IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 5 Issue 2, February 2018
ISSN (Online) 2348 – 7968
www.ijiset.com

Resolving Security and Privacy Issues in Radio Frequency Identification Middleware

Yassir Rouchdi[1], Khalid El Yassini[1] and Kenza Oufaska[2,1]
[1] IA Laboratory, Faculty of Sciences Meknes, Moulay IsmailUniversity, Morocco
[2] International University of Rabat, Rabat, Morocco

Abstract
This paper presents Radio frequency identification components, functioning and Middleware's role. It discusses RFID middleware architecture and explains its security and privacy issues; it includes a discussion about resolving these problems by applying Role based access Control model as an authentication tool regulating back-end application's access to data. Moreover, it presents the proposed architecture of our three layers middleware 'UIR-RFID', shows RBAC rules application and gives details on the implementation process.
Keywords: *Radio Frequency identification, Role based access control, middleware.*

1. Introduction

RFID technology has grown considerably in recent decades. The rapid advances of microelectronic transceivers have reduced the size and cost of HF and UHF RFID infrastructure, allowing longer and faster reading rates than ever before. RFID technology is now able to cope with new applications with greater mobility using a large number of components, allowing specific functionalities and general services and offering important advantages over other identification mechanisms.
However, new applications of technology require a robust, flexible and complex middleware platform to solve problems at different layers of the communication architecture in different contexts of the application. This complexity has set up research questions in the design of RFID middleware that still pose a high cost of entry for adopters of RFID technology.
The second-generation RFID (2G-RFID) systems contain not only static information such as object identification and description, but also introduce dynamic rules in encoding tags reflecting the update of Dynamic RFID sensor configuration service requirements. In practice, an RFID system always operates in an environment with multiple readout sensors and multiple protocols.
In order to ensure that the system can adapt to all working environments, RFID middleware must be used as it provides applications with a neutral device interface to communicate with different hardware components. The middleware design must be complete, fully compatible with standards of international organizations, and able to support simultaneous communication of several applications with several types of RFID hardware. EPCglobal, standards that are globally accepted standards that ensure global applicability, have been widely adopted by RFID middleware. EPCglobal Inc. is leading the development of industry-readern standards for electronic product codes (EPCs) to support RFID. It is the driving force behind the global adoption of CPE as a global standard to improve the visibility of products. Despite all this, RFID middleware still faces some major privacy and security issues. As a solution to these problems we developed a new middleware Architecture using Role based Access Control model as a method of regulating back-end application Access to data. In what follows, we will explain every layer of our middleware, its role, characteristics and implementation process.

2. Radio Frequency Identification Technology

RFID stands for Radio Frequency Identification. Its importance and efficiency are expressed by the vast amount of medical, military and commercial applications using this approach Worldwide [1]. Billions of the RFID systems are operated in transportation (automotive vehicle identification, automatic toll system, electronic license plate, electronic manifest, vehicle routing, vehicle performance monitoring), banking (electronic checkbook, electronic credit card), security (personnel identification, automatic gates, surveillance) and medical (identification, patient history) [2].

2.1 RFID Components

RFID systems are basically composed of three elements: a tag, a reader and a middleware deployed at a host computer. The RFID tag is a data carrier part of the RFID system, which is placed on the objects to be uniquely identified. The RFID reader is a device that transmits and receives data through radio waves using the connected antennas. Its functions include powering the tag, and reading/writing data to the tag [1].
Unique identification or electronic data stored in RFID tags can be consisting of serial numbers, security codes, product codes and other specific data related to the tagged object. The available RFID tags in today's market could be classified with respect to different parameters. For example with respect to powering, tags may be passive, semi-passive, and active. In terms of access to memory, the tags

may be read-only, read-write, Electrically Erasable Programmable Read-Only Memory, Static Random Access Memory, and Write-once read-many. Tags have also various sizes, shapes, and may be classified with respect to these geometrical parameters. The RFID reader is a device that transmits and receives data through radio waves using the connected antennas.

RFID reader can read multiple tags simultaneously without line-of-sight requirement, even when tagged objects are embedded inside packaging, or even when the tag is embedded inside an object itself. RFID readers may be either fixed or handheld, and are now equipped with tag collision, reader collision prevention and tag-reader authentication techniques [2, 3, 4]. Fig 1 illustrates RFID components.

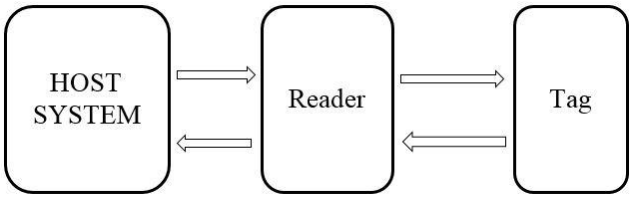

Fig. 1 RFID Components

2.2 Frequency Characteristics

Frequency refers to the size of the radio waves used to communicate between RFID systems components. RFID systems throughout the world operate in low frequency (LF), high frequency (HF) and ultra-high frequency (UHF) bands. Radio waves behave differently at each of these frequencies with advantages and disadvantages associated with using each frequency band. If an RFID system operates at a lower frequency, it has a shorter read range and slower data read rate, but increased capabilities for reading near or on metal or liquid surfaces. If a system operates at a higher frequency, it generally has faster data transfer rates and longer read ranges than lower frequency systems, but more sensitivity to radio wave interference caused by liquids and metals in the environment [5]. Table 1 shows different RFID Frequency applications.

Table 1: RFID carrier frequencies and their applications

	LF	***HF***	***UHF***	***Microwave***
Frequency range	125 - 134KHz	13.56 MHz	866 - 915MHz	2.45 - 5.8 GHz
Reading range	10 cm	1m	2-7m	1m
Applications	Smart Card, Ticketing, animal tagging, Access, Laundry	Small item management, supply chain, Anti-theft, library, transportation	Transportation vehicle ID, Access/Security , large item management, supply chain	Transportation vehicle ID (road toll), Access/Security, large item management, supply chain

2.3 RFID Middleware

Radio Frequency Identification (RFID) technology holds the promise to automatically and inexpensively track items as they move through the supply chain. The proliferation of RFID tags and readers will require dedicated middleware solutions that manage readers and process the vast amount of captured data [6].The efficiency of an RFID application depends on the precision of its hardware components, and the reliability of its middleware. Which is the computer software that provides services to software applications beyond those available from the operating system.

It can be described as "software glue" [7]. Middleware makes it easier for software developers to perform communication and input/output, so they can focus on the specific purpose of their application. Middleware includes Web servers, application servers, content management systems, and similar tools that support application development and delivery. It is especially integral to information technology based on Extensible Markup Language (XML), Simple Object Access Protocol (SOAP), Web services, SOA, Web 2.0 infrastructure, and Lightweight Directory Access Protocol (LDAP) [8, 9].

2.4 Middleware's basic functions

The three primary functions of an RFID middleware can be broadly classified as:

'Device integration' (that is, connecting to devices, communicating with them in their prescribed protocols and interpreting the data).

'Filtering' (the elimination of duplicate or junk data, which can result from a variety of sources, for example: the same tag being read continuously or spikes or phantom reads caused by interference)

'Feeding applications', with relevant information based on the information collected from devices after properly performing the appropriate conversions and formatting [7].

2.5 Middleware Architecture

The usual architecture of an RFID middleware is presented in figure 2:

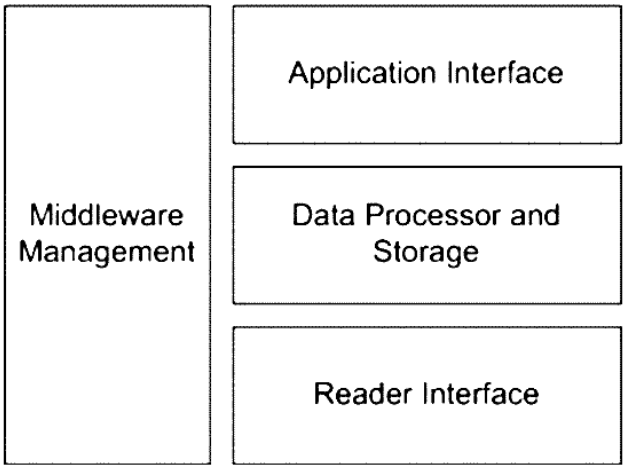

Fig. 2. Middleware Architecture [12]

Data Processor and Storage: The data processor is responsible for the management and processing of raw data from the readers. This component is also responsible for storing the raw tag data, so that it can be processed. Part of the important processing logic performed here is the Filtering and Grouping of RFID Data. This component also manages the level event data associated with the application. By way of example, when all applications request data captures in the same time interval, processing of the time stamp is performed by this component and the data is then transmitted to the applications.

Application Interface: The application interface component manages the interface of the middleware in dependence on those of the applications. It provides the application with an API (application programming interface) to communicate and evoke the RFID middleware. It accepts application requests and translates them down to the underlying components of the middleware. This component is responsible for integrating enterprise applications with RFID middleware.

Middleware Management: The middleware management component helps with the management of the RFID middleware conspiracy. It provides information about all the processes running in the middleware. The middleware management provides the administrator with the following features:

1. Add, remove, and modify the RFID readers connected to the system.
2. Change various settings by applications.
3. Enable and disable various functions supported by the middleware.

After going through RFID basic components and functioning process, we will now discuss the privacy and security issues occurring in both the Hardware and middleware parts of the system, the next chapter will present these issues and explain each one of it.

3. RFID Security and Privacy issues

3.1 Hardware Security and privacy issues

With the adoption of RFID technology, a variety of security and privacy risks need to be addressed by both organizations and individuals. RFID tags are considered "dumb" devices, in that they can only listen and respond, no matter who sends the request signal. This brings up risks of unauthorized access and modification of tag data [10]. In other words, unprotected tags may be vulnerable to eavesdropping, traffic analysis, spoofing or denial of service attacks.

- Eavesdropping (or Skimming): Radio signals transmitted from the tag, and the reader, can be detected several meters away by other radio receivers. It is possible therefore for an unauthorized user to gain access to the data contained in RFID tags if legitimate transmissions are not properly protected. Any person who has their own RFID reader may interrogate tags lacking adequate access controls, and eavesdrop on tag contents. Traffic Analysis: Even if tag data is protected, it is possible to use traffic analysis tools to track predictable tag responses over time. Correlating and analyzing the data could build a picture of movement, social interactions and financial transactions. Abuse of the traffic analysis would have a direct impact on privacy [11, 12].
- Denial of Service Attack: the problems surrounding security and trust are greatly increased when large volumes of internal RFID data are shared among business partners. A denial of service attack on RFID infrastructure could happen if a large batch of tags has been corrupted. For example, an attacker can use the "kill" command, implemented in RFID tags, to make the tags permanently inoperative if they gain password access to the tags. In addition, an attacker could use an illegal high power radio frequency (RF) transmitter in an attempt to jam frequencies used by the RFID system, bringing the whole system to a halt [11, 12].
- PERSONAL PRIVACY , As RFID is increasingly being used in the retailing and manufacturing sectors, the widespread item-level RFID tagging of products such

as clothing and electronics raises public concerns regarding personal privacy. People are concerned about how their data is being used, whether they are subject to more direct marketing, or whether they can be physically tracked by RFID chips. If personal identities can be linked to a unique RFID tag, individuals could be profiled and tracked without their knowledge or consent [12].

3.2 Middleware's Major issues

Some of the problems that are still occurring in nowadays middleware solutions are the following:
• Real time handling of incoming data from the readers Huge quantities of RFID tag data are directed to the middleware from different connected readers. The middleware must be able to process this flow of data in real time without any fault of reading.
• Multiple Hardware Support: The middleware must support multiple types of tags and readers by providing a common interface to access them. Different RFID hardware or different characteristics, the middleware must be able to access all the features and capabilities of the hardware.
• Synchronization and Scheduling in the middleware: The middleware is usually realized using several processes for processing readers, applications, and data processing elements. It should ensure intelligent planning and synchronization between all these processes. These factors define the latency and efficiency of the middleware.
• Servicing Multiple Applications: The middleware design must be able to provide service for several applications simultaneously. Different Applications therefore different requirements, the middleware must meet all the needs of applications with minimal latency.
• Device Neutral Interface to the applications: The middleware must provide the application of a hardware-independent display device. The author of the application should be able to develop applications using only the generic set of interfaces provided by the middleware.
• Scalability: The middleware design should allow new hardware to be easily integrated into the RFID system. This requires a modular design, where the modules can be added or removed depending on various configurations. The design should also allow easy integration of recent data processing features in the middleware.
The major problem facing RFID middleware is unauthorized access to data, occurring especially in the Application Abstraction Layer when backend end applications require information they are unauthorized to get. In order to resolve the problem we suggested applying an RBAC model in order to regulate access to data based on certain constraints, the next chapters will discuss RBAC policies and its application details.

4. Role-Based Access Control Model

4.1 Introduction (RBAC)

In order to resolve the security and privacy issues and prevent the RFID Tag data, we decided to use the role based access control access regulation method, so that only authorized users get access to specific data.

RBAC is a method of regulating access to computers or network resources based on the roles of individual users within a network [18]. In this context, access is the ability of an individual user to perform a specific task, such as view, create, or modify a file. Roles are defined according to authority and responsibility within the Network [19]. To clarify the notions presented in the previous section, we give a simple formal description, in terms of sets and relations, of role based access control. No particular implementation mechanism is implied. For each subject, the active role is the one that the subject is currently using: AR(s: subject) = {the active role for subject s}. Each subject may be authorized to perform one or more roles: RA(s: subject) = {authorized roles for subject s}. Each role may be authorized to perform one or more transactions: TA(r: role) = {transactions authorized for role r}. Subjects may execute transactions. The predicate exec(s,t) is true if subject 's' can execute transaction 't' at the current time, otherwise it is false: Exec(s: subject, t: tran) = true if subject s can execute transaction t.

4.2 RBAC Primary Rules

1. Role assignment: A subject can exercise a permission only if the subject has selected or been assigned a role.

   $\forall s$:subject,t :tran (, exec(s,t) $\Rightarrow$ AR(s) $\neq$ O/).

2. Role authorization: A subject's active role must be authorized for the subject. With rule 1 above, this rule ensures that users can take on only roles for which they are authorized.

   $\forall s$: subject (, AR(s) $\subseteq$ RA(s)).

3. Permission authorization: A subject can exercise a permission only if the permission is authorized for the subject's active role. With rules 1 and 2, this rule ensures that users can exercise only permissions for which they are authorized.

   $\forall s$: subject,t : tran (, exec(s,t) $\Rightarrow$ t $\in$TA(AR(s))).

A properly administered RBAC system enables users to carry out a broad range of authorized operations, and provides great flexibility and breadth of application. System administrators can control access at a level of abstraction that is natural to the way that enterprises typically conduct business. This is achieved by statically and dynamically regulating users, actions through the establishment and definition of roles, role hierarchies, relationships, and constraints [14, 15].
RBAC method of regulating access will guarantee data protection from unauthorized back-end applications, this method will be implemented in our own middleware solution 'UIR-RFID', the following chapter will illustrate UIR-RFID architecture and implementation details.

5. UIR-RFID Middleware

5.1 UIR- RFID Architecture

We propose to develop an RFID middleware called UIR-RFID bearing in mind the design problems discussed in the second chapter. Our RFID system is organized as a three-tiered architecture, with applications, middleware (UIR-RFID) and RFID hardware.
RFID hardware represents RFID readers and tags. The middleware (UIR-RFID) is the software component that provides applications with an interface independent of the device to access data from RFID hardware. RFID applications are independent software that use the services of (UIR-RFID) for various consumer requirements. These applications are developed using a generic Application Framework. This framework uses a device-independent visualization of the RFID hardware provided by UIR-RFID to create applications.
UIR-RFID offers a design that provides the application with a neutral device protocol and an independent platform interface. It integrates three hardware abstraction layer (HAL), event and data management layer (EDML) and Application Abstraction Layer (AAL).
The next figure (Fig. 3) illustrates UIR-RFID architecture.

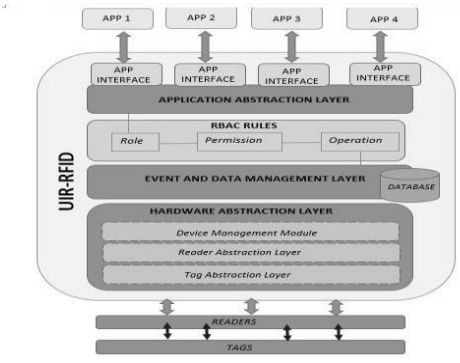

Fig. 3 UIR-RFID Architecture.

5.1. Hardware Abstraction Layer

HAL is the lowest layer of (UIR-RFID) and is responsible for interaction with the RFID hardware. It allows access to devices and tags in an independent manner of their various characteristics through layers of tag abstraction and reader. UIR-RFID provides the view of tag data as a byte stream.
This level of abstraction provides independence from various characteristics of tags such as protocols (ISO 14443 [2], EPC Gen2 [14], ISO 15693 [20], etc.), air interfaces [20] (HF, UHF) and Memory Size [19].
The reader abstraction provides a common interface for accessing RFID hardware devices with different characteristics such as protocols (ISO 14443, EPC Gen2, ISO 15693), UHF (HF) and host side interface Interface (RS232, USB, Ethernet).
The abstraction of the reader exposes simple functions such as opening, closing, reading, writing, etc. To accomplish complex operations of the readers.
The abstraction of readers and tags in UIR-RFID make it extensible to support various tags and readers.
The device management module in HAL is responsible for the dynamic loading and unloading of the readerr libraries depending on the use of device hardware. This allows the system to be light because only the required libraries are loaded. This layer contains the devices for various operations, as specified by the upper layers. It is also responsible for monitoring and reporting the status of the device. Some of the functions provided by the HAL to access RFID hardware are as follows:
• The Device-opening: function is responsible for opening a connection with the device. The connection parameters are provided as an argument to this function. When a successful connection is made to the reader, a response is returned by this function. This response is then used as a reference to access the device in subsequent calls.
• The Device-reading function: reads data from the internal reader. The read parameters such as the protocol to be read by the reader, the size of the data to be read, are specified as arguments of this function. The function responds successfully if valid data is present in the reader if not with an error code.
• The Device-Writing function writes data to the Tag. Arguments Specified with this function, the unique ID partially or totally, which triggers the data to be written to the tag. The function responds successfully if the data is written to the Tag or returns an error code (for example, when the tag is not identified only).

5.2 Event and Data Management Layer (EDML)

EDML handles various reader-level operations, such as reading tags and informing readers of disconnected notions

such as device failure, write failure, and so on. The layer acts as a conduit between the hardware abstraction layer (HAL) and the application abstraction layer (AAL). It accepts commands from AAL, processes them and therefore issues commands to HAL. Similarly, the responses are brought from the HAL, processed and transmitted to the LAA by this layer. This layer serves as temporary storage for the incoming data of the reader. It processes data by grouping, reading, and removing redundancy, as described below.
Grouping: Grouping involves the accumulation of data from different readers. Initially, readers read the data internally. By grouping, data received from these readers is accumulated in a single location. This accumulation is based on the hardware configuration made by the applications. These grouped data are then used for further processing by UIR-RFID such as sorting schematically.
Filtering: There are cases where applications only require identifiers specific to certain Tags. The filtering provides this functionality by selectively reporting the tag data. In UIR-RFID the abstraction tags are considered as a stream of bytes. Therefore, we choose the filtering values provided by the application, in the form of consecutive bytes. This allows you to tackle multiple tag data formats. Deleting duplicates: Tags near the readers are played continuously. This results in a large amount of data repeated by the readers. Removal of these duplicates is a feature provided by UIR-RFID, to prevent the report from repetitive data. It is characterized by a time value specified by the application. The same tag data read by the reader within the specified time is indicated only once.

5.3 Application Abstraction Layer (AAL)

The Application Abstraction Layer (AAL) provides various applications with an independent interface to RFID hardware. The interface is designed as an API by which Applications use UIR-RFID services. All operations at the application level such as reading, writing, etc. Are interpreted and translated into the lower layers of UIR-RFID by the AAL.

5.4 UIR-RFID Implementation

For the UIR-RFID implementation, we propose the use of Microsoft Visual Studio .Net 2010 as Framework and development tool. The reasons for this choice are the powerful utilities for Application Development that this framework provides. The code to use to develop the Project is C Sharp (C #). We propose the use of graphical user interface features provided by the .Net Framework and Microsoft Visio 2013 to develop conceptual models and middleware architecture. For the purpose of data management and storage, we offer Microsoft SQL Server 2008, It is a cohesive set of tools, utilities and interfaces collaborating to provide excellent data management. The database schema generated by this DBMS provides a comprehensive view of the data and its relationships. To view the database, retrieve, modify, delete, and store data, we propose the use of the SQL language (Structured Query Language).

5.5 RBAC Rules and Assignments

1. System defines
   Permissions: to perform some action
   Roles: a set of permissions that have some relation

2. Operations
   Allow: Assign and allow
   Deny: Assign and do not allow
   Remove: Remove

3. Precedence of operations
   Allow
   Deny

- If you are granted some action by a role but you have denied that permission, the action cannot be done.

4. Rules
   1. Permissions can have linked permissions (thus creating a role).
   2. An account can be assigned granted and denied roles.
   3. Permissions inherited from roles are granted if roles is granted and denied if roles is denied.
   An account can be assigned granted and denied permissions.
   5. An account can have multiple roles and permissions.
   6. An account can not have same permission or role granted and denied at same time.
   7. Id 0 can not be used to define a permission

table.2 shows a listing of available permissions and associated id.

Table 2: Listing of available permissions

Name	***Syntax***	***Description***
.rbac account	Syntax: .rbac account [$account]	View permissions of selected player or given account Note: Only those that affect current realm Note: Shows real permissions after checking group and roles
.rbac account permission	Syntax: .rbac account list [$account]	View permissions of selected player or given account Note: Only those that affect current realm Note: Only those directly granted or denied, does not include inherited permissions from roles
.rbac account allow	Syntax: .rbac account allow [$account] #id [#realmId]	Grant a permission to selected player or given account. #reamID may be -1 for all realms.
.rbac account deny	Syntax: .rbac account deny [$account] #id [#realmId]	Deny a permission to selected player or given account. #reamID may be -1 for all realms.
.rbac account remove	Syntax: .rbac account remove [$account] #id	Remove a permission from an account Note: Removes the permission from granted or denied permissions
.rbac list	Syntax: .rbac list	View list of all permissions. If $id is given will show only info for that permission

6. Conclusion

Radio Frequency Identification has shown its efficiency through its many applications in different areas such as airlines luggage tracking, Supply chain management, military use, healthcare and personal services. Despite the technology's worldwide deployment, researches carry on the purpose of resolving many related issues cited in the third section. Our proposed middleware (UIR-RFID) architecture offers a solution to many issues discussed in earlier chapters.

Starting by resolving the Multiple Hardware Support issue on The reader abstraction layer that provides a common interface for accessing RFID hardware devices with different characteristics such as protocols (ISO 14443, EPC Gen2, ISO 15693), UHF (HF) and host side interface (RS232, USB, Ethernet). Resolving Synchronization and Scheduling in the middleware on The EDML that manages data flow between the other layer handling various reader-level operations, such as reading tags and informing readers of disconnected notions such as device failure, write failure, and so on. Servicing Multiple Applications and offering a Device Neutral Interface to the applications on The Application Abstraction Layer (AAL) that provides various applications with an independent interface to RFID hardware. Resolving Scalability problems on The Hardware Abstraction Layer that allows access to devices and tags in an independent manner of their various characteristics through layers of tag and reader abstraction. Moreover, UIR-RFID fixes Regulating Access to data by using The RBAC Mechanism that makes sure only authorized users (applications) access the needed data depending on the permissions allowed and the role assigned.

References

1. M. Ajana, M. Boulmalf, H. Harroud and H. Hamam , "A Policy Based Event Management Middleware for Implementing RFID Applications", IEEE International Conference on Wireless and Mobile Computing, 2009

2. United States Government Accountability Office, "INFORMATON SECURITY Radio Frequency Identification

Technology in the Federal Government," United States Government Accountability Office, May 2005. [Online].

3. Q. Sheng, X. Li, and S. Zeadally, "Enabling Next-Generation RFID Applications: Solutions and Challenges", IEEE Computer, Vol. 41 (9), 2008

4. H. Al-Mousawi, "Performance and reliability of Radio Frequency Identification (RFID)", in Agder University College, 2004

5. N. Kefalakis, N. Leontiadis, J. Soldatos and D. Donsez, "Middleware Building Blocks for Architecting RFID Systems", Mobile Lightweight Wireless Systems, Vol 13, pp 325-336. 2009

6. X. Su, C. Chu, B.S. Prabhu, and Rajit Gadh, "On the creation of Automatic Identification and Data Capture infrastructure via RFID and other technologies" Taylor & Francis Group, 2007

7. M C. Bornhövd, T. Lin, S. Haller, and J. Schaper, "Integrating Automatic Data Acquisition with Business Processes - Experiences with SAP's Auto-ID Infrastructure". In Proceedings of the 30st international conference on very large data bases (VLDB). Toronto, pp 1182–1188, 2004

8. S. Bell, "RFID Technology and Applications", Cambridge University Press, pp. 6–8, London, 2011

9. M. Catherine O'Connor, "Rfid is the Key to Car Clubs Success", RFID JOURNAL, 2011

10. R. Russell, "Manufacturing Execution Systems: Moving to the next level", Pharmaceutical Technology, pp 38-50 , 2004

11. M. Darwish, "Analysis of ANSI RBAC Support in Commercial Middleware", PhD Thesis, University of British Columbia, Vancouver, Canada, 2009

12. R. Ferraiolo, D.F. Kuhn and D.R. Sandhu "The NIST Model for RoleBased Access Control: Toward a Unified Standard", 5th ACM Workshop Role-Based Access Contro, pp: 47–63, 2000

13. D.F. Kuhn and D.R. Sandhu, "RBAC Standard Rationale: comments on a Critique of the ANSI Standard on Role-Based Access Control", IEEE Press. 5 (6), pp: 51–53 , 2007

14. M. Zuluaga, J. Montanez and J. van Hoof, "Green Logistics – Global Practices and their Implementation in Emerging Markets", pp: 2-3, Colombia, 2011

15. R. Sandhu and Edward J. Coynek , Hal, L. Feinsteink and Charles E. Youman "Role-Based Access Control Models" , IEEE Computer, Vol 29 (2), pp: 38-47, 1996,

16. R. Sandhu "Role-Based Access Control (RBAC)", CS 6393 Lecture 3, 2016

17. T. Zhang, Y. Ouyang and Y. He, "Traceable Air Baggage Handling System Based on RFID
Tags in the Airport", School of Computer Science and Engineering, Beijing University of Aeronautics and Astronautics, China, Journal of Theoretical and Applied Electronic Commerce Research, Vol 3 (1) , pp: 106-115, 2008

18. R. Weil and E. Coyne, "ABAC and RBAC: Scalable, Flexible, and Auditable Access Management", IT Professional, vol. 15 (4), pp. 14-16, 2013

19. J. Caiyuan, S. Aodong and Y.Wenxue, "The RBAC System Based on Role Risk and User Trust", International Journal of Computer and Communication Engineering, 2016