish their data to anyone. After that, it is querying party's responsibility to collect evidence to support its query. The querying party generates different proof files for different parties due to various requirements from those parties. The querying party creates one *Query* instance for each party and sends the query instances' URIs to the relying parties respectively. Once the relying parties validate the *Query* instances, they return the EPC related information to the querying party. With the information, the querying party can do inference and extract the track of the RFID bearing object.

8.4 Benefits of introducing the IISS framework to supply chains

The discussion above illustrates the benefits of adopting the IISS framework in supply chains as follows. First, all the partners in supply chains do not need to trust one Identity Authority. As long as querying parties and relying parties have a common Identity Authority, authentication can be preformed. If they do not share any identity authorities, the provenance information in IISS can be used as proof. Second, instead of associating EPC events with parties by statically enumerating parties' credentials, relying parties can state declarative policies by using the policy language in the IISS framework. Third, once the IISS framework can be implemented in supply chains, a centralized registry server is not necessary for tracking purposes. If a company wants to locate an item is or know the item's track, it can send out a batched query to all the possible companies. If a company has the EPC information and verifies that the querying company is qualified for notification, the company can provide the querying company with the time and place the RFID tag of the item was read.

Chapter 9

IISS Performance

This chapter introduces a supply chain EPC simulator as the testing platform for the IISS framework. How to integrate the simulator with the IISS framework is demonstrated. Then how efficiently the IISS framework handles authorizition and authentication of EPC-related queries in supply chains are shown. This chapter is finished by discussing factors that affect the IISS framework's scalability and provide solutions for improving its performance.

9.1 Supply chain EPC simulator

The AutoID lab at MIT developed a supply chain EPC simulator in order to obtain quantified data about message flow in future EPC network infrastructures. It provides a good platform for testing and analyzing the performance and scalability of various software architectures and models applied to EPC networks[84].

As shown in Figure 9-1, the simulator is composed of multiple supply chain tiers which representing actual components of a supply chain. The supply chain starts with manufacturers, moves through multiple levels of transportation and storage, and culminates at retailers, each level being represented by a tier. Each tier may have an arbitrary number of facilities. Each facility is modeled as a state machine running in its own thread of execution. There are two special facilities in the simulator that are not in the physical supply chain. One is the *Source*, and the other is the *Sink*.

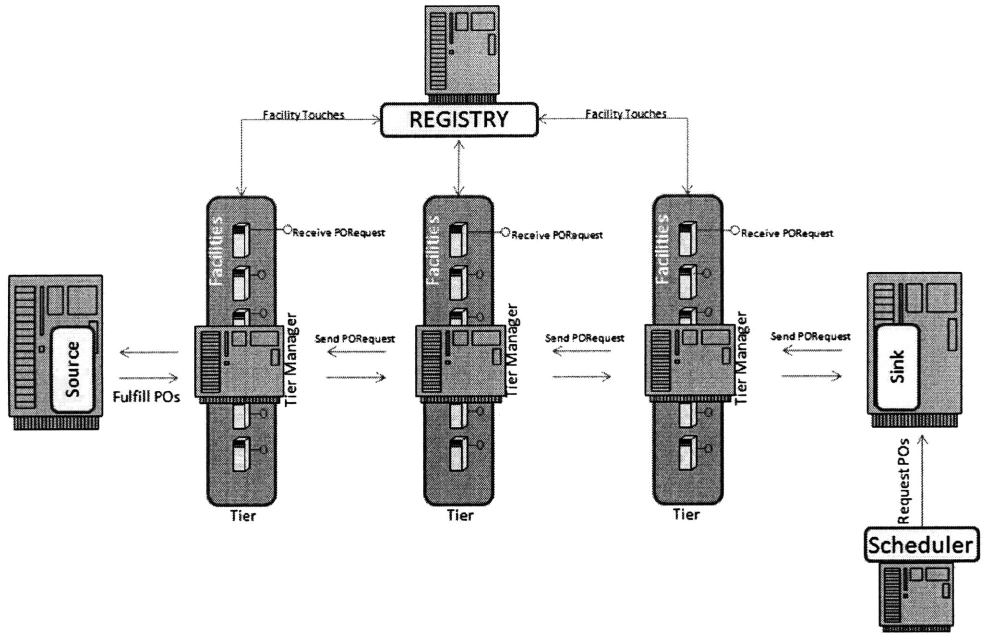

Figure 9-1: Supply chain EPC simulator[84]

They are named from the perspective of the production and consumption of physical goods. The *Source* plays the role of the universal producer of goods. The *Sink* acts as the universal purchaser of goods. Both the *Source* and the *Sink* are used to control the flow of goods through the system. The *Scheduler* is responsible for generating purchase order messages and injecting them into the simulated supply chain. The purchase order messages will trigger facilities to fulfill the purchase orders, and EPC events are generated during this process.

The simulator also has two components which exist outside of the supply chain. The *Scheduler* also provides a number of control parameters that determine the rate at which purchase orders are injected by the retailers in response to goods being sold, and the volume of the messages that are transacted by the simulated supply chain. The *Registry* is designed to simulate the centralized registry service in EPC networks. The registry handles two kinds of message traffic, namely "I've touched EPC X" messages and "Who has seen EPC X?" query messages. The first kind of message traffic results from shipments received and from shipments shipped. The second kind of message traffic results from queries.

The system is designed to run on multiple machines in a massively parallel threaded environment. The simulator is built upon the Microsoft Coordination and Concurrency Runtime[9] which provides support for multi-threaded programming.

9.2 Integration of IISS with the supply chain EPC simulator

In order to test the IISS framework's performance, the supply chain EPC simulator is used to simulate the basic message flow and goods flow in EPC networks. As shown in Figure 9-2, three modules in IISS- *Provenance Server, Rule and Policy Engine* and *Identity Selector* - are integrated with each facility in the simulator. Identity providers which can provide multiple identity tokens to facilities in the simulator are set up. The *Provenance Server* of a manufacturer facility generates provenance challenges for each EPC that the manufacturer facility generates. The provenance challenges are transmitted through the EPC-related supply chain which is simulated via fulfilling purchase orders in the simulator. Consistency between the trace of a provenance challenge and its related EPC is guaranteed in the simulator. In this way, the simulator provides both EPC events data and provenance data as the foundation for testing the IISS framework.

9.3 IISS performance

The IISS's performance are tested in a Dell PC machine with the following configuration.

Processor: Intel(R) Xeon(R) CPU 5130 @ 2.00GHz 1.99GHz.

Memory (RAM): 3069 MB, System type: 32-bit Operation System.

Network speed: 100.0Mbps.

First test IISS's performance is tested under the circumstances that an EPC.n3 has 100 EPC entries, 200 EPC entries, 300 EPC entries, 400 EPC entries, 500 EPC entries, and 600 EPC entries respectively. The results are shown in Figure 9-3. The

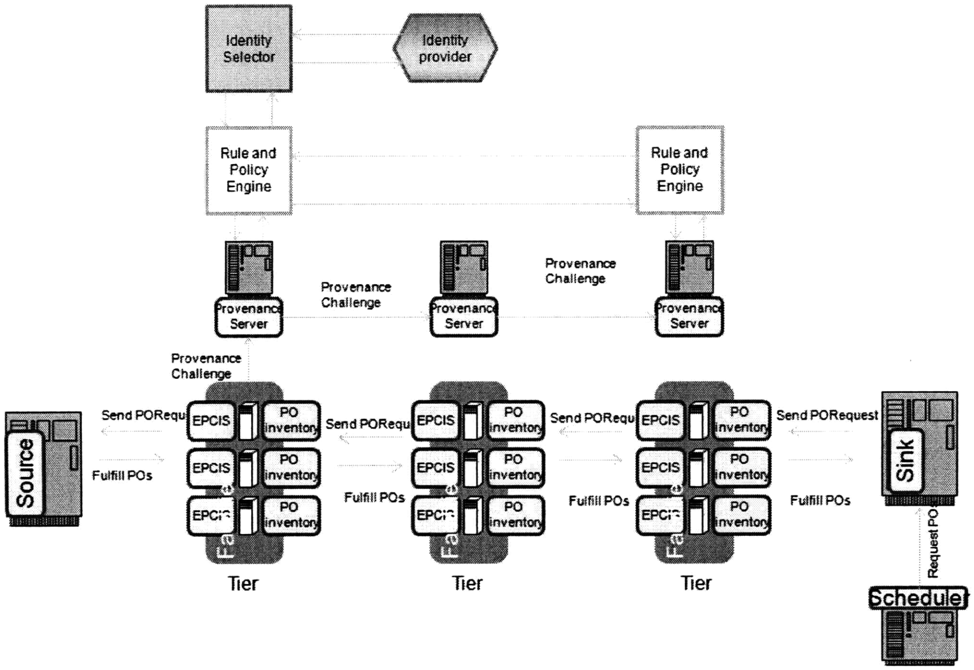

Figure 9-2: Integration of IISS with supply chain EPC simulator

x-axis represents the number of queries that are executed in the IISS framework. The y-axis represents the number of time (in milliseconds) that it takes the relying party to authenticate and authorize queries in the IISS framework. Each set of data represents the test result of the IISS's security overheads on an EPC.n3 file with different number of EPC entries.

These results lead to a couple of conclusions. First, the time used to perform authentication and authorization over queries in the IISS framework increases linearly as the number of queries increases. Second, the validation time per query in the IISS framework is affected by the number of EPC entries stored in the EPC.n3 file. The bigger the size of the EPC.n3 is, the greater the slope of the linear function between time and query number is. This means an IISS query on a bigger size EPC.n3 file will have a longer security lookup overhead.

Based on the second conclusion from the first test, another experiment is designed to test how IISS efficiency is affected by the size of an EPC.n3 file. In this test, fifteen EPC.n3 files EPC entry sizes of 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 1100, 1500, 2000, 2500 and 3000 respectively are generated. After measuring the security

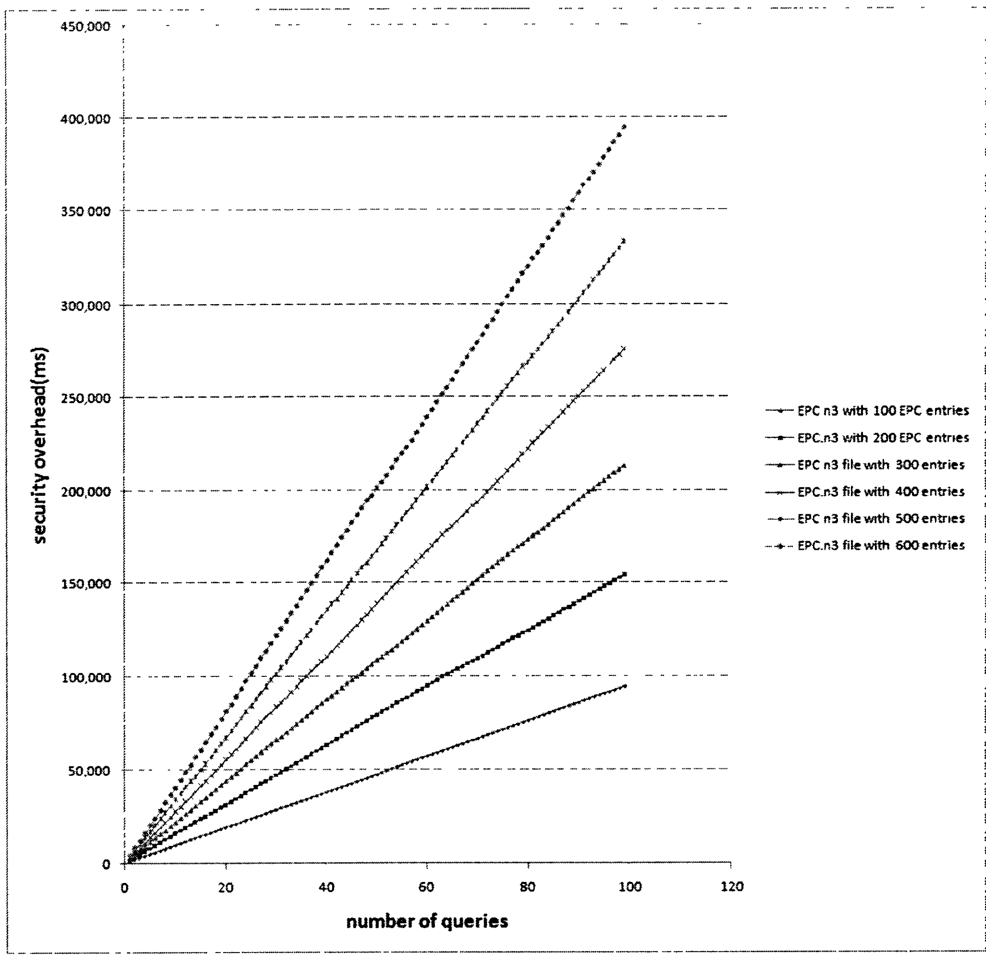

Figure 9-3: Latency vs the number of queries

overhead of 1000 queries over this set of files, the average time cost of validation time per query for each file is calculated. The test results from this experiment are shown in Figure 9-4.

y-axis represents the time(in ms) required to validate one query. The x-axis represents the number of EPC entries in the EPC.n3 file. The result shows that validation time using the IISS protocol increases linearly-from 955.9 ms to 19778 ms-as the number of EPC entries per EPC.n3 increases. The reason behind this is that, according to the IISS protocol, a querying party needs to search the EPC.n3 file to retrieve the provenance information of targeted EPCs. Bigger the EPC.n3 file result in longer seeking time. Moreover, the reasoning tool(Cwm) which the IISS framework uses does forward chaining over the merged formulas from query.n3, proof.n3,

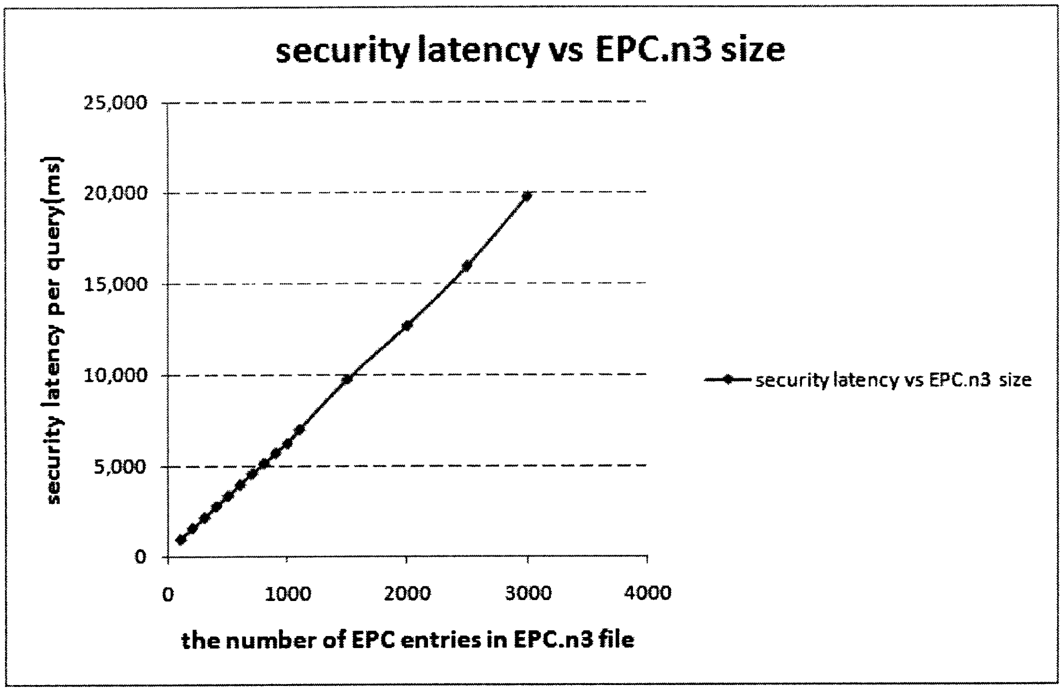

Figure 9-4: Latency vs EPC.n3 size

requirementpolicy.n3 and EPC.n3. A large EPC.n3 file will introduce a large number of formulas to the facts base of the reasoning engine. Therefore, it will take a longer time for the reasoning engine to search through its facts base for the instances that satisfy a policy rule's antecedents if the EPC.n3 file is big. If the IISS framework uses one EPC.n3 file to store all the EPC entries, the performance of IISS is far from ideal.

In order to find the real bottleneck which was causing the linear growth behavior relating to the number of EPC entries in the IISS protocol, The third experiment is designed. In this experiment, the entire latency introduced by the IISS protocol was broken down into three parts: the latency from retrieving requirement policies, the latency from collecting evidences and generating textitQuery instances, and the latency from reasoning over verification policies. I tested 100 queries over a set of EPC.n3 files with 500, 1000, 1500, 2000, 2500 and 3000 EPC entries respectively. As Table 9.1 shows, latency caused by reasoning over verification policies causes more than 98% of the entire latency of the IISS protocol. This clearly points out that the automated reasoning is the major bottleneck of the IISS protocol.

Entries in EPC.n3	500	1000	1500	2000	2500	3000
Latency from getting requirement policy(ms)	15.62	15.10	13.17	9.68	28.71	13.93
Latency from generating query(ms)	41.96	67.37	122.06	157.79	210.42	227.86
latency from reasoning(ms)	3292.17	6151.41	9153.37	12646.75	15625.5	19319.17
total latency(ms)	3353.62	6237.10	9291.82	12818.44	15870.14	19563.79
latency from reasoning/total latency	0.9816	0.9862	0.9851	0.9866	0.9845	0.9875

Table 9.1: Breakdown of the latency in the IISS framework

As shown from Figure 9-5, both the latencies from generating queries and from reasoning are increase linearly as the EPC.n3 file gets larger. However, the latency from getting requirement policies varies independently of the size of the EPC.n3 file. The reason for this phenomenon is that both the process of generating queries and reasoning must look up entries in the EPC.n3 file, and retrieving requirement policies does not require any reading from the EPC.n3 file.

All of these experiments really point out that one key characteristic, the size of the EPC.n3 file, is the major contributor to slowdown of the IISS framework's performance

9.4 Solutions to improve the IISS protocol's performance

In order to improve the IISS protocol's performance, I provide two solutions.

The first one is to partition the EPC.n3 file and index these partitions. When the reasoning engine and the provenance writer in the IISS framework process the EPC.n3 file, they are able to use the index to find the partition which contains the entry they need.

The structure of the EPC store is shown in Figure 9-6. Instead of having one EPC.n3 file, the EPC store contains multiple EPCn.n3 files. The index algorithm

Partition size of EPC.n3	1	10	50	100	200	300	500	750	1000
latency from retrieving requirement policy(ms)	11.8	9.4	14.1	16.3	14.7	12.8	13.2	16.3	13.6
latency from generating query (ms)	5.4	5.7	12.4	11.3	21.5	28.4	37.4	51.2	72.5
latency from reasoning (ms)	377.2	462.3	690.5	981.8	1588.6	2214.9	4168.6	5032.9	7202.7
total latency (ms)	398.6	480.1	722.2	1011.5	1627.3	2266.3	4221.4	5103.9	7292.8

Table 9.2: Latency of IISS with index

used in testing is described in the equation:

$$n = \lfloor \frac{serial\ number}{partition\ size} \rfloor \tag{9.1}$$

100 queries over 3000 EPC entries in different EPC.n3 partition senarios is tested. The average latency per query of the IISS protocol is shown per partition size breakdown in Table 9.2.

To further show the effect of the partition sizes, I plot this data in a graph in Figure 9-7. The results show that the latency of the IISS protocol drops sharply from 7292.82 ms per query to 398 .62 ms per query when the partition size of EPC.n3 files decrease from 1000 EPC entries per file to 1 EPC entry per file. In comparison with the IISS performance without any partitions-19563.79 ms per query, 398.62 ms per query with 3000 partitions indicates a major improvement in IISS efficiency. A latency below a half second would make IISS feasible when applied to supply chains in the industry.

According to these test results, the optimal design of the EPC store is to record one EPC entry in each EPC.n3 file and use an index algorithm to match a query with its target EPC's EPC.n3 file.

The second solution proposed for optimizing the IISS framework's performance is by generating policies for batched EPC queries. Instead of linking a *Query* instance and a policy to a particular EPC, parties can link them to a batch of EPCs with common features, such as the EPCs of the same batch of products from one manufacturer. In this way, a relying party does not have to validate queries individually under the circumstances that the queries have the same security properties and the target EPCs have the same access policies in a relying party. Since the improvement of the IISS performance in this way is obvious and hard to measure quantitively, I do not provide testing results for this solution here, leaving the implementation and measurement as future work.

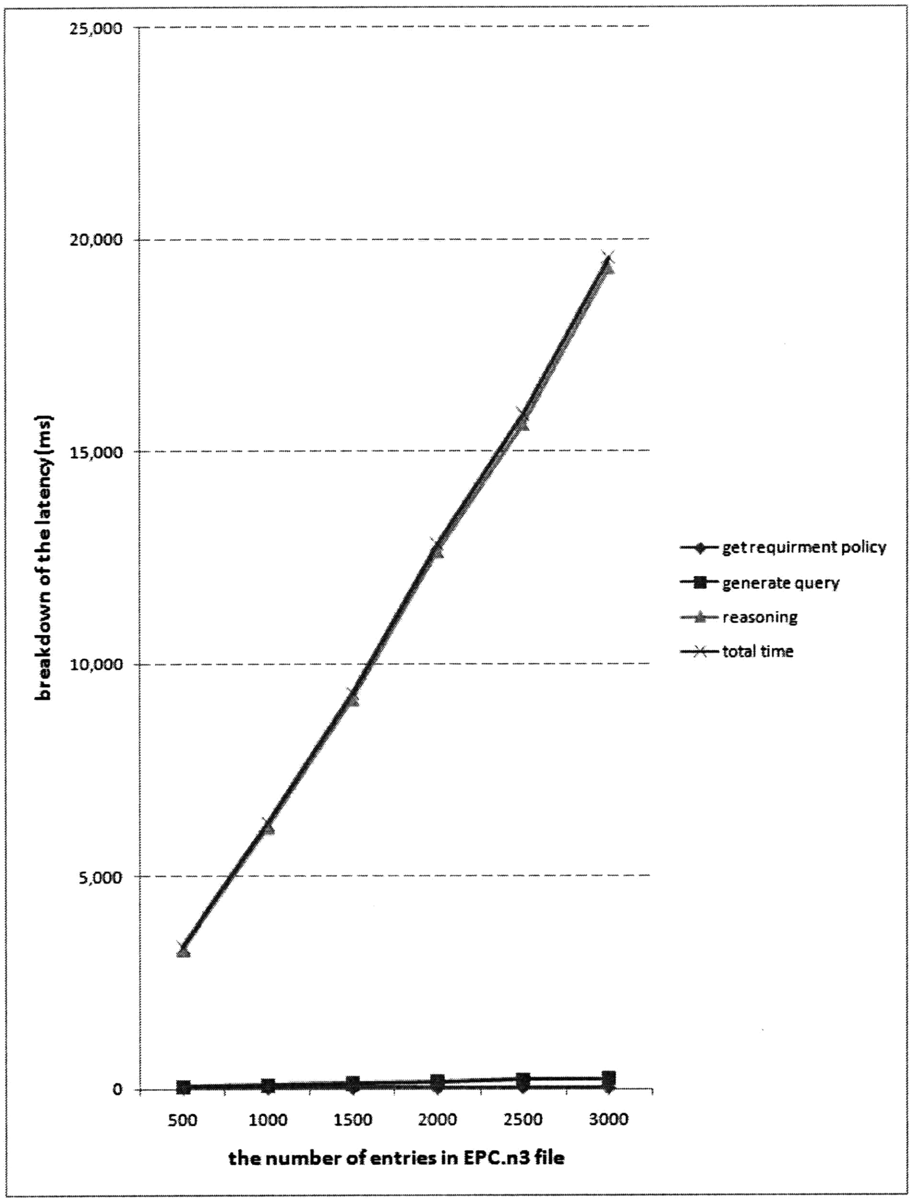

Figure 9-5: Breakdown of the latency in the IISS framework

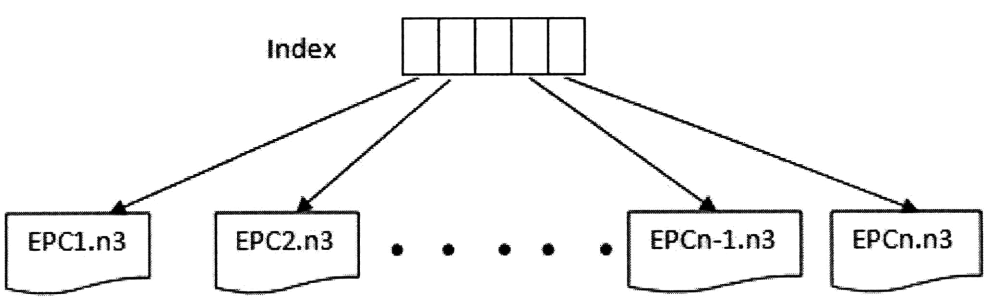

Figure 9-6: Index in *EPC store*

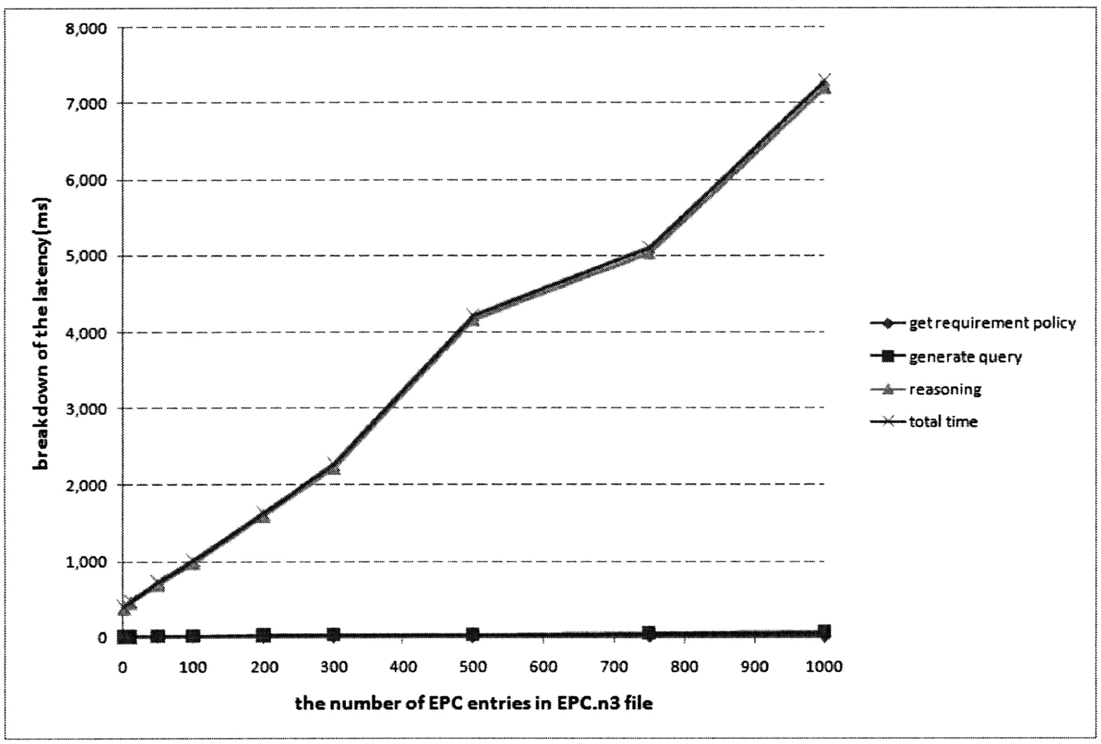

Figure 9-7: Latency of IISS with index

Chapter 10

Concluding Remark and Future Work

Throughout the thesis, how the IISS protocol is designed, deployed, and implemented has been described. A specific application of IISS in supply chains in order to demonstrate the feasibility of IISS has also been shown. In this chapter, the contributions of the IISS framework is summarized. Some remarks on the limitations of the IISS protocol and suggestions for future work are provided. In the end, important issues that need to be addressed in any practical implementation of our data model are discussed.

10.1 Contributions

The thesis presented six major contributions.

First, a solution to security problems in the RFID networks on which multiple partners with different identity schemes can rely - the Interoperable Internet-Scale Security (IISS) Framework is developed. By employing this IISS protocol, parties in RFID networks do not need to give up their original security schema to adapt a unified technique or standard.

Second, IISS is shown to be the first security mechanism that performs authentication based on an aggregation of business context, enterprise information, and RFID tag information. Users take these three kinds of information related to RFID-bearing objects into consideration when they decide whether to give permission to access information to a query in RFID networks. It is very important to provide a mechanism to collect information over these three aspects.

Third, an ontology for describing entities and business rules in RFID networks is developed. The ontology models concepts such as provenance, EPC, query, and policy. Based on the ontology, business rule vocabularies and a business rule language that allow parties to express complex rules with business-specific context in RFID networks are defined. Moreover, business rules defined on a shared business language can be interchangeable and understandable by parties from different domains.

Fourth, it is shown that IISS provides a dynamic mechanism to regulate access control of EPC- related resources. This scheme is an improvement over role-based access control solutions. It gives an enterprise more flexibility when defining its resource access criteria. Moreover, it will save resource managers a significant amount of labor from maintaining the role database and identity list.

Fifth, it is shown how IISS introduces provenance challenges in order to solve the trust problem in RFID networks. Provenance information can be used as a proof to validate a party's query to distant members of the supply chain who may or may not know about existence of this particular querying party. The provenance challenges are very lightweight and, thus, do not add much traffic to RFID networks.

Sixth, an analysis of the infrastructure of a centralized registration model in supply chains is detailed. Based on testing data, it is proved that centralized registration is not a feasible solution in terms of the security transaction workload. Moreover, merely learning the identity of who has data about a specific EPC from the centralized registry can reveal confidential information. Therefore, discovery information faces similar authorization issues as event data itself. IISS provides a strategy that enables enterprises to trace particular RFID-bearing objects without using a centralized registration service. A broadcasting mechanism may be used by parties to retrieve an RFID-bearing object's historic trace if the IISS framework is employed in RFID networks.

10.2 Limitations of IISS and future work

Through the IISS ontologies provide vocabularies for describing business relationships, they are not complete enough to cover all the business relationships in RFID networks. Further input from the EPC community's is necessary in order to enrich the ontologies for describing various business contexts.

One of the greatest prospects of the Semantic Web is its ability to merge separate fragments into a single unified semantic network. This feature relies on parties in different domains sharing the same concepts. This property of the Semantic Web extends to the IISS framework. Therefore, the proliferation of the IISS framework depends on parties' active involvement in deploying this infrastructure.

IISS is the first research project to explore how Semantic Web technologies can help people improve EPC networks from the perspective of security. It proves that the Semantic Web is a good choice for RFID networks due to its decentralized nature. The IISS framework provides a guide for future research to improve application interoperability and information exchange in RFID networks using Semantic Web technologies. For instance, parties in RFID networks can use a shared *Price* ontology to describe the price of each piece of EPC-related information. This description would then be understandable to all parties involved in the RFID network. In this way, a data trading system can be introduced for parties to swap EPC-related information among all of the entities in RFID networks.

Appendix A

Code Defining the IISS Ontologies

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
   @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
   @prefix owl: <http://www.w3.org/2002/07/owl#>.
   @prefix log: <http://www.w3.org/2000/10/swap/log#>.
   @prefix math: <http://www.w3.org/2000/10/swap/math#>.
   @prefix : <#>.
   # classes
   :Identity a rdf:Class.
   :IssuerCredential a rdf:Class.
   :SecurityToken a rdf:Class.
   :SecurityTokenFormat a rdf:Class.
   :Query a rdf:Class.
   :Resource a rdf:Class.
   :PO a rdf:Class; rdfs:subClassOf :Resource.
   :Date a rdf:Class; rdfs:subClassOf :Resource.
   :BusinessEvent a rdf:Class; rdfs:subClassOf :Resource.
   :Provenance a rdf:Class.
   :EpedigreeSignature a rdf:Class; rdfs:subClassOf :Provenance.
   :ProvenanceChallenge a rdf:Class; rdfs:subClassOf :Provenance.
   :EPCEvent a rdf:Class.
```

```
:EPC a rdf:Class.
:ReadPointID a rdf:Class.
:Time a rdf:Class.
:BusinessStepID a rdf:Class.
:Policy a rdf:Class.
#properties
:pissuer a rdf:Property;
rdfs:domain :Identity;
rdfs:range :IssurerCredential.
:psecuritytoken a rdf:Property;
rdfs:domain :Identity;
rdfs:range :SecurityToken.
pformat a rdf:Property;
rdfs:domain :Identity;
rdfs:range :SecurityTokenFormat.
:phasanenterpriseidentity a rdf:Property;
rdfs:domain :Query;
rdfs:range :Identity.
:phasProvenanceInformation a rdf:Property;
rdfs:domain :Query;
rdfs:range :Provenance.
:pepc a rdf:Property;
rdfs:domain :Query;
rdfs:range :EPC.
:pwhat a rdf:Property;
rdfs:domain :EPCEvent;
rdfs:range :EPC.
:pwhere a rdf:Property;
rdfs:domain :EPC;
rdfs:range :ReadPointID.
```

```
:pwhen a rdf:Property;
rdfs:domain :EPC;
rdfs:range :Time.
:pwhy a rdf:Property;
rdfs:domain :EPC;
rdfs:range :BusinessStepID.
:ppolicy a rdf:Property;
rdfs:domain :EPCEvent;
rdfs:range :Policy.
:pgivepermissionto a rdf:Property;
rdfs:domain :Query;
rdfs:range :Resource.
:pkeyto a rdf:Property;
rdfs:domain :EPC;
rdfs:range :Resource.
:pneedidentityfrom a rdf:Property;
rdfs:domain Resource;
rdfs:range :IssuerCredential.
:pneedprovenancechallenge a rdf:Property;
rdfs:domain :Resource;
rdfs:range :ProvenanceChallenge.
:pprovenance a rdf:Property;
rdfs:domain :EPC;
rdfs:range :provenance
#instance
:OpenID a :SecurityTokenFormat.
:Government a :IssuerCredential.
```

Bibliography

[1] *http://rfid.idtechex.com/knowledgebase/en/breakdown.asp.*

[2] *http://www.idealliance.org/papers/dx_xmle04/papers/04-01-04/04-01-04.html.*

[3] *http://p3pbook.com/w3c/policy1.xml.*

[4] *http://www.w3.org/2000/10/swap/doc/formats.*

[5] *http://www.epcglobalinc.org/standards/.*

[6] *http://www.epcglobalinc.org/standards/ons/ons_1_0-standard-20051004.pdf.*

[7] *http://www.omg.org/cgi-bin/doc?ad/97-08-11.*

[8] *http://www.omg.org/mof/.*

[9] *http://forums.microsoft.com/MSDN/ShowForum.aspx?ForumID=1424&SiteID=1.*

[10] The Platform for Privacy Preferences 1.0 (P3P1.0) Specification. *http://www.w3.org/TR/P3P/*, April 2002.

[11] Enterprise Privacy Authorization Language. *http://www.w3.org/Submission/-2003/SUBM-EPAL-20031110/*, November 2003.

[12] 2007 Annual Study: U.S. Cost of a Data Breach, Ponemon Institute, LLC. *http://download.pgp.com/pdfs/Ponemon_COB-2007_US_071127_F.pdf*, 2007.

[13] EPC Information Services (EPCIS) Version 1.0.1 Specification. *http://www.epcglobalinc.org/standards/epcis/epcis_1_0_1-standard-20070921.pdf*, September 2007.

[14] Pedigree Ratified Standard, Version1.0. *http://www.epcglobalinc.org/standards/-pedigree/pedigree_1_0-standard-20070105.pdf*, January 2007.

[15] Public-Key Infrastructure (X.509) (pkix). *http://www.ietf.org/html.charters/pkix-charter.html*, April 2008.

[16] Smart Card Alliance. RFID Tags and Contactless Smart Card Technology: Comparing and Contrasting Applications and Capabilities. *http://www.hidcorp.com/documents/tagsVsSmartcards_wp_en.pdf.*

[17] S. Anderson, J. Bohren, T. Boubez, M. Chanliau, G. Della-Libera, B. Dixon, et al. Web Services Trust Language (WS-Trust). *Public draft release, Actional Corporation, BEA Systems, Computer Associates International, International Business Machines Corporation, Layer*, 7.

[18] B. Atkinson, G. Della-Libera, S. Hada, M. Hondo, P. Hallam-Baker, J. Klein, B. LaMacchia, P. Leach, J. Manferdelli, H. Maruyama, et al. Web Services Security (WS-Security). *IBM DEVELOPERWORKS*, 5:2002–04, 2002.

[19] K. Ballinger, D. Box, F. Curbera, S. Davanum, D. Ferguson, S. Graham, and K. Liu. Web Services Metadata Exchange (WS-MetadataExchange), 2004.

[20] S. Bechhofer, F.Harmelen, J. Hendler, I. Horrocks, D. L.McGuinness, P. F. Patel-Schneider, and L. A. Stein. OWL Web Ontology Language Reference. *http://www.w3.org/TR/owl-ref/*, 2004.

[21] D. Beckett. RDF/XML Syntax Specifictaion(Revised),W3C Recommendation. *http://www.w3.org/TR/rdf-syntax-grammar/*, 2004.

[22] T. Berners-Lee. Notation 3 (N3)–a readable RDF syntax, 1998.

[23] T. Berners-Lee. Cwm: Generation-purpose Data Processor for the Semantic Web. *http://www.w3.org/2000/10/swap/doc/cwm*, 2002.

[24] T. Berners-Lee. Semantic Web Tutorial Using N3. *Twelfth International World Wide Web Conference*, 2003.

[25] R. Bhattacharyyam, T. Mao, J. Williams, A. Sanchez, P. Hofmann, T. Lin, M. Lipton, and K Mantripagada. Distributed simulation to consider the scalability of proposed architecture models in rfid enabled supply chains.

[26] M. Biezunski, M. Bryan, and S. Newcomb. ISO/IEC 13250, Topic Maps. *Approved revised text*, 2002.

[27] S. Bono, M. Green, A. Stubblefield, A. Juels, A. Rubin, and M. Szydlo. Security Analysis of a Cryptographically-Enabled RFID Device. *14th USENIX Security Symposium*, pages 1–16, 2005.

[28] D. Brickley and R. V. Guha. RDF Vocabulary Description Language 1.0: RDF Schema. *http://www.w3.org/TR/rdf-schema/*, 2004.

[29] D. Brickley and R.V. Guha. RDF Vocabulary Description Language 1.0: RDF Schema. *W3C Working Draft*, 23, 2003.

[30] D.L. Brock. The Electronic Product Code (EPC): A Naming Scheme for Physical Objects. *Auto-ID Center. Cambridge, USA, Massachusetts Institute of Technology*, 2001.

[31] K. Cameron and M.B. Jones. Design Rationale behind the Identity Metasystem Architecture. *Isse/secure 2007 Securing Electronic Business Processes: Highlights of the Information Security Solutions Europe/secure 2007 Conference*, 2007.

[32] D. Carluccio, K. Lemke, and C. Paar. Electromagnetic side channel analysis of a contactless smart card: First results. *Proc. Ecrypt Workshop on RFID and Lightweight Crypto.*

[33] D. Connolly, F. van Harmelen, I. Horrocks, D.L. McGuinness, P.F. Patel-Schneider, and L.A. Stein. DAML+ OIL (March 2001) Reference Description. *W3C Note*, 18, 2001.

[34] G. Della-Libera, P. Hallam-Baker, M. Hondo, T. Janczuk, C. Kaler, H. Maruyama, N. Nagaratnam, A. Nash, R. Philpott, H. Prafullchandra, et al. Web Services Security Policy Language (WS-SecurityPolicy). *December*, 919:920–921, 2002.

[35] T. Dimitriou. A lightweight RFID protocol to protect against traceability and cloning attacks. *Proceedings of the IEEE International Conference on Secu*, 2005.

[36] H. Dobbertin. Cryptanalysis of MD5 Compress. *rump session of Eurocrypt*, 96, 1996.

[37] D. Engels, R. Rivest, S. Sarma, and S. Weis. Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems. *First International Conference on Security in Pervasive Computing, SPC*, 2003, 2003.

[38] EPCGlobal. Framework for Multi-Party Data Exchange. June 2007.

[39] D.C. Fallside and P. Walmsley. XML Schema Part 0: Primer Second Edition. W3C Recommendation 28 October, 2004. *World Wide Web Consortium*, pages 0–20041028, 2004.

[40] M. Feldhofer, S. Dominikus, and J. Wolkerstorfer. Strong authentication for RFID systems using the AES algorithm. In *Workshop on Cryptographic Hardware and Embedded Systems CHES 2004, LNCS 3156*, pages 357–370, MIT, MA, USA, 2004. Springer.

[41] K.P. Fishkin, S. Roy, and B. Jiang. Some Methods for Privacy in RFID Communication. *Security in Ad-Hoc and Sensor Networks: First European Workshop, ESAS 2004, Heidelberg, Germany, August 6, 2004: Revised Selected Papers*, 2005.

[42] B. Fitzpatrick. OpenID: an actually distributed identity system. *http://openid.net/*, 2005.

[43] C. Floerkemeier, R. Schneider, and M. Langheinrich. Scanning with a purpose-supporting the fair information principles in RFID protocols, 2004. 2005.

[44] J.T. Foley. An infrastructure for electromechanical appliances on the Internet. 1999.

[45] S.L. Garfinkel, A. Juels, and R. Pappu. RFID Privacy: An Overview of Problems and Proposed Solutions. *IEEE SECURITY & PRIVACY*, pages 34–43, 2005.

[46] P. Golle, M. Jakobsson, A. Juels, and P. Syverson. Universal reencryption for mixnets. *Conference -Cryptographers' Track (CT-RSA)*, 2964:163–178, 2004.

[47] N. Good, J. Han, E. Miles, D. Molnar, D. Mulligan, L. Quilter, J. Urban, and D. Wagner. Radio frequency identification and privacy with information goods. *Workshop on Privacy in the Electronic Society–WPES*, pages 41–42.

[48] T.R. Gruber. Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies*, 43(5/6):907–928, 1995.

[49] G.P. Hancke. A practical relay attack on ISO 14443 proximity cards. *Technical report, University of Cambridge Computer Laboratory*, 2005.

[50] P. Hayes. RDF Semantics. *W3C Recommendation*, 10, 2004.

[51] D. Henrici and P. Muller. Hash-based enhancement of location privacy for radio-frequency identification devices using varying identifiers. *Pervasive Computing and Communications Workshops, 2004. Proceedings of the Second IEEE Annual Conference on*, pages 149–153, 2004.

[52] S. Inoue and H. Yasuura. RFID privacy using user-controllable uniqueness. *RFID Privacy Workshop, MIT*, 2003.

[53] ISO. ISO 7816-4: Interindustry Commands for Interchange. *http://www.vdc-corp.com/*.

[54] A. Juels. Pappu., R.(2003). Squealing Euros: Privacy Protection in RFID-Enabled Banknotes. *Financial Cryptography–FC*, 3.

[55] A. Juels. Minimalist cryptography for low-cost RFID tags. *The Fourth International Conference on Security in Communication Networks–SCN*, 3352:149–164, 2004.

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

[57] A. Juels, R.L. Rivest, and M. Szydlo. The blocker tag: selective blocking of RFID tags for consumer privacy. *Proceedings of the 10th ACM conference on Computer and communications security*, pages 103–111, 2003.

[58] A. Juels, P. Syverson, and D. Bailey. High-Power Proxies for Enhancing RFID Privacy and Utility. 2005.

[59] R. Kasturi. Tapping the RFID Data Flood. *http://www.intelligententerprise.com/showArticle.jhtml?articleID=163100818*, 2005.

[60] Z. Kfir and A. Wool. Picking Virtual Pockets using Relay Attacks on Contactless Smartcard. *IEEE/CreateNet SecureComm*, 2005.

[61] D.S. Kim, T.H. Shin, and J.S. Park. A Security Framework in RFID Multi-domain System. *Proceedings of the The Second International Conference on Availability, Reliability and Security*, pages 1227–1234, 2007.

[62] R. Koh, E.W. Schuster, I. Chackrabarti, and A. Bellman. Securing the Pharmaceutical Supply Chain. *Auto-ID Center MIT, White Paper, September*, 2003.

[63] U. Kuter and D. Nau. Forward-chaining planning in nondeterministic domains. *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 513–518, 2004.

[64] H. Lockhart, B. Parducci, and A. Anderson. OASIS eXtensible Access Control Markup Language (XACML). *http://www.oasis-open.org/committees/tc-home.php*, 2005.

[65] H. Meisel and E. Compatangelo. EER-CONCEPTOOL: a "reasonable" environment for schema and ontology sharing. *Tools with Artificial Intelligence, 2002.(ICTAI 2002). Proceedings. 14th IEEE International Conference on*, pages 527–534, 2002.

[66] P. Mishra, H. Lockhart, S. Anderson, J. Hodges, and E. Maler. OASIS Security Services (Security Assertions Markup Language). *http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security*, 2006.

[67] P.V. Mockapetris and K.J. Dunlap. Development of the Domain Name System. 1988.

[68] D. Molnar, A. Soppera, D. Wagner, UC Berkeley, and B. Telecom. A Scalable, Delegatable Pseudonym Protocol Enabling Ownership Transfer of RFID Tags. *Selected Areas in Cryptography: 12th International Workshop, SAC 2005, Kingston, ON, Canada, August 11-12, 2005: Revised Selected Papers*, 2006.

[69] D. Molnar and D. Wagner. Privacy and security in library RFID: issues, practices, and architectures. *Proceedings of the 11th ACM conference on Computer and communications security*, pages 210–219, 2004.

[70] B. Clifford Neuman and Theodore Ts'o. Kerberos: An Authentication Service for Computer Networks. *IEEE Communications*, 32(9):33–38, September 1994.

[71] M. Ohkubo, K. Suzuki, and S. Kinoshita. Efficient hash-chain based RFID privacy protection scheme. *International Conference on Ubiquitous Computing–Ubicomp, Workshop Privacy: Current Status and Future Directions*, 2004.

[72] P.F. Patel-Schneider, P. Hayes, and I. Horrocks. OWL Web Ontology Language Semantics and Abstract Syntax. *W3C Recommendation*, 10, 2004.

[73] PingIdentityTM. Internet-Scale Identity Systems, An Overview and Comparison/whitepapers. Technical report, February 2007.

[74] H.S. Pinto, S. Staab, and C. Tempich. DILIGENT: Towards a fine-grained methodology for distributed, loosely-controlled and evolving engineering of ontologies. *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI)*, pages 393–397, 2004.

[75] E. Prudhommeaux, A. Seaborne, et al. SPARQL Query Language for RDF. *W3C Working Draft*, 4, 2006.

[76] M. Rieback, B. Crispo, and A. Tanenbaum. RFID Guardian: A battery powered mobile device for RFID privacy management. *Australasian Conference on Information Security and Privacy - ACISP*, 3574:184–194, 2005.

[77] S. Sarma, D.L. Brock, and K. Ashton. The Networked Physical World Proposals for Engineering the Next Generation of Computing, Commerce & Automatic-Identification http://www. autoidlabs. com/whitepapers. Technical report, MIT-AUTOID-WH-001. pdf.

[78] S.E. Sarma, S.A. Weis, and D.W. Engels. RFID Systems and Security and Privacy Implications. *Lecture Notes in Computer Science*, ISSU 2523:454–469, 2003.

[79] S.Santesson and R.Housley. Internet X.509 Public Key Infrastructure Authority Information Access Certificate Revocation List (CRL) Extension. *RFC 4325*, December 2005.

[80] Y. Sure, J. Angele, and S. Staab. OntoEdit: Multifaceted inferencing for ontology engineering. *Lecture notes in computer science*, pages 128–152.

[81] Dinoj Surendran. ARCHITECTURAL CHALLENGES OF SMART CARD DESIGN. *http://people.cs.uchicago.edu/~dinoj/smartcard/security.html.*

[82] K. Takaragi, M. Usami, R. Imura, R. Itsuki, and T. Satoh. An Ultra Small Individual Recognition Security Chip. 2001.

[83] S.A. Thomas. SSL and TLS essentials securing the Web. 2000.

[84] J. Williams, A. Sanchez, P. Hofmann, T. Lin, M. Lipton, and K Mantripagada. Modeling Supply Chain Network Traffic. Technical report, MIT-AUTOID Lab, April 2008.