Enhancing RFID Performance and Security in Networked Environments

by

Harinda Sahadeva Fernando
BBIT (Hons.)

Submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy

Deakin University
July, 2012
I am the author of the thesis entitled “Enhancing the Performance and Security of Networked RFID Systems” submitted for the degree of “Doctor of Philosophy”.

This thesis may be made available for consultation, loan and limited copying in accordance with the Copyright Act 1968.

'I certify that I am the student named below and that the information provided in the form is correct'

Full Name: ...Harinda Sahadeva Fernando .................................................................

Signed:........................... Signature Redacted by Library ...........................................

Date:............07/11/2012............................................................................................
DEAKIN UNIVERSITY

CANDIDATE DECLARATION

I certify that the thesis entitled “Enhancing the Performance and Security of Networked RFID Systems” submitted for the degree of “Doctor of Philosophy” is the result of my own work and that where reference is made to the work of others, due acknowledgment is given.

I also certify that any material in the thesis which has been accepted for a degree or diploma by any university or institution is identified in the text.

'\textit{I certify that I am the student named below and that the information provided in the form is correct}'

Full Name .............................................. Harinda Sahadeva Fernando

Signed .............................................. [Signature Redacted by Library]

Date .............................................. 13/06/12
Acknowledgements

First of all, I would like to thank Prof. Jemal Abawajy, my supervisor. Without his generosity, time and his motivation it would have been impossible to stay focused and finish the thesis on time. I admire his skill and knowledge in many areas of computer science and his abilities as a teacher, mentor and supervisor. I am grateful for the constant interest and time he attested in my work and for all those opportunities he afforded me to conduct my research, to publish, to travel and to teach, etc. What impressed me the most was his humbleness and his willingness to share and engage with all around him: truly an inspiration to everyone who has had the honour to work with him. I thank him for the trust he placed on me and consider myself to be blessed to have him as my supervisor.

I really enjoyed the time I was PhD student at the Networked Computing and System Security research lab. This was mainly due to the wonderful atmosphere that was in the NCSS research group. I met here people from many countries who generously shared their work and life experiences with me. I thank all the NCSS members for those four years in their company. I would like to salute the PhD students with who I shared this adventure: especially Farhan, Nizam, Soon and Shivali.

During the course of my thesis work, there were many people who were fundamental in helping me. Therefore, special thanks go out to all the staff of the Deakin, School of IT at Waurn Ponds. Without their guidance, help, and patience, I would have never been able to accomplish the work of this thesis. I would like to take this opportunity to acknowledge them for their help, either technical or moral, or both.

I would also like to thank my two cousins and their families who live here in Melbourne. There constant help and support over the duration of my candidature has been a great source of comfort. Last but not least, I would like to thank my parents and sister, for the support they provided me through my entire life. Without their limitless love, encouragement, confidence in my abilities and assistance, I would not have finished this thesis.
Abstract

Radio Frequency Identification (RFID) is a technology that allows the identification of objects from a distance even without line-of-sight. RFID systems hold the potential for significant improvements in the speed and data sharing in a number of different types of systems. But they also present a number of performance issues and significant security risks. This is especially true in very large scale networked systems used by multiple independent partners: mainly due to some of the specific features of those types of systems. The main performance issues associated with RFID use in these types of systems are (1) The lack of a system architecture optimised for the specific requirements of large scale networked RFID systems and (2) The amount of computational power available on low cost RFID tags. Security-wise the lack of a comprehensive security framework targeted at networked RFID systems is a big concern as there are a large number of different types of RFID systems with different security needs. Additionally most of the currently existing RFID protocols are either unsecure or too resource intensive for use on low cost RFID tags. The possibility of RFID based malware, which can infect RFID tags and from there spread to backend databases and other tags, has also recently been identified by researchers.

In this thesis we propose and present a number of methods with which the performance and security issues in networked RFID systems can be improved. First an RFID architecture optimised for large scale networked RFID systems is developed. The proposed architecture has a fully modularised middleware allowing for easy development and uses P2P techniques for both the data lookup and data sharing. In the comparative analysis carried out we show that the proposed architecture has a number of advantages over existing solutions when used in developing very large scale networked systems. Next we present a holistic and comprehensive security framework for networked RFID. The security framework takes into account the threats and attacks faced by networked RFID systems, the system components that can be affected by them and the security functionality required to secure the overall system. By applying the framework to any RFID system developers/users can easily identify potential threats and attacks to that system and identify how those threats and attacks can be mitigated. Next, we develop and then present a hybrid RFID authentication protocol developed
using a mix of traditional cryptographic primitives and ultra-light-weight cryptographic
techniques. The security and performance evaluations show that the proposed protocol
is very secure and that it is lightweight enough to be implemented on low cost RFID
tags. Finally we develop and present a policy-based RFID malware detection and
prevention approach. The proposed approach uses both defensive coding techniques
(data validation and sanitization) and active detection (SQL query structure matching).
It is also a simple solution that is easily implementable at a single point in the
middleware and allows the automated detection of tags that are infected with malware
and prevents that malware from spreading to the backend databases. The security
evaluation shows very high detection rates and low false positives. It is also capable of
detecting second order injection attacks: an attack that most existing approaches are
unable detect.
Publications


# Table of Contents

Acknowledgements .............................................................................................................. iv  
Abstract ................................................................................................................................. v  
Publications .............................................................................................................................. vii  
Table of Contents ................................................................................................................... viii  
List of Figures ........................................................................................................................ xiv  
List of Tables ............................................................................................................................ xv  
List of Acronyms .................................................................................................................... xvi  
Introduction ............................................................................................................................ 1  
  1.1 Radio Frequency Identification ....................................................................................... 1  
  1.2 Current Issues and Research Motivation ........................................................................ 3  
    1.2.1 Performance Issues in Networked RFID ................................................................. 4  
    1.2.2 Security Issues in Networked RFID ....................................................................... 4  
  1.3 Research Problems ....................................................................................................... 5  
  1.4 Methodology ................................................................................................................. 7  
  1.5 Major Contributions and Significance ......................................................................... 8  
  1.6 Thesis Organization ..................................................................................................... 9  
Literature Review .................................................................................................................. 12  
  2.1 RFID in Global Supply Chains ...................................................................................... 12  
  2.2 Networked RFID System Model .................................................................................. 15  
    2.2.1 RFID Tags ............................................................................................................... 16  
    2.2.2 RFID Readers ....................................................................................................... 17  
    2.2.3 RFID Middleware ................................................................................................... 17  
    2.2.4 Lookup Service ...................................................................................................... 18  
    2.2.5 Data Storage .......................................................................................................... 18  
    2.2.6 Business Applications ........................................................................................... 19  
    2.2.7 Communication Network ...................................................................................... 19  
  2.3 RFID Architectures ..................................................................................................... 20  
    2.3.1 The EPCGlobal Architecture Framework ................................................................. 20
5.3.4.2 Performance Enhancement ................................................................. 116

5.5 Security Analysis and Comparison ............................................................ 117

5.5.2 Protocol Security Properties ..................................................................... 117

5.5.2.1 Mutual Authentication ....................................................................... 118

5.5.2.2 Data Confidentiality ........................................................................... 118

5.5.2.3 Data Integrity ..................................................................................... 119

5.5.2.4 Tag Anonymity .................................................................................. 119

5.5.2.5 Backward Security ............................................................................ 119

5.5.2.6 Forward Security .............................................................................. 120

5.5.3 Protection from Common RFID Attacks .............................................. 120

5.5.3.1 Protection against Replay Attacks .................................................... 120

5.5.3.2 Protection against De-synch Attacks ............................................... 121

5.5.3.3 Protection against Man-in-the-middle ............................................. 122

5.5.3.4 Protection against Eavesdropping Attacks ....................................... 122

5.5.3.5 Protection against Crypto Attacks .................................................... 122

5.5.3.6 Protection against Data Leakage Attacks ........................................ 123

5.5.4 Security Comparison ............................................................................. 123

5.5.5 Potential Weaknesses .......................................................................... 124

5.5.5.1 Limited Tag Tracking ....................................................................... 125

5.5.5.2 Jamming Attacks .............................................................................. 125

5.6 Performance Analysis and Comparison ................................................. 126

5.6.1 Computational Cost ............................................................................ 126

5.6.2 Storage Requirements ......................................................................... 127

5.6.3 Communication Cost ......................................................................... 128

5.7 Summary ................................................................................................. 128

RFID Malware Detection ............................................................................. 130

6.1 Introduction ............................................................................................ 131

6.2 SQL Injection Attacks ............................................................................ 133

6.2.1 Definition of SQLIA ........................................................................... 133

6.2.2 Malware in RFID Systems ................................................................. 134
List of Figures

Figure 2.1 A supply chain
Figure 2.2 Networked RFID architecture
Figure 2.3 EPCGlobal RFID architecture framework
Figure 3.1 P2P Networked RFID architecture
Figure 3.2 RFID tag data storage
Figure 3.3 A RFID system with multiple readers
Figure 3.4 RFID tag data translation
Figure 3.5 RFID event generation
Figure 3.6 RFID data retrieval, aggregation and formatting
Figure 3.7 Discovering the data services of new partners
Figure 3.8 Discovering new data services of existing partners
Figure 4.1 Threat model for networked RFID
Figure 4.2 Attack model for networked RFID
Figure 4.3 Networked RFID security framework
Figure 4.4 Components of networked RFID
Figure 4.5 Applying the security concepts to the components
Figure 5.1 Protocol architecture
Figure 5.2 Proposed RFID protocol
Figure 5.3 Data packet structure
Figure 5.4 Security enhancement for data transmission
Figure 5.5 Performance enhancement for data transmission
Figure 6.1 Tag based malware in RFID systems
Figure 6.2 Policy based RFID malware detection and prevention
Figure 6.3 Sanitization data creation
Figure 6.4 Tokenized query
Figure 6.5 Tokenized query with literals removed
Figure 6.6 Example of a parsed query
Figure 6.7 Example of a query comparison
List of Tables

Table 1.1 RFID system features
Table 2.1 EPCGlobal data lookup
Table 2.2 Security framework comparison
Table 3.1 RFID tag reads from multiple readers
Table 3.2 RFID transaction data
Table 3.3 Middleware interfaces
Table 3.4 RFID data types
Table 3.5 Identification data for transaction information
Table 3.6 Example Service Profile
Table 3.7 Example Data Profile
Table 3.8 Comparison of EPCGAF and proposed architecture
Table 4.1 Networked RFID threat model
Table 4.2 Mapping of attack classification to Microsoft STRIDE model
Table 4.3 RFID attacks by their possible impact on system
Table 4.4 RFID attacks by access method and attack source
Table 4.5 Networked RFID security requirements
Table 4.6 Threats, weaknesses and attacks applicable to E-Passport systems
Table 4.7 Impact that the attacks can have on system in question
Table 4.8 Access methods and attack sources of attacks on system
Table 4.9 E-Passport security requirements
Table 4.10 Defences for RFID enabled passport systems
Table 5.1 Data storage on key components
Table 5.2 Functionality of key components
Table 5.3 Security comparison of some common RFID Protocols
Table 5.4 Performance comparison of some common RFID Protocols
Table 6.1 Differences in web based SQLIA and RFID SQLIA
Table 6.2 Types of SQLIA
Table 6.3 Example validation policy table
Table 6.4 Example sanitization policy table
Table 6.5 Legal query structure table
Table 6.6 Evaluation results
Table 6.7 Security comparison table
Table 6.8 Comparison of SQLIA detection techniques
List of Acronyms

RFID - Radio Frequency Identification
UHF - Ultra High Frequency
EPCGAF - EPCGlobal Architecture Framework
DoS - Denial of Service
NDFA - Non-Deterministic Finite Automaton
P2P - Peer-to-peer
SQLIA - SQL Injection Attacks
EPC - Electronic Product Code
ONS - Object Name Service
LAN - Local Area network
WAN - Wide Area network
EPCIS - Electronic Product Code Information Service
DNS - Domain Name Service
DHT - Distributed hash table
CPN - Company prefix Number
SOAP - Simple Object Access Protocol
SPOF - Single Point of Failure
DDoS - Distributed Denial of Service
CRC - Cyclic Redundancy Check
PRNG - Pseudo-Random Number generator
BAC - Basic Access Control
EAC - Extended Access Control
MRZ - Machine Readable Zone
HRSP - Hybrid RFID Security Protocol
ROM - Read Only Memory
DBMS - Database Management System
Chapter 1

Introduction

1.1 Radio Frequency Identification

At its core Radio Frequency Identification (RFID) is a technology that consists of a electromagnetic tag or chip attached to an object and contains a unique identification number. A reader is used to read the contents of the tag from a distance without line-of-sight [1]. Therefore RFID can be defined as a non-contact, proximity based, automatic identification and information generation technology that does not require manual scanning or line-of-sight [2]. Use of RFID over legacy tracking and identification technologies such as barcodes enable a number of advantages including automation, improved response times, much greater information generation and real-time sharing of that data [3].

RFID technology as a concept is relatively old. It was first used during the Second World War as a method of differentiating between enemy and allied air craft. One of the earliest academic papers on RFID was [4] in which Stockholm, the author, presented the very first concept of a passive radio transmitter powered by the radio waves from another transmitting device. But the first “true ancestor” of RFID can be considered to be the device patented by Mario Cardullo in 1973 as it was the first passive radio transponder with on-board memory. Since then RFID technology and tags have advanced in leaps and bounds. In 2009 the typical cost of passive RFID tags had already dropped as low as 0.07-0.05 US dollars [5]. This sudden advance in RFID technology and the corresponding drop in prices have made RFID enabled systems appealing in a number of different environments and areas.
Therefore the potential applications of RFID technology are vast and currently RFID systems are used in a number of diverse application areas such as transportation, fleet management, payment systems and livestock identification [1]. Due to the diverse nature of potential RFID systems their features also greatly differ. Standalone payment systems such as toll payment systems are typically implemented in a very small geographic area while cattle tracking systems are used in large geographic areas. Some systems such as global supply chain management systems are distributed over extremely large networked RFID systems used by multiple partners spread over a very large geographic area [6]. Table 1.1 illustrates the features of some of these different RFID systems.

Table 1.1: RFID system features

<table>
<thead>
<tr>
<th></th>
<th>Geographic area</th>
<th>Number of partners</th>
<th>Number of tags</th>
</tr>
</thead>
<tbody>
<tr>
<td>Toll payment systems</td>
<td>Very small</td>
<td>Single</td>
<td>Medium</td>
</tr>
<tr>
<td>Cattle tracking systems</td>
<td>Large</td>
<td>Single</td>
<td>Small</td>
</tr>
<tr>
<td>RFID enabled hospitals</td>
<td>Very small</td>
<td>Few</td>
<td>Medium</td>
</tr>
<tr>
<td>RFID enabled passports</td>
<td>Very large</td>
<td>Medium</td>
<td>Large</td>
</tr>
<tr>
<td>Global supply-chain management</td>
<td>Very large</td>
<td>Very large</td>
<td>Very large</td>
</tr>
</tbody>
</table>

One application of RFID that has attracted a large amount of attention in recent years is its use in very large networked RFID systems [7]. One such example is the use of RFID in global supply chain management applications. In 2003-2005, Wal-Mart, the biggest retailer in the world, and the creator of many supply chain management innovations such as hub and spoke system and just-in-time inventory replacement, started using RFID to streamline its supply chains [8]. Lately a number of other organizations including Marks and Spencer of UK and Airbus have also started RFID projects. Airbus is claiming savings of millions of Euros each year by cutting process cycle times, eliminating paperwork, and reducing inventory through the use of RFID [9]. When deployed in this type of environment, RFID infrastructure can provide increased visibility throughout the network by enabling an efficient means to uniquely identify each container, pallet, case and item being manufactured, shipped and sold [10]. The automation possible with the RFID technology allows the automatic generation and sharing of large amounts of transaction data with partners along the network. It also allows a great number of business processes to be automated and made more efficient.
In addition, RFID allows the leveraging of more real time information to manage and streamline the physical goods network more efficiently [11]. Therefore Wal-Mart hoped to cut storage costs, reduce stock disappearances and minimize lost sales due stock outages in retail stores by moving from barcodes to RFID [12].

With the newly generated interest in the use of global RFID systems there needed to be a centralized organization to set the standards for, and ensure the interoperability of, different RFID applications and systems. Therefore the EPCGlobal was founded in 2003 by the Auto-ID Centre, an academic research project run by the Massachusetts Institute of Technology. Its main goal was “to try and facilitate the creation of RFID systems and to develop standards for those systems” [13]. Since then the main focus of this organization has been the development of worldwide standards for RFID and the use of the internet to share RFID data via the EPCGlobal network. To this purpose they created the EPCglobal architecture framework (here on referred to as EPCGAF). In 2010 the adoption of RFID was greatly accelerated by three main factors: (1) the steadily decreasing price of RFID tags as well as the increase in their computational power and memory capacity expanded the number of possible profitable uses of RFID systems, (2) Increase in tag read reliability and read rates leading to much more reliable RFID systems [14] and (3) The global adoption of a stable international standard around UHF passive RFID [15]. All of these advances were mainly made possible by the work done by the EPCGlobal.

1.2 Current Issues and Research Motivation

While the promise and possibilities of RFID-enabled supply chains were huge the benefits failed to actually materialize to quite the degree forecast in some types of systems. This was because RFID-enabled large scale global supply chains have a number of requirements and features that differentiate them from other, simpler, RFID enabled systems [16]. In addition, while the security and performance of RFID systems have greatly increased over the last few years there are still security and performance issues, especially in networked systems employing low cost passive tags, that create barriers to truly global adoption of networked RFID [11, 17].
1.2.1 Performance Issues in Networked RFID

A main requirement in large-scale networked RFID systems is the need for high throughput of tag reads. This is because some points, such as large-scale warehouses and other storage facilities, can store millions of RFID-tagged items. Therefore to ensure timely reading and retrieval of the tagged information, the performance of RFID tag reads for supply chain applications are quoted at 100/sec. With multiple readers at each location this means that the overall data lookup and retrieval rate for that particular location and its middleware may well exceed 5000/sec for large RFID locations. Therefore one of the biggest barriers to networked RFID adoption is read and data retrieval performance [18].

The current performance issues in large scale RFID systems also stem from the use of unoptimised system architecture when building the systems in question. The current standard for RFID systems architecture is the EPCGlobal Architecture Framework. To ensure that the framework can be used to develop many fundamentally different types of RFID systems the EPCGlobal has built the architecture framework around a highly generalized model of an RFID system. Therefore the resulting architecture contains some weaknesses when used in massively networked environments [2]. In addition, a majority of the other architectures found in literature also contain various weaknesses when building global supply chain management systems. Therefore identifying and then removing or minimizing the weaknesses of existing architectures when building large scale networked RFID systems will increase the performance of those systems and thereby increase the adoption rates.

1.2.2 Security Issues in Networked RFID

Another major barrier to the adoption of RFID is the current security and privacy issues associated with it. Because RFID tags use wireless communications with a max read distance of around 10 meters and typical read distance of around 3-5 meters it becomes very easy for potential attackers to eavesdrop or modify legitimate communications between tags and readers [15]. In extreme cases it is even possible for the attackers to masquerade as authorised tags or readers and gain access to valuable private information [19]. Additionally because RFID tags are attached to mobile physical objects attackers can invade the privacy of people and corporations using RFID tags.
simply by tracking their movement over time [20]. Finally unlike other data storage components which are typically physically secured, RFID tags are much more physically accessible to attackers. This compounds the above security issues as attackers can potentially steal the tags and bypass any security functionality and directly access any data stored on the tag [17].

In addition to the above mentioned issues networked RFID systems with multiple independent partners face a number of additional security and privacy issues. In this kind of system the RFID tag can be thought of as a communication device between partners. One partner writes/changes the data on the tag and another partner later on receives that data. Therefore non-repudiation becomes a requirement. Like-wise in this kind of system a partner may wish to store private data that only he should have access to on the tag. Therefore functionality such as access control and data ownership is required as well.

Another major factor contributing to the lack of security in RFID systems is the low amount of computational power available on low cost tags. Due to price constraints most global supply chain management systems use low cost passive tags [21]. Unfortunately these tags only contain around 10000 gates. This means that the use of any complex computations or processes on the tag will dramatically increase the time required to carry out that process. Therefore another very interesting technical hurdle is implementing the above mentioned security functionality on low cost RFID tags while still ensuring that the time required to carry out those processes remain at an acceptable level [22].

1.3 Research Problems

The overall goal of this thesis is on improving the performance and security of networked RFID systems. In particular we focus on large scale systems that are spread over a very large geographic area and are used and shared by a number of independent business entities e.g.:– Global supply chain management systems. To achieve this goal we tackle a number of different research challenges. The main challenges when conducting this research was ensuring that the complex nature of networked RFID systems are taken into account in any proposed solutions and that the trade-off between security and performance vs. the amount of resources available on low cost tags is
managed in the best possible manner. In particular we identify and investigate the following four research problems:

- **How can the architecture of a large scale networked RFID system be improved** – Currently there are a number issues in the existing RFID architectures including lack of sufficient scalability, security issues, key and data management issues, data lookup and retrieval problems and concerns on overall system availability [23]. Here the focus should be on improving the overall architecture used for developing this type of system and improving the manner in which the data lookup, retrieval and formatting can be done between partners and ensuring strong scalability, availability and security of the overall system.

- **What are the security issues inherent in large scale networked RFID systems with multiple independent partners, and how can those security issues be mitigated/removed** – The complex nature of networked RFID systems mean that they face a lot of different threats and attacks [24], and therefore have a large number of different security requirements [25]. Hence a comprehensive framework needs to be created that takes into account various different factors such as the threats and attacks possible on RFID systems and the core components that must be protected and the security concepts that must be maintained. The framework should allow methodical identification of each of these factors for any given RFID system. It must also be able to identify novel ways in which the security of that particular system can be improved.

- **How can the tags and readers of a system securely communicate** – Because of the unsecure broadcast mechanism used for communication between tags and readers, RFID communications can be subject to a number of attacks including man-in-the-middle, DoS, eavesdropping, replay, forward and backward tag tracking, reader and tag spoofing, tag cloning and transmitted data corruption [24]. Hence, the protocol used for tag-reader communication must be secure against these attacks and must provide a large number of different security functionality. In particular the protocol must be able to first authenticate the tags and readers, communicate data between them in a secure manner. It should also carry out this functionality with the minimal resources available on low cost RFID tags.
How can an RFID system be secured against tag based malware – Recently the possibility of tag based RFID malware infecting RFID systems was proven. These RFID malware, which are based on SQLIA, has the ability to spread extremely fast. And like typical malware, they also have range of detrimental effects on the infected systems ranging from loss of data and denial of service to compromise of private data and systems [26]. Therefore the defence mechanism must be able to detect the presence of tag based malware and prevent its propagation into the backend databases. In particular it must focus on carrying this task out at high throughput rates to ensure that the minimum read rates for large networked systems can be maintained and it must ensure infected tags are identified so they can be cleaned.

1.4 Methodology

The proposed research will be carried out based on the experimental computer science method. This method has two main requirements that must be met of any research project: proof-of-concept and proof-of-performance.

To demonstrate the proof-of-concept a number of steps were carried out for each research contribution. We first reviewed the existing work in a number of RFID performance and security areas to identify weaknesses and gaps and formulate open problems. We then modelled the systems in question and designed systems/solutions that solved the open problems we identified. Because we had modelled the system in question beforehand we were then able to use a number of different methods such as case studies and comparative analysis to demonstrate the validity of our solutions in the context of the system we were trying to improve.

The next step in this methodology is demonstrating proof-of-performance of these solutions. This was done by first designing the more technical details of each of the solutions such as the necessary algorithms, protocol descriptions and process maps. Then those technical details were used to carry out various evaluations in such areas as accuracy (for malware detection) and computational overhead (security protocol) and comparative performance evaluation (RFID architecture). Also in some cases other informal methods such as security and system analysis and case study evaluations were also used to further evaluate the accuracy and correctness of the proposed solutions.
1.5 Major Contributions and Significance

The major research contributions of this thesis are as follows:

1. **Comprehensive review and critical analysis of current research literature in a number of different areas associated with networked RFID performance and security including:**
   a. The currently existing architectures for developing networked RFID systems.
   b. Security frameworks and classifications that can be applied to networked RFID systems.
   c. Security protocols that have been proposed for use in networked RFID systems.
   d. Malware detection techniques (focusing on SQLIA detection and prevention) which can either be directly applied or modified and applied to protecting RFID systems from tag based malware.

2. **Proposal of a modular and extensible networked RFID architecture which uses P2P technology and is optimised for large scale networked RFID systems and their specific requirements.** The comparative analysis carried out shows that the proposed architecture is better than existing architectures in a number of key areas when used for developing large scale networked RFID systems.

3. **Creation of a comprehensive networked RFID security framework composed of three main components: threat model, attack model and security model.** The security framework enables the methodical identification of threats and attacks possible on RFID systems, the core components and security concepts that must be maintained and the full security functionality required to protect it. The evaluation of the framework was carried out by applying it to real-life system.

4. **Proposal of a secure RFID communication protocol that can be used for communication of tag data other than the tag identifier to readers.** The security analysis proved that the proposed protocol provides most of the required security functionality and is secure against a number of common attacks on RFID systems. The performance comparison showed that its performance was comparable to other protocols proposed for RFID systems and that it can be implemented on resource-constrained RFID tags.
Proposal of a policy based malware detection and prevention system for RFID systems. Based on existing methods developed for Web-based SQLIA defence our solution has been custom-tailored to work in an RFID system environment and secure it against tag-based malware. The security evaluation proved that it can provide security against all currently known types of tag based RFID malware. It was also shown that the proposed system is relatively simpler and less complex than other comparable security systems as it uses simple string comparisons rather than using more complex methods such as parse tree analysis on NDFA.

Overall the research carried out and presented in this thesis is significant because it can be used to enhance the performance and security of large-scale networked RFID systems used by multiple independent partners while still ensuring that the system performance is not affected too negatively.

1.6 Thesis Organization

The thesis consists of seven chapters in total. These consist of 1 literature review chapter, 4 research contribution chapters and an introduction and a conclusion. The framework of this thesis is as follows:

Chapter 1 – Presents a brief introduction of RFID technology and its improvement over time. We then present an overview of some of the current issues and open research questions in the area and discuss the motivation for addressing those issues and answering those open questions. We then present and discuss the major research contributions of this thesis and their significance.

Chapter 2 – Introduces the background and system model for the research presented in this thesis. We first introduce global supply chains and how RFID can be employed in managing them. We then present the basic model for highly networked RFID systems with multiple independent partners. Next we present the literature review for the research carried out in this thesis. The literature review includes the following:

(1) An in depth analysis of the weaknesses of a number of existing networked RFID architectures in context of the requirements of highly networked RFID systems.

(2) A review of traditional security frameworks and RFID security frameworks and classifications identifying their weaknesses and omissions.
(3) A review of some of the main RFID security protocols proposed in the recent years. Here we focus on both protocols using traditional cryptographic techniques and ultra-light-weight techniques and identify the common weaknesses that led to their failure.

(4) A review of traditional SQLIA detection and prevention methods and their weaknesses in context of securing RFID systems from tag based malware attacks

Chapter 3 – In this chapter we propose an RFID architecture framework specially designed for developing and implementing networked RFID systems. The proposed architecture uses P2P technology for both data lookup and data sharing. In building the architecture we take into account the specific features of networked RFID systems and optimize it for them. We then carry out a comparative analysis of our architecture and existing architectures and demonstrate that our architecture has a number of advantages and improvements over existing solutions.

Chapter 4 - In this chapter we develop and present a comprehensive networked RFID security framework. The proposed framework is aimed specifically at networked RFID systems and takes into account a number of key factors such as threats and their effect on the system, the attacks possible on the system and how they can be classified, and the key components that must be secured and the security concepts that must be preserved in each component. The developed framework presents a methodical manner in which to analyse a networked RFID application and identify its security vulnerabilities and requirements. We then evaluate the framework using a real life RFID system and identify some key vulnerabilities of that system and novel ways in which those vulnerabilities can be eliminated.

Chapter 5 – In chapter five we develop and present a hybrid RFID security protocol. The protocol allows the secure communication of tag data other than the tag identifiers which is a first to the best of our knowledge. We also present two modifications to the protocol: one that increases security at the cost of decreased performance and one that increases performance at the cost of decreased security. The security evaluation demonstrates that the proposed protocol is secure against a large number of common attacks as well as providing most of the required security functionality of networked RFID systems. The performance comparison shows that the
protocol is comparable with recently proposed ultra-light-weight protocols while being more secure.

**Chapter 6** - In chapter 6 we develop and present a policy based dual pronged tag malware detection and prevention system for RFID systems. The proposed approach consists of a tag data validation and sanitization method and a dynamic query verification system. The proposed technique provides security against all currently known SQLIA attacks possible on RFID systems and therefore against all currently known malware possible on RFID. Because we use a simple string comparison method for the dynamic query verification our system is a lot more efficient than other similar techniques while being similarly or more secure.

**Chapter 7** – Summarizes the major findings and contributions of the thesis and provides some suggestions for interesting future research directions in this area.
Chapter 2

Literature Review

Out of all the different types of RFID systems the most complex ones are very large scale networked RFID systems spread over a large geographic area which are used by multiple different partners [2]. This type of RFID systems are widely utilized in RFID enabled global supply chain management systems. But, while the advantages of using RFID in such an environment are numerous there are also a large number of performance and security issues inherent in such RFID systems due to their complexity.

In this chapter we will first explain what a global supply chain is and how they normally operate. We will then describe the basic conceptual model of a very large scale networked RFID system. We will also clearly differentiate it from standalone RFID systems by identifying the differences between these two types of systems. Next we will present some literature reviews that identify some of the open areas in the field of networked RFID security and performance. The first literature review is an analysis of the current RFID architectures which identifies their weaknesses when used to develop very large scale networked RFID systems. Next we will present an analysis of general security frameworks and RFID security taxonomies in context of using them for identifying and securing large scale networked RFID systems. Then we present a critical review of some of the recently proposed RFID security protocols. Finally we analyse and present the weaknesses of current SQLIA detection techniques when it comes to securing RFID systems from tag based malware attacks.

2.1 RFID in Global Supply Chains

A supply chain (Figure 2.1) is a system of organizations, people, technology, activities, information and resources involved in transforming raw materials into end products and
selling those products to the end consumer. Hence supply chain activities transform natural resources, raw materials and components into a finished product that is delivered to the end customer [27]. Global supply chains, which are spread over multiple continents and have billions of items flowing through them each year, can have as many as tens of thousands of storage locations and hundreds of independent partners. The geographic size and the number of partners, stores, products and storage location involved in global supply chains means that sharing information in real time and in detail is of paramount importance for the supply chain to be responsive to customer needs and demands [11]. By attaching RFID tags to the items and associating those tags with information stored in the backend databases, supply chain partners can gain a number of advantages for supply chain management over traditional systems that employ bar codes [28].

The best example for a truly large scale networked RFID system is the global supply chain management systems being deployed by large retailers such as Wal-Mart [8]. The RFID systems in this case spans all the way from the initial raw materials manufacturer to the final retailer and in some cases even continue past sales. When raw materials are first generated and packed those packages are attached with an RFID tag. The tags can be placed on containers, pallets, cases or even on individual products. Then the manufacturer stores static information such as date, batch number, price and expiry date on either the tag itself or the backend database. The manufacturer may also associate transaction data such as the buyer and delivery addresses with the specific RFID tag identifier as they become available and store them on the backend database [29].

When the logistics get the tagged package they use that data to properly it. They also update the data in their backend database with details such as unit’s current location.
and the shipment and transport truck it’s been attached to. This kind of “transactional data” is automatically generated and will be constantly updated in a large number of different RFID data repositories as new readers pick up that specific RFID tag and the middleware associates new business processes with that item [27]. By tagging the products as they travel along the supply chain the partners are able to track all the products in real time wherever they may be. In addition the automation possible with RFID tags due to their wireless communication and extended range means that a great number of business processes can be made more efficient by their use. The possibility of generating large numbers of “transaction data” automatically and sharing it with the partners along the supply chain dramatically increases the visibility of the full supply chain for all involved partners. It also allows the leveraging of more real time information to manage and streamline the supply chain better [28].

When the final product producer receives that item he unpacks it and uses the raw materials contained inside to create products. These products each have new RFID tags but those tags are also linked with the tags of the raw materials packages they were developed from. Once again the producer stores static information on either the tag or the back end and then the final products starts moving along the supply chain moving from each partner to partner. At different points in this chain each partner generates more and more transactional data concerning the products attached to each tag and stores them in his personal RFID repository. Most of time each partner is only aware of the partner directly up and down chain from them [30]. But, while they may not know the other partners they still need to access all the transactional data that was generated by each partner if the full power of the RFID system is to be leveraged. Therefore they need to be able to locate all the data repositories that contain information about any given RFID tag. This is currently where the EPCGlobal comes in.

The EPCGlobal provides services that allow partners to identify the data repositories anywhere in the world that contain information about any given tag [31]. Different companies use different business applications and place more importance on different types of data. Therefore the data stored in each repository must not only contain as much information as possible but it must also be able to cater to the data format requirements of a large number of different business applications [2]. Because RFID tags and EPC numbering have been conceived with the idea of item-level tracking
rather than product-level tracking it is possible to generate and share information at a much greater level of detail than is possible with barcodes [27].

2.2 Networked RFID System Model

At its very basic level a networked RFID system consists of collection of RFID readers, tags and business entities who work together to generate and share information about the physical objects attached to the RFID tags. This is done in real time to improve and speed up the performance of day to day business processes [32]. A typical networked RFID system is depicted in Figure 2.2.

RFID based systems are generally composed of a number of main components including [32]: (1) The RFID tags which are affixed to the products and which identify each product uniquely, (2) The RFID readers or interrogators that read and write the data stored on the RFID tags (3) The RFID middleware that manage the RFID equipment, filter the data and interact with the enterprise applications and (4) the backend storage system which stores the EPC data generated by the system. These components are explained in further detail below.

![Networked RFID architecture](image)

Figure 2.2: Networked RFID architecture
2.2.1 RFID Tags

The most important component of a networked RFID system is the RFID tag. The items in the system are tagged with an RFID tag when they are first manufactured. RFID tags used in networked systems are typically low cost passive tags with update functionality. These RFID tags typically comprise an electronic memory module, a logic module, and a communication antenna. The memory module of the tag can consist of read only memory (manufactured with data which can never be changed), write once memory (manufactured blank and can be written on once) or read-write memory (manufactured blank and can be written on and updated many times). The logic module of typical RFID tags consists of between 3000 to 10000 gates. The tags can be powered with passive power (power is generated by the radio signal transmitted by the readers), or a semi-passive power source (the power for broadcasting is supplied by the reader while the power for other functions is supplied by a dedicated battery), or an active power source (a dedicated battery supplies all the required power) [33].

RFID tags are what links the electronic RFID data stored in the system to the physical object they are attached to. The tag will always hold a unique identifier which allows the system to associate data stored in the backend databases with the attached item. When a new RFID tag is entered in the system the manufacturer of the item allocates its identification number based on a global hierarchy. Because RFID tags use an insecure channel (radio waves over open air) to communicate the reader and tags need to provide the security functionality required to ensure the security of the communications. Additionally because the RFID tags are physically attached to objects they are relatively less secure than typical data storage devices in networked IT systems. In addition, most large networked RFID system need to use very low cost RFID tags. Therefore the amount of resources available on those tags (both performance and power-wise) is considerably low and it is nearly impossible to implement the standard cryptographic security measures employed in IT networks on those tags [22]. But, because RFID tags contain a memory module that contains sensitive data it’s still imperative that the data stored on the tag be secure from both logical and physical access. Therefore RFID tags must have enough resources on board to provide some security functionality. In some current systems they also need to hold additional data
such as brand, product, expiry date, price and ingredients of the object they are attached to [33].

### 2.2.2 RFID Readers

RFID readers are used to read and write data to and from RFID tags. They are composed of three main parts: the receiver, transmitter and controller [33]. A RFID reader needs to manage the large number of simultaneous transmissions it may receive and respond to them [11]. In systems that employ passive RFID tags the readers must also provide power to the tags [34]. The readers also need to read the identifier and any additional data stored on any authenticated tags within reader distance and update the tag data when necessary [35]. The range of an RFID reader depends on the frequency of the radio wave used and the power used to generate its radio waves. There are four main reader frequency ranges for RFID systems: Low Frequency (9-135 KHz) with a reader range of only few centimetres, High Frequency (13.56 MHz) with a reader range of 1cm to 1.5 meter, Ultra High Frequency (0.3-1.2GHz) with reader range of up to 15 meters and Microwave (2.45-5.8GHz) with a reader range of up to 80 meters [33]. In systems employing passive or semi passive tags the tag to reader communication distance is typically much shorter than the reader to tag communication distance because the amount of power the tag can harvest from the reader’s signal is very small. When an authorized reader comes in contact with an RFID tag it first retrieves the identifier of the tag. That identifier and any other data stored on the tag is then passed on to the RFID middleware of the system [22].

### 2.2.3 RFID Middleware

The RFID middleware is the heart of the RFID system and is arguably the most complex component of the system. It needs to carry out a number of different tasks to ensure that the overall RFID system functions correctly [36]. The middleware cleans and collects the data received from RFID tags as low cost RFID tags are notorious for bad and false reads. The middleware also needs to carry out most of the security tasks to ensure the integrity and authenticity of the tags and the data received from them [21]. In systems employing the EPCGlobal architecture the middleware translates the EPC and data retrieved according to the tag data specification as data is stored in binary format on the tag [13]. In most RFID systems multiple readers are set in an overlapping
formation to ensure full coverage of the read area. Therefore the middleware needs to also aggregate, filter and format the data streams from multiple different RFID readers so they are in the format required by the enterprise applications. The middleware also generates transaction data based on business events as pre-defined by the system developers and end users [37]. Therefore, when it receives tag data from readers the middleware first filters it to ensure that they are not false or bad reads. Then any security functionality is carried out to ensure that the data is safe to be used. The middleware then locates any additional data locations using a lookup service and retrieves any required information from those partners’ data servers. It then uses pre-set business rules to generate transaction data, associates that data with the identifier and stores that data in the system database. The middleware also communicates with other applications that require RFID data and retrieves that data, either from the local database or from partners' databases, and forwards it to them. This architecture is very different from normal networked systems which typically do not have a dedicated middleware component, as different partner applications communicate directly with each other as required [2].

2.2.4 Lookup Service
The lookup service is what allows different independent partners to locate and communicate with each other. In some very large networked RFID systems all the partners of the system may not even directly know each other or of each other’s existence [11]. Therefore networked RFID systems require a method with which the partners can locate all the data stored in the overall system concerning a specific tag. This service is normally run by an independent entity that is trusted by all partners. In EPCGlobal systems this service is provided by the EPCGlobal ONS service [13]. The lookup directory needs to have the identification numbers of all the RFID tags of the system. It also needs to know the location of all the data concerning each specific RFID tag and the details on how to contact that data server.

2.2.5 Data Storage
The RFID repository is where the data concerning the tagged objects are stored. In some cases this may be one physical server or database while in other systems this may be multiple dependent or independent databases or servers [2]. In some systems the data
servers are only accessible by the RFID middleware. Internal or external applications which need to access that data need to do so through the RFID middleware. In other systems these servers can be directly accessed by external partner applications [13]. Either way the RFID repository needs to allow more access to its data by external entities and programs than is typically allowed in IT systems. In addition the data stored in the RFID data servers must be stored in as granular a form as possible to preserve as much of the information as possible. This is because different partners need different information from the same set of data. Therefore it is up to the querying middleware or the business applications to extract the required information from the raw data retrieved and format it in a way that can be used by the business applications [2].

2.2.6 Business Applications

The business applications use the data received from the RFID system to carry out existing business processes [38]. They may also update the data stored in the repository or the RFID tags. These applications are mostly pre-existing ones that have been modified to integrate with the RFID system and use the information supplied by the RFID middleware [39]. They retrieve information as required by either going through the middleware or by directly communicating with the external and internal RFID data repositories. Therefore, in networked RFID systems the business applications must be modified to directly communicate with either the middleware of the system or other partners RFID repositories. Unlike in typical networks where the different partners have data format and storage standards and agreements, the business applications using a networked RFID system may be required to use data stored in a number of different formats and granularities by different partners [2]. The retrieved data is then used to automate, improve and streamline existing business processes.

2.2.7 Communication Network

As in any networked system all these different components have to be connected by a communication network. The network used in RFID systems can be divided into two main parts: the internal network and the external network. The internal network (shown in the green lines above) connects the components of a single partner together. This part of the network typically consists of a LAN or WAN and is protected from outside intrusion by the partner’s firewall and intrusion detection and prevention software. The
external network on the other hand connects the components of different partners as well as connecting the middleware of the system with the centralized lookup service [40]. This part of the network is generally implemented over the internet.

2.3 RFID Architectures

For the purpose of this thesis we define “system architecture” as follow: architecture is a description of a system, organized so that it explains the rationale behind the overall system structure which compromises different components. It also explains the properties of those components and the relationships and interactions between them. Therefore the architecture of a system describes the different components and how they will work together to implement the overall system. Cohesive, well-developed and accepted system architecture is very important to not only ensure that the system works but also to ensure compatibility between systems that are developed separately.

2.3.1 The EPCGlobal Architecture Framework

The EPCGAF is “a collection of hardware, software, and data standards, together with core services that are operated by EPCglobal, its delegates or third party providers in the marketplace, all in service of a common goal of enhancing business flows and computer applications through the use of Electronic Product Codes” [13]. The architectural framework designed by the EPCGlobal has on purpose been developed at a high level thereby ensuring that different vendors and organizations have as much freedom as possible when developing applications based on the framework. It can be used in creating the architecture for a large number of different types of RFID applications while ensuring interoperability and the possibility of data sharing with other organizations that use the same framework when developing their RFID enabled applications [13]. Currently this is the accepted architecture standard when developing any RFID enabled systems. In the following section we will explain in summary some details of the EPCGAF. For full information on it and its use please refer to [13].

2.3.1.1 EPCGlobal Data Storage and Recovery

The EPCGAF is built around the concept of a globally unique Electronic Product Code. This is a single identification number which allows the unique global identification of the object it’s associated with. To handle the large number of possible EPCs, various
EPC manager organizations are allocated blocks of EPCs which they then allocate to various other organizations as needed. There can primarily be 2 types of data associated with an EPC:

**Static data** – Static data is created at the birth of the object and it does not grow or change over the life of the object. This type of data is normally created and stored by the manufacturer of the object and is of limited size. E.g.: product name, price, expiry data, manufacture data, etc.

**Transaction data** – Transaction data is data created during various business processes by various organizations that handle that object. Over the lifetime of the object the amount of data created about it can grow immensely. E.G.: EPC A was seen at warehouse B of organization C at a specific time. 250 counts of EPC class D are available at warehouse C of organization D.

Due to the immense amount of transactional data created, using a centralized approach to store the data is not viable. Storing the data at the organization it was generated at is possible, but there has to be a way to allow other organizations to locate that data. The Object Name Service (ONS) is a hierarchal look-up service, that’s based on the internet DNS service. It provides a means for looking up a reference to an EPCIS service or other service associated with an EPC. The typical lookup for EPC data retrieval is shown in Table 2.1:-

<table>
<thead>
<tr>
<th>Lookup Step</th>
<th>Lookup Service Employed</th>
<th>Who Maintains the Service</th>
<th>What Data is Retrieved</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Root ONS</td>
<td>EPCglobal</td>
<td>Address of Local ONS for given EPC Manager Number</td>
</tr>
<tr>
<td>2</td>
<td>Local ONS for given EPC Manager Number</td>
<td>Holder of EPC Manager Number</td>
<td>Address of EPCIS Service for given EPC Class</td>
</tr>
<tr>
<td>3</td>
<td>EPCIS (manufacturer)</td>
<td>End user responsible for commissioning EPC</td>
<td>Commissioning data about the EPC</td>
</tr>
<tr>
<td></td>
<td>EPCIS (partner)</td>
<td>Organizations that handled the object associated with the EPC and generated Transactional data</td>
<td>Transactional Data about EPC</td>
</tr>
</tbody>
</table>

The ONS takes an EPC as input, and produces as output the address of an EPCIS service designated by the EPC manager of the EPC in question. The root ONS service is
run by EPCglobal and contains information on how to locate and contact the ONS services run by EPC manager organizations and other organizations. The local ONS contains look-up information that allows an organization to locate and communicate with all the EPCIS of organizations that have data about any given EPC. The EPCGAF has number of weaknesses including scalability and availability issues, SPOF, unnecessary duplication of data retrieval and formatting and centralised data locations and lookup services. For a full discussion of these weaknesses refer to Section 2.3.3.

2.3.2 Other Networked RFID Architectures

Over the years a number of architectures have been proposed for networked RFID systems. In this section we will discuss some of them and identify their weaknesses.

One of the first peer-to-peer based RFID architectures were proposed in [41]. This architecture uses hybrid method for peer resolution. The information discovery is done using the traditional EPC based ONS while service discovery is done using a DHT based system. The proposed architecture removes some of the bottlenecks and scalability and availability issues associated with the EPCGAF. Unfortunately this approach has one major weakness. The peers are networked by chain linking the address entries. If any participants are not available or their services fail the chain would break and no information from one side of the broken chain could be retrieved by peers on the other side of the break. This meant that the solution was not feasible for global supply chain management systems as the chain in that kind of system would be very long and a break would create major issues for all partners.

In [23] the authors present a peer-to-peer based alternative to the EPCGAF. It is also based on the DHT model of data lookup used in a lot of modern peer-to-peer networks. This architecture allows for the use of any type of tag identifier to allow greater interoperability with other architectures. The hash values of the tag identifier, which is also used to identify information sources about that tag, are mapped to a distinct location in the network where the participants can retrieve the entry directly. The actual data lookup is carried out using either a direct search or indirect search. In direct search the object identifiers are used as keys and looked up in the DHT key space. For indirect searches indices have to be created and updated periodically. But all information associated with the tags remains in the participant’s local system and other partners all retrieve the required data from that one location. Once the data service is
located using the DHT, the required information can be retrieved via a SOAP or EPCIS service using the SOAP internet interface or the URI of the service. The proposed architecture has a number of improvements over the EPCGAF including greater scalability, because the data look-up is done in a distributed manner. It also has better interoperability as any identifier can be used rather than just the EPCIS. But the system still has issues. The main issue with this architecture is that the partner data sources which contain the data generated by that partner acts as a single server distributing all of that data to all other partners. This creates a single point of failure for that data service as well as scalability issues. In addition it still holds all the duplicate data lookup retrieval and formatting requirements that plague the EPCGAF which requires partners to unnecessarily carry out the same work multiple times. The system decentralizes the data lookup ensuring that the EPCGlobal cannot abuse its power, but it requires additional security measures to ensure the authenticity and access of partners accessing the data locations.

In [42] the authors present another peer-to-peer based RFID resolution framework which is based on the original proposal presented in [41] but without chain links. In this approach instead of using hash values of the identifier as keys the system uses the EPC company prefix number (CPN), which is a unique identifier issued by a central authority, to map the keys to the nodes which contain data about it. Node IDs are used as follows: for companies with a CPN their CPN is used as the node ID followed by a random number, for companies without a CPN the three digit country code for the country they originate from are used followed by a random number. The nodes in the system are then arranged in a logical circle based on their node ID. Because the first part of the node ID is based on the country this ensures that the nodes in the logical network are arranged with physically closer peers that are logically closer to each other as well. The data resolution is done by going along the circle till a peer with the required information is found. To make the process more efficient another list containing the peer’s addresses logarithmically is also used. Due to its similarity to the previously discussed architecture this architecture also has very similar strengths and weaknesses. While its scalability and resistance to failure is higher than the EPCGAF due to the use of a peer-to-peer address resolution it still has scalability and availability issues at the actual data services. In addition, it also has issues with duplicate data look,
retrieval and formatting as retrieved data is discarded once it’s used and must be re-retrieved when needed again.

In addition to the architectures meant for large scale networked RFID systems there are also a number of other RFID architectures which are meant for other types of systems but have strengths that can apply for networked RFID systems.

In [43] an interesting architecture is proposed which can be used for tracking the position and movement of RFID tags in small environments. It uses a tag identifier swapping mechanism to protect the location privacy of the tags. This kind of architecture is useless in large scale networks spread over a wide geographic area as it’s meant for use in small areas. But it still has its uses in global supply chain management systems in areas such as warehouse management. By implementing this kind of architecture in the confines of a warehouse the users will be able to track and locate the location of individual tags to a very small area enabling the easy and efficient access to that tag. It may also help identify inefficiencies in the physical layouts of the warehouses and in the business processes that are carried out therein by using the flow of the tags to identify potential bottlenecks in the physical goods flow.

2.3.3 Weaknesses of the Current Solutions

Because the EPCGlobal architecture (Figure 2.3) is highly generalised and not aimed specifically at large scale networked systems with multiple partners there are a number of issues and weaknesses it presents when building these types of systems. In addition, as discussed above, most of the other architectures that have been proposed also contain a number of weaknesses. By analysing the EPCGAF and the other architectures we identify the following common weakness in them, in the context of the environment and requirements of global supply chain management systems [27].
Figure 2.3: EPCGlobal RFID architecture framework
2.3.3.1 Low Failure Tolerance
System availability and security is a major concern for networked RFID systems. The structure of the EPCGlobal architecture framework introduces a number of Single Points of Failure (SPOF) to the system. The most critical point that could cause complete system failure is the ROOT ONS run by EPCglobal itself. The ROOT ONS is the starting point in which the lookup process for all data retrieval transactions is initiated, and its failure would affect all applications of all organizations that use this EPCglobal core service [13]. Because the local EPCIS of each partner is the only method in which external partners can retrieve transactional data generated by events happening at a partner’s reader location these also act as SPOFs [44]. The peer-to-peer architectures reviewed has improved fault tolerance by not relying on a single centralised service for address resolution. But they still have low fault tolerance at the data services as the data from a particular company is only available from a single service.

2.3.3.2 Non Optimal Data Formatting and Aggregation
The manner in which data formatting and aggregation is done in the EPCGAF and most of the other architectures is sub optimal for large scale networked systems [2]. Because different partners use different applications the data that is required by those applications are also different in terms of format and aggregation level. Therefore most of time the data recovered from another partner's data service needs to be filtered, aggregated and formatted before being used by the business applications that use it [45]. As most of the architectures require that data be retrieved from the partner's data service each time it’s needed (no local caching is specified for retrieved data) this requires the application using that data to filter, aggregate and format the same data multiple times whenever it’s required [13]. This unnecessary duplication of work needlessly increases the load placed on the local middleware and the network.

2.3.3.3 Low Scalability
The proposed architectures are susceptible to scalability issues due to the method in which the data is stored and retrieved [2]. All the transaction data collected by a single organization about all the RFID tags in the network are stored and distributed by a single data service belonging to that organization [13]. Therefore all the other
organizations need to contact and request that single service for any and all data generated by that organization about any and all the RFID tags in the network. As is obvious, when the network grows (it is common for large scale networked RFID applications to have millions of tags with hundreds of different locations requiring data) the amount of data requests received by the services of any given organization will grow radically [46]. This situation, known as “message explosion”, will eventually overwhelm the data service’s ability to receive and send messages and will cause the single data service to be overwhelmed with the amount of computational work needed to serve those data requests. The duplication of data retrieval that needs to be carried out by business applications needlessly exacerbates this problem. In addition the ONS service provided by the EPCGlobal is of very big concern when it comes to scalability. Because it’s the sole provider of all address resolution requests by all EPCGlobal users around the globe its scalability will become an issue very soon as RFID adoption continues at the current high rate.

2.3.3.4 Difficult Integration with Existing Systems

Another weakness in the architectures (specially the EPCGAF) is the amount of redevelopment required for existing systems to work in conjunction with the new RFID system [2]. In the EPCGAF each individual business application is in charge of locating, retrieving and manipulating external data to gain the required information [13]. This means that all legacy applications must be redeveloped to do the data lookup as defined by the EPCGlobal using the core lookup services they provide. In addition those legacy applications must also be able to communicate with the EPCIS that has the required data, retrieve it and then use that data to extract the required information. This requires large scale redevelopment and modification of each and every existing business application and software that uses or generates RFID data [3]. Most other architectures also require extensive modification of existing applications and systems to properly work as the business applications are in charge of the data lookup and retrieval and formatting rather than a middleware.

2.3.3.5 Security Concerns

Another one of the main issues of the EPCGAF is that it does very little to address and supply the huge amount of security requirements for a truly secure networked RFID
application [17]. By its own admission the security features that are provided by the EPCGAF is use of pseudonyms to make unauthorized tag tracking more difficult, memory locking – which seriously diminishes the power and functionality of the application employing those locked tags, and kill commands which are useless in a networked, multi-organization environment. Additionally the ONS used in the EPCGAF is based on the DNS services used on the world-wide-web. Therefore it has inherited all the security weaknesses inherent in the DNS service such as vulnerabilities to cache poisoning and DDoS attacks. The peer-to-peer based architectures remove the DNS weaknesses but bring with them their own security concerns such as proper distributed authentication and access control of partners due to a lack of a central authoritative partner [23].

2.3.4 RFID Architecture Requirements
Because of the above shortcomings in the currently existing RFID architectures they are not the best solution to be used for developing systems such as RFID enabled global supply chain management systems. The key requirements when developing a large scale networked RFID system is: (1) high scalability and expandability, (2) interoperability with a number of different systems, (3) easy integration with existing systems, (4) high failure tolerance and (5) increased data lookup, retrieval and formatting efficiency [2, 23]. Therefore a new architecture framework needs to be developed that meets these requirements. Additionally, because the current standard for RFID systems developments is the EPCGAF it is imperative that any new architecture be fully compatible with systems developed using the EPCGAF to ensure global accessibility. In chapter 3 of this thesis we therefore present a “P2P RFID Architecture Framework” which has been specially developed for creating large scale systems with multiple independent partners with pre-existing business connections and are spread over a large geographic area. This architecture removes/minimises most of the weaknesses inherent in other networked RFID architectures when used to develop this type of RFID system.

2.4 Current Security Frameworks
A security framework can be thought of as a document that allows a person to methodically and comprehensively analyse a system and identify the threats and attacks it faces, the vulnerable security components and the security requirements that are
required to secure those vulnerable components against those threats and attacks. While all security frameworks have some common aspects a good security framework needs to be specifically tailored to the system requirements and features of the type of system it’s aimed at securing.

A majority of the current existing security frameworks are aimed at general networked systems such as the one presented in [47, 48]. But the architecture of massively networked RFID applications is significantly different from typical IT networks and standalone RFID systems. These frameworks do not take into account these differences and therefore cannot be fully applied to networked RFID systems. Overall currently no complete security framework has been developed specifically for networked RFID systems with multiple independent partners [17]. But there exists a number of generic security and network security frameworks as well as some classifications of RFID attacks and defences that present some interesting insights in to this research area. In the following section we will examine some of those papers and analyse their weakness in context of the security of networked RFID systems.

2.4.1 Typical Security Frameworks

The ‘Framework for ensuring network security’ [48] has been built around a threat model developed using a modified version of the Confidentiality, Integrity and Authentication (C.I.A) threat model. It uses a layering technique to ensure that all areas of the system’s security have been covered whether it’s at hardware level or at application level. It also contains a list of the security requirements needed to secure an IT network and a number of cross analysis tables mapping the security requirement, the threat model and the different layers of the network to identify what needs to be done at each layer to negate each threat identified in the threat model. The framework also compares the developed framework with some other popular security frameworks. The main weakness in this framework is that it’s developed for typical network systems and not networked RFID systems. Therefore this framework overlooks some threats specific to networked Multi-entity RFID systems such as tag data ownership, and tag access control as well as some threats that all RFID systems face such as tag privacy. Also some of the preventive measures discussed are not possible in an RFID system due to the differences in its architecture compared to a normal wired IT network.
The ‘Integrated security framework’ presented in [49] is also of the same nature but even more generic framework than the one presented in [50]. This framework has been developed to apply to all types of wirelesses networks ranging from complex and high powered cellular networks to basic RFID networks employing low cost RFID tags. While the authors discuss the basic security concepts that apply to all wireless systems (Confidentiality, Authentication, Integrity, Availability and Non-repudiation) it does not mention any of the RFID specific attacks such as data leakage, cloning or tag tracking. While the author does discuss some potential security solutions such as public key cryptography they are not suitable for RFID systems due to the high overhead they require. The RFID security measures the authors present are three very high level generic approaches such as “Non-Cryptographic Schemes” and “Lightweight Cryptographic Schemes” with no further details or explanations and are therefore not of much use when it comes to deciding how a specific RFID system should be secured.

2.4.2 RFID Security Frameworks and Classifications

While there are some RFID security frameworks that are targeted at full RFID systems a majority of them only cover a specific area of the system, such as [51]. Some other frameworks focus on standalone RFID and not at networked RFID, such as [50]. Therefore most of these frameworks are either not applicable to networked RFID systems or they are insufficiently detailed to fully secure a networked RFID system. Table 2.2 shows the comparison of some security frameworks.

In [52] the authors present a ‘Roadmap to solving security and privacy concerns in RFID systems’. This paper identifies a few of the potential threats and attacks possible on RFID systems. Unfortunately the only threats identified are data mining, tag tracking and unauthorised tag reading and therefore are in no way comprehensive. It also fails to mention some of the more dangerous threats such as RFID malware or RFID cloning. The paper then goes on to present some proposed technical solutions such as kill commands, faraday cages and blocker tags. But none of these solutions are new or novel. In addition the authors suggest some policy propositions that can be used to neutralize the identified threats and attacks but most of these are suggestions by other authors and do not contain any new ideas. Overall the list of proposed technical solutions presented is too short and do not afford sufficient protection for a networked RFID application. In some cases the proposed solutions are not viable due to practical
or performance issues (e.g.: using kill commands in a networked RFID system is not viable as it will remove the ability of the partners to use the RFID tags).

Table 2.2: Security framework comparison

<table>
<thead>
<tr>
<th>Threats</th>
<th>Attacks</th>
<th>System components</th>
<th>Customised for networked RFID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Roadmap to solving security and privacy concerns in RFID systems [52]</td>
<td>Partially Partially Partially NO NO YES NO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Framework for ensuring network security [48]</td>
<td>YES YES YES YES YES YES NO NO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Integrated security framework [47]</td>
<td>Partially NO Partially NO Partially NO YES NO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Classification of RFID attacks [51]</td>
<td>YES YES YES YES NO NO YES NO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A Security Framework in RFID Multi-domain System [53]</td>
<td>NO NO NO NO YES Partially YES YES Partially</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RFID Systems: A Survey on Security Threats and Proposed Solutions [54]</td>
<td>NO NO Partially NO YES NO YES NO</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In [50] the authors present “A framework for assessing RFID System Security and Privacy Risks’. The work presented contains a lot of information about the potential security and privacy threats faced by a large number of different types of RFID systems. The paper also tabulates which threats attack which components of the RFID system and also describe the consequence of each threat. It is also novel in that it depicts the security demands of different types of RFID systems based on the metrics of system deployment type and link between tag and physical object data. Unfortunately it makes no mention of some of the threats and issues specific to networked RFID systems with multiple partners such as the need for data ownership, access control and non-repudiation which is a major gap in the work presented. The lack of these requirements means that some types of systems are erroneously shown as having less of a security demand than they actually do. This paper also contains no information about the
security functionality required to defend against the identified threats which means it
can’t be used to actually identify the key security requirements of each of the different
types of RFID. While the framework is suitable for identifying the threats against
standalone RFID systems and the consequences of those threats it’s too simple to be
applied to networked RFID systems and does not have enough information (namely the
security requirements) to be used as a proper security framework for securing RFID
systems.

‘Classification of RFID attacks’ [51] classifies a majority of the currently possible
attacks on RFID systems based on the layer at which the attack is targeted at. The
taxonomy contains all the common RFID threats such as replay attacks, impersonation
attacks and denial of service attacks. It also discusses some of the lesser known attacks
such as malicious code injection and traffic analysis. While this classification is helpful
in securing a majority of RFID systems it still does not look at some of the threats
uniquely present in networked multi-entity RFID systems such as repudiation,
unauthorized data modification by partners and corporate data theft. It is also taxonomy
rather than a security framework and therefore focuses more on identifying and
classifying threats rather than identifying how to secure the system against them.

In [53] the authors present a ‘A Security Framework in RFID Multi-domain
System’. In this paper the authors identify some of the security requirements of an inter
domain (multi-entity) RFID system. Then they go on to present a security framework
that can be used for authorization and authentication in this kind of system. This paper
is novel in the sense that it not only looks at securing the communications between tags
and readers but it also looks at how the communication between the back ends of two
different entities can be secured. Unfortunately this paper only looks at two of the key
security requirements: namely authentication and authorization. All other security
requirements such as data confidentiality and integrity, tag access and tag data
ownership are ignored. Therefore the solution presented is severely limited in the
protection it can offer. It also does not try to identify all the threats and attacks that
RFID systems face or try and classify them in a meaningful manner.

In [54] the authors present “RFID Systems: A Survey on Security Threats and
Proposed Solutions”. Here the authors first identify some key attacks that can be
mounted on RFID applications. But the list is not complete and they do not try to
classify the identified attacks in a meaningful manner. They then present a survey on a large range of possible technical solutions to the identified problems. The lack of identification of the general security requirements/functionality required by networked RFID systems means that it’s impossible for the reader to understand what needs to be done to fully secure one. Therefore while this paper is good in becoming familiar with the security solutions available this does not help the reader verify if implementing those solutions would actually fully secure any given RFID system.

2.4.3 Networked RFID Security Framework Requirements

Large scale networked RFID systems have the following unique features that set them apart from typical IT network systems [17]: (1) While most IT systems are accessible by a single authorized entity, the RFID tags of the system must be fully accessible by all partners in the network [55]. (2) The RFID tag is much more physically accessible by attackers than typical IT components. (3) Low security resulting from the lack of resources available on RFID tags makes the wireless communications of the system highly vulnerable to attackers [56]. (4) The mobile nature of RFID tags makes it possible to invade the privacy of the system without ever gaining access to the communications or the memory modules of the system [57].

Additionally networked RFID systems also have a few main features that set them apart from standalone RFID systems [17]. (1) Networked RFID systems have a number of users while standalone RFID systems have only one user [2]. (2) The tags of networked RFID systems are comparatively a lot more mobile and physically accessible by attackers than the tags of a standalone RFID system. (3) Overall system structure for standalone RFID is considerably simpler than the system structure for a networked RFID system [13].

Therefore when developing a security framework specifically for networked RFID systems both sets of above differences must be taken into consideration. But as the above literature review illustrates there is still no complete and cohesive security framework for large scale networked RFID systems with multiple partners. Therefore in chapter four of this thesis we develop and present a complete and cohesive “Networked RFID Security Framework” that takes into account all the features numbered above and fulfils all criteria in Table 2.2.
2.5 RFID Security Protocols

Like with any networked IT system the different components of an RFID system must communicate with each other. In RFID systems the most vulnerable of these communications take place between the readers and tags of the system [20]. Therefore it is very important that this communication takes place using a secure security protocol. A communication protocol defines the format, the messages and the rules that two components use for communicating with each other. A security protocol is specifically built and designed to protect the communication that takes place from attack by malicious parties. Because a protocol normally defines the syntax and semantics of the communication it’s independent of the exact implementation method. Therefore a protocol can therefore be implemented as hardware or software or both.

The number of security requirements and functionality to fully secure the tag-reader communications of networked RFID systems is very high. Therefore in the past few years, a very large number of RFID security protocols have been proposed for RFID systems. These range from ultra-light-weight protocols in [58-62] to the more computationally expensive protocols employing traditional cryptographic techniques presented in [63, 64].

2.5.1 Protocols Using Traditional Techniques

Because traditional cryptographic methods require high computing resources most of the protocols employing them have proven to be too resource-intensive for use with low cost RFID tags. The protocols proposed in [63, 64] both use tag side PRNG and cryptographic hashing while the two protocols presented in [65, 66] uses CRC and PRNG on tag side. Because most of these cryptographic primitives take well over 5K gates minimum to implement any protocol employing them cannot be used with low cost tags which normally have around 3-4K tags maximum dedicated for security. In addition to the high resource requirement the protocols presented in [63, 64] are also susceptible to tag cloning and confidentiality. While the protocols discussed above are all concerned with providing standard security functionality such as mutual authentication, confidentiality and availability there are some RFID protocols that provide other kinds of security functionality. The two protocols described in [67, 68] are concerned with providing tamper detection for data stored on the tag itself. Both
approaches rely on using watermarking techniques to embed patterns into the tag data allowing any possible unauthorised changes to be detected. Likewise [69, 70] also only provide tamper detection (using other techniques). But because the security requirements of RFID systems include a lot more than just tamper detection these protocols are not sufficient to secure a networked RFID application [17]. Therefore most of these protocols have failed to provide the full functionality required to secure networked RFID applications by either being too resource intensive or not being secure enough.

2.5.2 Ultra-Light-Weight Protocols

Because traditional techniques proved to be too resource intensive a new technique had to be developed to protect RFID applications employing low cost passive tags. As a result Ari Juels introduced the concept of minimalist cryptography which employs only simple bitwise operations in [56]. This protocol however was soon discovered to be fully susceptible to a number of attacks including full disclosure attacks. After this a large number of protocols based on this concept were proposed over the next few years. The EMAP family of protocols [59-61] were specifically built using this concept to require low computing resources. These protocols are based on only using the bitwise XOR, bitwise OR, bitwise AND and addition mod 2m operations. A key divided in several sub-keys is shared between legitimate tags and readers. Both readers and tags use these sub-keys to build the messages exchanged in the mutual authentication phase. Only readers need to generate pseudorandom numbers. The communicated PRNG numbers are then used by the tags when creating fresh messages. These protocols also employed a novel method of updating pseudonyms to remove the traceability problem. But the weaknesses in this family include the use of triangular functions, which made the messages vulnerable to crypto attacks. The large number of different messages exchanged during the authentication phase further allowed the attackers to further exploit this vulnerability to crypto attacks. They also employed bitwise AND and OR operations, which have strongly biased outputs on public messages allowing attackers to use statistical methods to attack the protocol [62]. By exploiting these weaknesses other researchers were able to mount full disclosure attacks on these protocols fully compromising the data they transmitted [71].
The SASI [58] protocol had the same basic phases and used the same method for key and index pseudonym updating as the EMAP family. SASI was developed incorporating a non-triangular function (left rotation – ROT) so as to remove the vulnerability that the EMAP family showed towards crypto attacks. But it still exchanged a large number of messages during the key updating phase and it also employed bitwise OR and AND in public messages. By exploiting these weakness and some additionally vulnerabilities full disclosure attacks were mounted on the SASI protocol [72]. The Gossamer protocol [62] had the same basic structure and used the same method to update the keys and index pseudonyms as the previously discussed protocols. But it also included some new features. In particular they removed any public messages which included OR and AND operations which in turn increase security against crypto attacks. They also included a strong non-triangular function in the message generation and also developed a propitiatory iterative function named Mixbits to further increase security. But the protocol still had one weakness: it still had to send 3 messages, all containing the same secret numbers and keys encrypted using only simple bitwise operations, during the mutual authentication phase. While the protocol has yet to be fully breached it has been shown to be susceptible to DoS, De-synch and replay attacks [73]. It also uses a very large number of simple bitwise operations (more than 50 in total during operation) and employs a proprietary function and requires a large temporary memory area to hold the intermediate values of the messages while the computations are taking place. Therefore this protocol not only requires significantly more resources than the other ultra-light-weight protocols but it also needs additional memory for temporary storage of data [20].

2.5.3 Common Weaknesses in RFID Security Protocols

Most of the protocols which have been proposed in the recent past have had a few key weaknesses which have made them unsuitable for securing networked low cost RFID systems. In developing our protocol we have ensured that those weaknesses do not impact it.

A major weakness in most of the ultra-light-weight protocols [59-61] is the use of multiple messages during the authentication stage. This is necessary to provide integrity verification of those messages. The messages all include the secret keys and have been encrypted using low strength encryption techniques (simple bitwise operations). This
feature of these protocols allows potential attackers to intercept those multiple messages and mount crypto attacks on those protocols. In addition, it also allows the attacker to verify that the data he retrieved using the crypto attacks is correct. A majority of these ultra-light-weight protocols were attacked using this method and it led to the complete disclosure of the tag's contents as illustrated in [71, 73-76]. In our protocol, we use a one-way hash function to provide integrity verification. Therefore, only one message which holds the secret values is broadcast. Because hash functions are strictly one-way this means the only way for an attacker to gain access to the secret values is to carry out a brute force attacker.

Another common weakness displayed in ultra-light-weight protocols is that they broadcast the EPC of the tag during the authentication phase [58-61]. Unlike the Index pseudonym or the keys which are constantly updated the EPC of the tag never changes. The constant nature of the EPC and the poor encryption strength of the bitwise operations allowed data leakage attacks on these protocols. In data leakage attacks potential attackers eavesdrop over a very large number of the tags communications and gradually build the EPC over time. In our protocol we avoid this weakness by always broadcasting the EPC of the tag only in one-way hashed form. And even when the hash of the EPC is broadcast it’s hashed together with another key or PRNG which constantly changes making data leakage attacks impossible.

Some of the protocols also employ bitwise AND and OR in generating their public messages. Due to the nature of these operations they have poor statistical properties which leads to strongly biased results [62]. Therefore the use of these operations in public messages allows the attackers to exploit the statistical properties and mount crypto attacks on the protocol [58]. Our protocol only employs the XOR function which does not have strongly biased output eliminating this weakness.

Most of the RFID protocols employing more traditional cryptographic techniques make the mistake of implementing resource intensive cryptographic primitives such as PRNG, public key encryption or cryptographic hash functions on the tag [65]. Due to resource constraints of low cost tags this makes these protocols unsuitable for securing systems using low cost tags. Crypto primitives that are too intensive for low cost tags include CRC algorithms, PRNG generators, keyed and cryptographic hashes and most encryption algorithms such as SHA-1 and RSA. In our protocol we only employ a
simple one-way hash and simple bitwise operations on the tag side. A simple one-way hash can be implemented with as little as 1.7K gates as shown in [77] and as the standard low cost tags as defined by EPCglobal class-1 generation-2 tags can have up to 3K gates for security functionality [73] our protocol is suitable for these low-cost tags.

2.5.4 RFID Protocol Requirements

The above literature review illustrates a few things about the needs and requirements of security protocols built for networked RFID systems. The use of standard cryptographic primitives such as PRNG, cryptographic and keyed hashes, CRC checks public key encryption makes the protocols too computationally expensive to be implemented on low cost tags. Therefore any proposed protocols, if they are to be practical, must use novel methods or employ low-cost primitives to provide the required security functionality while still requiring minimal resources. The easiest method of this is the use of low cost cryptographic techniques pioneered by Juels in [56]. But the over-use of low strength bitwise operators reduces security strength too much, especially in relation to brute force or crypto attacks. This is mainly because bitwise operations (due to the simplicity of the operation) make messages thusly encrypted susceptible to crypto attacks. This weakness is compounded if multiple messages, built by using bitwise operations on the same “hidden keys”, are sent because this allows the attacker to exploit those multiple messages and the weaknesses of the bitwise operations used on them to retrieve the data hidden in them. In addition over use of bitwise operations also means that protocols gets more and more complex and requires more and more resources as the developers resort to using a larger number of bitwise operations as well as proprietary functions to provide additional resistance against crypto attacks. Finally the number of security requirements of RFID systems mean that any protocol proposed must implement a significant number of those security requirements such as mutual authentication, broadcast integrity and confidentiality and tag anonymity. In addition, they must also provide significant resistance to a number of attacks such as eavesdropping, replay, man-in-the-middle and de-synch attacks.

In chapter five of this thesis we develop and present a “Hybrid RFID Security Protocol”. The approach we have taken in creating our security protocol has led to the development of a protocol that is markedly different from the above protocols. Unlike the above mentioned protocols which use only traditional methods or ultra-lightweight
methods we have used a mix of both. This hybrid method has allowed us to create a protocol that has significantly more security than an ultra-lightweight while at the same time requiring considerable less resource than a traditional security protocol. We have also addressed the problem of how to transmit data other than the tag identifier between tags and readers which has not been addressed before. Our protocol is also more complete as it implements a number of different security concepts and is secure against a large number of attacks compared to other protocols which only implement one or two security concepts or are secure against only a few attacks.

2.6 RFID Malware Detection

In IT and Computer Science malware typically refers to a class of programs that are specifically written to be disruptive to the system that executes it. These detrimental effects can range from disruption or denial of operations, gathering of information that leads to loss of privacy or exploitation, or the opening of unauthorized access to system resources to outside malicious parties. In RFID systems all currently known malware is based on the concept of SQL Injection Attacks (SQLIA) [78]. SQLIA is the process of changing the inputs that are used to build dynamic SQL queries in such a way as to have those queries carryout malicious actions on the database when they are executed. With RFID systems properly built tag based SQLIA malware can even propagate from a tag to a backend database and then back to other tags acting like a typical computer virus [79].

Unfortunately due to their nature most general security tools such as firewalls, virus guards and Intrusion Detection Systems (IDS) are ineffective against SQLIA and therefore SQLIA based RFID malware. This is because SQLIA normally travel through the ports used by regular web traffic (kept open in firewalls), work at application level (making IDS useless against them) and are non-executable malicious code which operate in the confines of the DBMS and carryout processes that are allowed to queries (making heuristic virus checks useless against them). Therefore SQLIA detection requires specialized security mechanisms [78]. There is a large amount of techniques in preventing and detecting SQLIA in web based systems but very little work in detecting SQLIA for RFID systems. The techniques for web systems range from simple methods such as type and length validation to automated systems which use a mix of static and
dynamic analysis to identify all possible input sources and the valid query structures and then match the dynamically generated queries from any of those sources against the identified valid query structures. Overall SQLIA defence techniques for web systems can be classified into two main types [80]: (1) defensive coding practices and (2) detection and prevention techniques.

### 2.6.1 Defensive Coding Techniques

Because SQLIA attacks depend on inputting invalid inputs ensuring the validity of input used in generating a dynamic query will mitigate a majority of the possible SQLIAs. Defensive coding practices are simple SQLIA prevention techniques that revolve around ensuring that all accepted inputs are validated before begin accepted [80]. Some of the best practices when it comes to defensive coding are input type checking and encoding of inputs.

Input type checking consists of ensuring that the input data is type consistent with the expected data for that value [81]. For example fields defined as numbers will only allow digits while fields defined as text will only allow alpha characters. Because most SQLIA depends on inserting special characters or strings into inputs, these types of SQLIA can be blocked by this technique. In the same manner if a max and min length of specific inputs are known beforehand these lengths can ensure that the additional characters have not been entered into the input [81]. As most SQLIAs require that the input is significantly larger than the expected length of the input this can catch a majority of the more complex SQLIA attempts. Another defensive coding technique: Encoding of inputs consist of encoding the input in such a way as to ensure that the database does not mistake Meta characters in the input for keywords, tokens or operators. Injection is often accomplished by tricking the system into accepting special characters embedded string inputs as meta characters [80]. If the system can ensure that all string inputs are recognized as string and not Meta characters, attacks using these methods would fail.

Overall defensive coding techniques still remain one of the simplest and best ways with which to prevent SQL injection attacks. Unfortunately, defensive coding in web systems is prone to human error, mainly due to the fact that most developers do not remember to put in the required validation at all possible input [80]. Therefore if the
human error aspect of defensive coding can be minimized it’s an extremely potent and simple method for preventing SQLIA attacks.

### 2.6.2 SQLIA Detection and Prevention Techniques

Because defensive coding techniques proved to be unreliable in practical web applications, researchers have proposed and developed a wide range of other techniques to detect and prevent SQLIA. Most of these techniques depend on detecting weaknesses in the code that generate the dynamic queries or identifying SQLIA after the queries have been generated. Unfortunately as most of these techniques have been developed specifically to protect web based systems they do not translate well into the architecture used in RFID systems. For example most of the methods below use automated scanning tools such as the ones analysed in [82] to locate and identify possible input sources in web sites. Because RFID do not use web based input scanning is not required nor is it possible for RFID systems.

Black box testing techniques such as the one proposed in [83] uses a web crawler to identify all possible attack points in the web application. Because there is only one possible attack point for SQLIA in RFID systems this type of technique is overkill and not necessary for RFID systems. The new query development paradigms proposed in SQL DOM [84] use encapsulation of database queries to provide a safe and reliable way to access the database. While this technique is secure they cannot be used for existing legacy systems without major redevelopment. They also require programmers to learn a completely new development process based on the query development paradigm which is time consuming. The intrusion detection system presented in [85] use a machine learning technique trained using a set of typical application queries to try and detect SQLIA. This system first build models of valid queries and then uses pattern matching during run time to ensure that all received queries match a valid query model. Because the success of this approach is directly based on the quality of the training set used and a bad training set can result in a system with a large number of false positives and negatives. Therefore in RFID systems it’s better to use manual techniques due to the much lower amount of possible queries.

SQL rand [86] is an instruction set randomization technique which uses a proxy based method which allows developers to create SQL queries using randomized instructions. This technique is based on cryptographic integrity check systems and,
similar to those systems, not only places significant overhead on the system but is also fully dependent on the security of the secret key used in the randomization of the queries. Therefore the use of this type of system would negatively affect the performance of the RFID systems implementing it reducing the overall tag read throughput. Static code checking is a method by which the source code of the application is checked for various weaknesses that make it vulnerable to SQLIA. The main drawback of this approach is that because only static code is analysed it can only spot a limited number of weaknesses. For example the approach presented in [87] can only detect and prevent tautologies while the approach presented in [88] can only spot weakness to incorrectly types inputs. This means that these types of approaches do not provide sufficient security for RFID systems.

Another suggested technique consists of a hybrid of static code analysis and dynamic run time monitoring. In this technique the code is analysed for weaknesses and all legal query patterns during the static analysis phase. Then the identified query patterns are used to validate the SQL queries generated and submitted during the runtime monitoring phase. AMNESIA [89], SQLGuard [90] and SQL-Check [91] all use different query pattern matching techniques. AMNESIA uses a web crawler to identify possible input sources (Hotspots) for the system which makes this approach impossible for RFID systems as they don’t have web inputs. Once all possible hotspots have been identified it uses the Java String analysis library to analyse the string operations carried out in each string of interest and deduct a non-deterministic finite automaton that expresses all possible values the considered string can assume. Because the NDFA are an overestimate this may result in illegal queries being mistaken for legal queries. Additionally the comparison of the NDFA is a complex task. Both SQLGuard and SQL-Check take a different approach. They generate a parse tree to represent legal queries and compare them to the parse tree of the dynamically generated query. The difference is that SQLGuard the model is deduced automatically while the model for SQL-Check is developed by the programmer. Unfortunately both approaches use generated secret keys which must be kept secret and they both require the developer to use special intermediary libraries or to manually insert special markers in the code [92]. Additionally, parse trees, especially for more advanced SQL queries, can be extremely complicated and therefore properly comparing two parse trees is generally a very
complex task. Hence these approaches use considerably more resources than can be justified for use in RFID systems.

In general all current query pattern checking techniques have the following weaknesses in common in the context of their use in RFID systems. (1) un-needed complexity and computational overhead both in generating the legal query patterns and when comparing them with the patterns of dynamically generated queries (2) weakness in the query models due to the automated manner in which they are built resulting in possible false positives and negatives. As the above review shows the techniques developed for web based SQLIA detection and prevention do not work very well in the simpler environment of RFID tag based SQLIA. A majority of the proposed approaches are unnecessarily complex and resource intensive while some others are simply not compatible with RFID systems due the differences in the architecture of the two systems. Finally and most significantly none of these systems are capable of detecting or preventing second order SQLIA attacks.

2.6.3 SQLIA Prevention in RFID Systems

While there has been a lot of work done in detecting and preventing web based SQLIA attacks very little work has been done on the same research for RFID systems. The papers [93, 94] discuss in detail how RFID systems can be subject to SQLIA attacks but present very little work in actually how to detect or prevent them. In [94] the authors mention the possibility of using input validation or attribute code technology to detect RFID based SQLIA but does not elaborate any further. In [93] the authors list some areas the database server administration must take into consideration when setting up the system but no further elaboration is done. In [79] the authors discuss the possibility of infection databases with traditional viruses using RFID SQLIA. But as the bases for infection is still SQLIA prevention of SQLIA will stop this type of attacks. Once again in this paper the authors list some rudimentary steps that can be taken to prevent this type of attack but no further elaboration is done on how or the exact mechanism behind these suggestions. Finally in [95] the authors present a digital forensic system for tracking and identify SQLIA attacks on RFID. This approach is only useful after the fact and cannot be used to either detect possible SQLIA before they are executed or to actually prevent their execution.
2.6.4 RFID Malware Detection Requirements

Overall the key differences in RFID systems and web based systems mean that the solutions developed for web based systems do not translate too well to RFID systems. Additionally, very little work has been done in actually protecting RFID systems from SQLIA or SQLIA based malware. Most worryingly the architecture of RFID systems makes it possible to create and deploy RFID malware based on SQLIA [26, 79]. Therefore it is imperative that a SQLIA detection and prevention method is developed for RFID systems taking into account the unique architecture features that differentiate them from web based systems. To fill this significant gap in the current literature in chapter six we present a “Policy based RFID malware detection and prevention technique”. In developing this system we have not only taken into consideration the unique features of RFID systems but we also ensure that the high throughput required of RFID tag reads can be met by using a simple string comparison rather than the more complex approaches used in the web systems. In addition we have developed and justified an RFID tag data validation and sanitization technique which has the possibility to detect and prevent even second order SQLIA based on the strength of the rules set by the developer.

2.7 Summary

RFID technology, while a relatively old concept, has risen to prominence in the last decade. Even though the worldwide adoption and the benefits associated with RFID has steadily increased there are yet a number of security and performance issues that still hold back both its acceptance and the benefits it offers. In this chapter we looked at four distinct areas of research in RFID performance and security research: (1) RFID system architectures, (2) Networked RFID security frameworks, (3) RFID communication, authentication and security protocols and (4) RFID malware detection techniques. For each of these areas we have identified some key literature and analysed the weaknesses in the solutions proposed therein. In the next four chapters we will present our own contributions to those above mentioned areas of research and discuss how the solutions proposed by us do not contain the weaknesses we identified in the solutions in the current literature.
Chapter 3

P2P RFID Architecture Framework

In this chapter, we first analyse the specific requirements of very large scale networked RFID systems such as global supply chain management systems. Then we develop and present a P2P RFID architecture that is optimized for developing that type of RFID system. The developed architecture is built around a modular middleware system that allows for easy system development and integration. We also employ P2P technologies to improve system scalability and availability. We explain each of the components in the architecture as well as the different middleware modules and the P2P approach we use in detail. Finally we carry out a comparative analysis of the proposed architecture compared to the EPCGlobal and other recent RFID architecture frameworks which shows that our architecture has a number of significant advantages over them.

3.1 Introduction

To realize the maximum benefits of RFID technology in large scale networked environments, the use of an architectural framework, when developing those systems, which fulfils the specific requirements of those systems is paramount. The EPCglobal Architecture Framework (EPCGAF) [2] is the main framework currently used in developing all RFID systems and applications. Unfortunately, the EPCGAF is designed at a high level to allow the development and deployment of a number of fundamentally different systems. Therefore, specialist systems (such as global supply chain applications) based on the EPCGAF will run into a number of issues due to the nature
of those applications and the environment they are deployed in [13]. Additionally, as the literature review shows even the other recently proposed RFID architectures have a number of weaknesses making them unsuitable for use in developing global supply chain management applications. Therefore, it is imperative that an RFID architecture framework be developed with the specific requirements of large networked RFID systems in mind.

In this chapter, we present a P2P RFID architecture that can be used to develop very large scale RFID systems such as RFID enabled global supply chain management systems. The main contributions of our work are: (1) Extensive analysis and identification of RFID based supply chain management system requirements (2) The creation of a P2P RFID system architecture that fulfils those requirements and is specifically aimed at multi organizational networked RFID systems spread over a large physical area and (3) Comparative analysis of the proposed architecture with the EPCGlobal architecture framework (EPCGAF) and other RFID architecture frameworks identifying its advantages over them. The comparative analysis suggests that the proposed architecture has a number of advantages over systems developed using other architecture frameworks. These advantages include improved scalability, reliability and performance and easier integration with existing systems.

3.2 Networked RFID Systems

There are a number of key requirements and features of global supply chains that must be taken into consideration when creating an RFID enabled simply chain management system. If the underlying architecture and related infrastructure of the system being developed do not adhere to the requirements set by the features of a global supply chain the system will not be able to perform at full potential and therefore, will not produce all the forecast benefits at the level expected [3]. In Section 3.2.1, we will identify and discuss these requirements. In Section 3.2.2 we will discuss the possibility of using a peer-to-peer technology to meet these requirements and improve the performance of RFID systems.

3.2.1 RFID System Requirements

Depending on the size of the supply chain and the number of independent partners in it, the number of RFID tags in a global supply chain management system can grow to
The system will also have a very large number of RFID reader installations required at various locations all generating a massive amount of data [1]. It is estimated that Wal-Mart generates around 7TB of data each day [6]. Therefore the RFID system implemented must be able to scale to match the needs of extremely large and complex networked supply chain applications [46]. In addition, the central look-up process needs to scale extremely well. Considering that the Wal-Mart supply chain has up to 4 billion tags passing through it a year with thousands of different readers, this central lookup needs to store all the data locations address for all those tags and reply to queries sent by hundreds of thousands of readers from all over the world.

The use of RFID in supply chain management is centred on the need for improved, real-time information sharing between independent partners [11]. In a large networked RFID system the transaction data automatically generated by partners are stored at different partner locations while the static data is stored with the manufacturer of the product the tag is affixed to. Therefore when retrieving the RFID data, the system has to first locate that data wherever it may be situated in the world and then it must retrieve it. Consequently an RFID system that caters to global supply chains has to make the process of data location and retrieval as easy as possible. But RFID data can also be successfully used to automate and increase the efficiency of some of the organization’s internal processes as well [11]. Accordingly, partners in an RFID enabled supply chain may want to use RFID to generate private data to enhance, automate and carryout internal processes. In that event an RFID system that is created for a global supply chain has to store that private data so that it’s not accessible to external independent partners. It also needs to store that data in a manner which does not negatively affect the performance of the system [6].

Global supply chains have a large number of independent partners all using different business applications. This means that the stored data must contain a large amount of information with minimal loss and that information needs to be presented in different formats and at different levels of filtering and aggregation to different partners and different business applications [96]. Therefore to ensure that the maximum amount of information is retained the transaction data generated must be stored in a highly granular form. But storing data at an extremely granular level has two major drawbacks. First, it increases the computational load on the database server by increasing the
amount of data stored and shared. Second, it increases the computational load placed on the retrieving partner’s system since that system would be burdened with the task of filtering, aggregating and formatting the raw granular data that is retrieved. Because currently this data is stored by external partners and needs to be retrieved repeatedly each time its required, it leads to unnecessary duplication of data retrieval and manipulation and formatting process [13]. Therefore unless the system architecture takes explicit steps to avoid or minimize this, the system retrieving data will be forced to carry out large amount of unnecessary duplicate data manipulation.

Possibility of easy integration with existing software and systems with minimum redevelopment is another significant advantage when creating an RFID enabled supply chain application [6]. Commonly an RFID system will be developed to replace an existing legacy system which integrates with a wide variety of software systems and business applications [97]. If the already existing software systems and business applications have to be significantly modified or redeveloped to integrate with the new system it would require a significant monetary and time investment from a company which is already developing the new RFID. In addition to easy integration with the current systems the architecture for newly developed RFID systems must also ensure interoperability with other partner systems, as data sharing is the key advantage of networked RFID [46].

Networked RFID technology at its core is a data storage and communication mechanism. Therefore, the fundamental information security objectives such as confidentiality, integrity, availability, authentication, authorization and non-repudiation must be preserved [21]. But at the same time, RFID is the product of complex technology with the characteristics of a number of different types of IT systems. This complex nature of networked RFID systems calls for a high level of data integrity and security [58]. Accordingly, an RFID system deployed in a networked global supply chain setting must be able to provide the data integrity and security requirements of a number of different types of IT systems. Typically, Modern IT systems are so tightly integrated into the day-to-day business processes of the owning organizations that the systems become critically important. This is especially true of RFID systems and therefore high availability of the RFID system becomes a key requirement.
3.2.2 P2P Technology in RFID

Most current RFID systems use a distributed client server model. In essence, each partner has data stores, which act as the sole server for all data that is stored on that server. All other partners act as clients and retrieve the required information from the single server. However, the client server model poses many issues when used in this manner in a networked RFID system. The main problems are lack of scalability, unbalanced loads on systems, bandwidth bottlenecks and creation of SPOF at the servers. Nevertheless, most of these weaknesses are removed or reduced in P2P systems.

Peer-to-peer networks are networks where each participating node acts as both a server and a client for all or a majority of the data contained in the overall system. The distributed and shared nature of P2P networks has a number of advantages including resilience to DDoS attacks, removal of SPOF at server locations, enhanced bandwidth, very high scalability, independence on a centralized server and freedom of monitoring by a central authority, all of which are highly desirable to networked RFID systems such as global supply chain management systems [42].

P2P technology for RFID systems can be divided into two main parts: P2P data sharing and P2P data lookup. In P2P data sharing, the RFID data generated by partners will be shared using P2P technology rather than client server technology. Here, when a partner retrieves P2P data from another partner, selected portions of that data (the parts that do not change) will be made available to other partners of the system via P2P data sharing. This will enable a number of improvements at the data servers including increased scalability and availability.

P2P technology can be used to carry out the centralized data lookup (ONS) service offered by the EPCGlobal. The current ONS service has a number of weaknesses including bad scalability, weaknesses to DDoS and cache poisoning attacks, being controlled by a central authority and being a SPOF to the overall RFID networks used globally [23]. By using a P2P based data lookup system all of these weaknesses can easily be removed from the RFID data lookup process.

However, like all technologies P2P has its own drawbacks. The two big issues for P2P networks are node churn and security and privacy concerns [42]. In normal public P2P networks, partners/nodes are constantly joining and leaving the network. In these
situations, the overall network topology and query routing must be reorganized each time this happens to ensure proper network structure. This creates additional overhead on the system, increases the difficulty of identifying the required data sources and sometimes makes the overall data routing paths non-optimal. Nevertheless, in supply chain systems the network partners and therefore the P2P nodes are very stable. In supply chain systems the nodes (the partner data services and middleware) are fixed. When one such node enters the system it is there permanently, except for occasional down time, till that partner leaves the supply chain. Therefore the issue of node churn does not apply to a P2P network in a supply chain management RFID system, and the need for constant rearranging of the network topology and routing paths is removed [42]. The other main drawback of P2P systems is security concerns. In public P2P networks, anyone can join the network and then share and retrieve data anonymously. Therefore, access control and privacy concerns come into play as well as trust concerns. However, partners in a supply-chain management system are not strangers. They are business entities with existing business partnerships and connections. Therefore the security and privacy concerns plaguing public P2P networks do not apply to supply chain RFID P2P networks as users know that people in their P2P RFID network are people with legitimate access to that data [23].

3.3 P2P Networked RFID Architecture Framework

The role of the RFID architecture outlined below is to organize and manage RFID infrastructure throughout the enterprise in order to capture data and generate RFID data in real time, store it with minimal loss of information and share that data in real time with other independent partners of the supply chain. Both the data lookup and data sharing in the proposed architecture will be done using P2P technology rather than client server systems. In Section 3.3 we will describe the overall architecture. Sections 3.4, 3.5 and 3.6 will discuss in depth the modularized middleware, P2P data sharing and P2P data lookup in that order.
Figure 3.1: P2P Networked RFID architecture

The proposed architecture (shown in Figure 3.1) is divided into 6 main layers. The main components in each layer and their functionality are explained in detail in the rest of this section.

### 3.3.1 RFID Layer

The RFID layer contains RFID components that are used in the system. For the proposed architecture, these are RFID tags, RFID readers.

RFID tags and the identifier contained on them links the electronic data stored in backend databases to the physical object they are attached to. RFID Tags typically comprises an electronic memory module, a logic module, communication antenna and possibly a power source [33]. In our architecture the RFID tag can also store the static data (item name, price, manufacturer etc) associated with the objects it’s attached to in addition to its identifier for ease of access. It may also store some amount of transaction data. Figure 3.2 is an example of the typical content of a tag in our system.
The RFID readers are used to read and write data to and from RFID tags. The readers used in our architecture will be standard RFID readers which can be used to update the tags. They can either be mobile/handheld readers or more powerful stationary readers and are composed of three main parts: the receiver, transmitter and controller [33]. The readers will read the EPC and any additional data on any authenticated tags within reader distance and update the data on the tags as and when necessary [98]. Read RFID tag data will be forwarded to the middleware (specifically the reader interface) via the secure internal network. Any updates the readers do to the tags will be based on data they receive from the middleware.

### 3.3.2 Data Storage Layer

The data storage layer contains the databases and house the data services that store RFID data that is used by the company and shared with other partners. In the proposed architecture we have two main databases for each location: they are the shared RFID repository and the private RFID repository. The shared RFID repository is the database where transaction data generated by the middleware and transaction data retrieved from external partners is stored. Only the data that the organization wants to share with its partners will be stored here. As this data has to cater to the information requirements of all the partners of the organization, it will be filtered and aggregated sufficiently to ensure that the database does not grow to an extreme size, but it will still be kept granular enough so that not too much information is lost. In contrast, private transactional data and external data that have been fully formatted as required by the internal software are stored in the private RFID repository. This data is only accessible by internal business applications. Further details of data storage and P2P sharing are given in Section 3.5.

### 3.3.3 Middleware Layer

The middleware acts as a hub that connects all other components and allows inter component communication. It also acts as the software link between RFID readers and
the business applications. It is the most important and complex component of the proposed architecture [96]. The middleware in our architecture is developed to be modular. This helps ensure that system is easily upgraded and those different modules can be upgraded and maintained separately if required. It also helps compartmentalize the different tasks carried out by the middleware as each module is in charge of one specific task, or a group of interrelated tasks. The middleware in our architecture will have a number of specific tasks to carry out including the following:

- Filter and collect the data received from RFID tag reads from multiple readers [16].
- Carry out security tasks to ensure the integrity confidentiality and authenticity of data [21].
- Translate the tag identifier and data retrieved to information from their raw binary form according to the tag data specification and vice versa [13].
- Generate transaction data based on business events [99].
- Retrieve, aggregate, filter and format RFID tag data as required by applications or by storage [100].
- Act as the communication hub for different components [36, 100]

Full details on each module the tasks they carryout and the high level logic for carrying out those tasks are given in Section 3.4.

### 3.3.4 Software Layer

The software layer contains all the business applications that are used by the company and communicate with the RFID system [96]. These applications are typically pre-existing applications that have been modified to integrate with the RFID system, and use the information supplied by it. They retrieve information as required by contacting the middleware and use that information to automate, improve and streamline existing business processes. They also generate business application data that is then combined with RFID tag reads and actual business processes to generate transaction data about particular RFID tags [39]. In our architecture, the business applications will only query for the information it requires from the middleware. The data lookup, data retrieval and formatting and consolidation of data from multiple sources are all done by the middleware. Therefore, the only changes that need to be done to the current applications
are that they must be changed to communicate with the middleware for any data from the RFID system. The application interface provides a method with which this communication can take place with the exact specifics of how that data they receive is obtained by the middleware remaining invisible to the business applications.

### 3.3.5 Data Lookup Layer

The data lookup layer typically contains the components that allow partners to locate data about any given RFID tag regardless of which partner is storing it. The current standard for RFID systems is the global ONS service offered by the EPCGlobal in [13]. Unfortunately, this centralized service has lead to a number of issues including lack of scalability, security issues, control of overall system by one entity and the creation of a SPOF at the EPCGlobal run ONS service. Therefore, in our architecture we propose a P2P data lookup approach, which removes or minimizes most of these concerns. Full details of the P2P data lookup functionality are given in Section 3.6.

### 3.3.6 Communication Layer

The communications layer contains the network hardware that allows different components to communicate with each other. The communications layer consists of the internal (red arrows) and external networks (blue arrows). The internal network is a secure communication and IT network that physically connects all the internal components of a single partners system [36]. Therefore, the internal network will connect the readers, middleware, private RFID repository, shared RFID repository and the business applications to each other. It is normally implemented as a LAN network. In contrast, the external network allows secure communication between the components of the system belonging to different partners and the EPCGlobal services. Therefore, the external network will connect the middleware of different partners. The external network is normally implemented via the internet. In addition, both the internal and external networks can and should be fully secured using tried and tested security methods such as IDS, firewalls and encryption technologies used in any other networked system.
3.4 Modularized Middleware

The modularized middleware is the most important and complex component of the architecture. Not only does it do a majority of the data processing and collecting it also carries out the data lookup for external data sources. It also acts as the hub of the overall system and allows the different components effectively communicate with each other. The modularized middleware consists of five main modules, each of which carries out a distinct and self-contained task or set of tasks and five interfaces, which allows it to communicate with the other components of the architecture. The tasks of each module are explained in detail below.

3.4.1 Data Cleaning and Filtering Module

In networked RFID systems, readers are simultaneously bombarded by the transmissions of a large number of different RFID tags and may only have a limited time in which to respond [46]. Another important factor is the relatively low read rates of RFID readers. Therefore, most RFID systems recommend that duplicate readers are deployed at important reader locations to increase the read speeds of tags and to minimize possible interference from environmental factors. But the use of multiple overlapping readers and the low read rates of typical RFID readers mean that the data streams must be cleaned and filtered before it can be used [45]. Figure 3.3 shows a system which has 3 readers physically (R₁, R₂, R₃) located to try and cover the full tag area which contains 6 tags (T₁, T₂… T₆).

---

¹ For the sake of simplicity, in the following subsections the diagrams only include the system components that are of concern to that particular module's task.
Given the distance and location of the tags and readers Table 3.1 shows the read capabilities of the readers for the tags

Table 3.1: RFID tag reads from multiple readers

<table>
<thead>
<tr>
<th></th>
<th>T₁</th>
<th>T₂</th>
<th>T₃</th>
<th>T₄</th>
<th>T₅</th>
<th>T₆</th>
</tr>
</thead>
<tbody>
<tr>
<td>R₁</td>
<td>High</td>
<td>Low</td>
<td>None</td>
<td>High</td>
<td>Low</td>
<td>None</td>
</tr>
<tr>
<td>R₂</td>
<td>None</td>
<td>Low</td>
<td>High</td>
<td>None</td>
<td>None</td>
<td>High</td>
</tr>
<tr>
<td>R₃</td>
<td>None</td>
<td>High</td>
<td>Low</td>
<td>Low</td>
<td>High</td>
<td>High</td>
</tr>
</tbody>
</table>

Therefore, when doing tag reads the above system must use the data from multiple different readers to remove false reads and get a full coverage stream of all tags. Hence, readers R2 and R3 can be used for creating the final data stream for tag T6 while all three readers can be used for creating the final data stream for tag T2. In the case on tag T1 only one data stream is available.

In the proposed architecture the Data Cleaning and Filtering Module (DCFM) of the middleware is in charge of accepting RFID tags reads from multiple different readers, cleaning it by removing false reads and filtering duplicate reads and collating the reads from multiple readers [101]. When carrying out this task the DCFM first receives the tag read stream from all readers in the area. It then removes the bad reads.
from each stream for each tag. Once all bad reads have been eliminated unnecessary duplicate reads is filtered out to create a single stream, which is then forwarded to the security module.

Please note that there are a number of different data filtering and collection mechanisms proposed in recent literature that the developer could implement. For further details on the different challenges in RFID data filtering and management and a comparison of the different approaches possible refer to [102].

### 3.4.2 Security Module

There are a number of security requirements that must be met to fully ensure the complete security of a networked RFID application. The security module is in charge on managing and providing all those security, privacy and data integrity requirements [103]. Overall, the minimum security functionality that the security module needs to provide includes mutual authentication, transmissions confidentiality, transmission integrity and anonymity for the mobile tags. In addition, for networked systems with independent partners which use updatable tags, security requirements such as storage confidentiality and integrity, non-repudiation and tag malware protection and access control need to be implemented as well [17]. There are a number of security solutions and protocols that offer varying levels of protection and features in current literature as discussed in [104]. The developer should select and implement in the security module the security solutions he thinks are suitable for the specific system he is building. For detailed analysis of a number of different RFID security protocols and their weaknesses refer to [104]. Additionally, in chapters five and six of this thesis we present an RFID security protocol and an RFID malware detection technique, which can be implemented in this module.

### 3.4.3 Data Translation Module

In RFID systems, data stored on the RFID tags are saved in raw binary format. They are normally separated into fields for each data item. Therefore the binary data needs to be translated into an understandable form before it can be used [105]. The data translation module is in charge of translating the binary data stored in the RFID tag into text form usable by the data storage and business applications and text form data received from
data storage and business applications into binary form to be stored on the tag. Figure 3.4 shows this process:

Figure 3.4: RFID tag data translation

The data translation module in our middleware needs to carry out two types of data translations. It needs to translate raw binary data from the tag into a format usable and understandable by the databases and business applications and it also needs to translate the information received from the databases and business applications into raw binary data to be stored on the tag.

3.4.3.1 Tag Data Translation
For the first task the module receives the cleaned and collected data stream from the DCFM module and using the separators imbedded on the tag data splits it into different fields. Then for each field it retrieves the data translation rules and applies it to the field data. If the translation happened without any issues the data is forwarded onto the database and business applications (via the higher modules and interfaces). If there is an issue during translation, the data is discarded and the module requests a reread of the tag.

3.4.3.2 Tag Update Translation
The other translation task that must be done by the data translation module is the process of translating the update data received from the database and the business applications into the raw binary data that will be stored on the tag. To do this the module first receives the required data from the higher modules and identifies which field each data should be stored to. Then for each field to be updated it retrieves the data translation rules and applies them to the received data to transform them into the binary data that will be stored on the tag. Once the translation is complete, the update is forwarded to readers.
3.4.4 Event Generation Module

Because the main task of RFID systems is the creation of information, one of the most important functions of an RFID system is the automated generation of transaction data concerning the EPCs it identifies. This process is shown in Figure 3.5. Transaction data is created by associating EPCs with specific business events and transactions [99]. The event recognition module is responsible for recognizing such preset business events and transactions and using tag read events and business application data for generating transaction data for various EPCs. In RFID systems the middleware can also be tasked with controlling and coordinating certain actions in the physical environment in response to the automatically generated transaction data as well as generating certain electronic business processes in response to the actual RFID event generated [106]. The event generation module carries out this process in our architecture,

For example, imagine that the middleware for a company receives the tag reads for ten thousand new tags at a specific warehouse. At the same time, the system recognizes that the physical process of receiving a goods shipment from logistics company X was initiated at that warehouse. Additionally it also receives information from the business applications that a logistics company X is delivering the goods for invoice I₁ for that warehouse from seller Y. By combining this data the Event Generation Module (EGM) generates the invoice received event for that warehouse for invoice Y and associates all the newly picked up tags with that invoice. It may also initiate the opening of outbound logistics for sales for that specific good from that warehouse as well as initiating the
payment process for the invoice in question for the business applications. Table 3.2 shows some examples of transaction data generated by the event generation module.

Table 3.2: RFID transaction data

<table>
<thead>
<tr>
<th>Actual business process</th>
<th>Business application data</th>
<th>Final data stored in shared RFID repository</th>
<th>Business process trigger</th>
</tr>
</thead>
</table>
| Tags $T_1$ picked up by readers at warehouse | Receiving of new stock at warehouse from logistics supplier $S_1$ delivered by truck $TR_2$ | Goods for invoice $I_1$ received at warehouse $W_1$ | Tag $T_1$ is at warehouse $W_1$  
- Tag $T_1$ was delivered by truck $TR_2$ of company $S_1$  
- Tag $T_1$ belongs to delivery invoice $I_1$ | Start shipping out goods for orders which can be fulfilled using the received goods  
Divert any more shipments if warehouse space is full |
| Tag $T_1$ leaves warehouse | Logistics supplier $S_2$ picks up stock for delivery using truck $TR_1$ | Goods for sale $SL_1$ shipped out from warehouse $W_1$ | Tag $T_1$ is no longer at warehouse $W_1$  
- Tag $T_1$ was picked up by truck $TR_1$ of company $S_2$  
- Tag $T_1$ belongs to sale invoice $I_2$ | Request for more stock if extra space is available |

Transaction data generation is very strongly dependent on the businesses processes of the company developing the RFID system. Therefore, it is up to the developer to develop the actual transaction event generation logic for the various business processes used by that company.

3.4.5 Data Management Module

In networked RFID systems, different partners at different data storage locations store data about the tags used in the system. Additionally in our approach, the RFID data is shared using a P2P model rather than a client server model. Therefore, proper management and identification of this distributed data as it's being saved and retrieved is required if the system is to work efficiently. In the proposed architecture, this task is the responsibility of the Data Management Module (DMM). Therefore the DMM is tasked with the responsibility of locating, retrieving, aggregating and formatting data from multiple different sources in such a manner as to most effectively respond to any single data request [96]. It is also responsible for deciding how the data generated by the EGM and data retrieved from external partners should be saved between the private repository and shared repository. Full details of the P2P processes carried out by this module can be found in Sections 3.5 and 3.6.
3.4.6 Communication Interfaces

In the proposed architecture, the middleware acts as the hub between other components. Therefore, for each component the middleware communicates with there needs to be communication interface that enables that communication. Each communication interface defines a number of requests and responses that the interface will receive and respond. Table 3.3 describes each interface in more detail.

<table>
<thead>
<tr>
<th>Middleware Side</th>
<th>Other side</th>
<th>Main messages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reader Interface</td>
<td>Data Cleaning and Filtering (DCFM)</td>
<td>RFID readers</td>
</tr>
<tr>
<td>Private DB Interface</td>
<td>Data management module (DMM)</td>
<td>Private RFID repository</td>
</tr>
<tr>
<td>Shared DB Interface</td>
<td>Data management module (DMM)</td>
<td>Shared RFID repository</td>
</tr>
<tr>
<td>Application Interface</td>
<td>Data management module (DMM)</td>
<td>Local business applications</td>
</tr>
<tr>
<td>Data Query Interface</td>
<td>Data management module (DMM)</td>
<td>EPCIS lookup services Root ONS of EPCIS External ONS of partners External EPCIS</td>
</tr>
</tbody>
</table>

The reader interface serves as the communication channel between the middleware and the RFID readers and allows the middleware to command RFID readers to read the EPC of tags, read other data stored on tags, write to tags, and kill or lock tags. It also allows middleware access to a variety of reader management features and abstracts the actual hardware details of the readers from the next component [13]. When new readers or a new reader model is added to the system the reader side of the interface will change while the DCFM side will remain the same. Therefore, the reader interface allows for easy expansion of the system readers with minimal changes to core modules.

The middleware also has four following additional interfaces, which allow it to interface with other components of the system. The private DB interface acts as the communication channel between the DMM and the private RFID repository. The shared DB interface acts as the intermediary of the DMM and the shared RFID repository. The application interface acts as the intermediary between the DMM and the internal
business applications of the organization. The data query interface is what enables the RFID system of external business partners to communicate with the middleware of the organization and vice versa by allowing it to receive and respond to data requests from external independent partners.

3.5 P2P Data Storage and Identification

The architecture we proposed in this chapter uses P2P technology to improve the data sharing enabled by RFID systems. In this section, we will explain the proposed RFID data identification and storage approach.

3.5.1 RFID Data Types

Because of differences in RFID data and typical files shared via P2P networks not all RFID data is suitable for sharing. Therefore, the first step is identification of the RFID data suitable for sharing in a P2P method.

<table>
<thead>
<tr>
<th>RFID data types</th>
<th>Generated at</th>
<th>Generated by</th>
<th>Is it updated</th>
<th>Example</th>
</tr>
</thead>
</table>
| Static data                      | Birth of object | Manufacturer | No            | • Batch number of item is 3476  
• Item expires on 14/08/2012     |
| Constant Transaction data        | Over lifetime | Supply chain partners | No | • Item was checked into warehouse x23 on 21/10/2010  
• Item was sold to supply partner Y as part of invoice 21187 |
| Updatable Transaction data       | Over life time | Supply chain partners | Yes | • The next destination for item is warehouse x56  
• There are currently 1863 lots of model number Z at warehouse 34 |

RFID data is currently categorized into two groups: Static data (data created at the birth of the object and which does not change over its lifetime) and Transaction data (data that is generated by different partners over the course of its lifetime and is subject to change). However, sharing data that is constantly changing over a P2P network creates data synchronization problems and should be avoided. However, sharing only static data would defeat the purpose of using P2P as it’s only a very small percentage of the total data concerning any given RFID tag. Therefore, to remove data synchronization requirements and ensure that the highest amount of data can be shared via P2P we
further split the transactional data into constant and updatable transaction data. Table 3.4 highlights the differences of the three types.

Only the static data and constant transaction data will be shared as P2P data while the updatable transaction data will only be available from the original partner who creates and maintains it.

### 3.5.2 P2P RFID Data Identification

The next challenge in sharing RFID data over a P2P network is the proper identification of the meta-data required to properly identify any given RFID transaction data. Unfortunately, unlike typical fixed data files that are shared over P2P networks the data that is generated and shared by RFID systems is granular information concerning specific RFID tags. Hence, normal techniques, such as the use of hashes to identify files, cannot be used as the lookup tables would grow too large relative to the actual data tables.

<table>
<thead>
<tr>
<th>Normal RFID System</th>
<th>Proposed System</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag identifier</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Original partner identifier</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Date generated on</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Data class</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Transaction information</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In current RFID systems there are three main types of identifiers: tag identifiers, partner identifiers and data service (EPCIS) identifiers. In non P2P systems RFID data can be identified using just the unique identifier of the tag the information is about and therefore data is associated with only the tag identifier. Therefore, in a normal client server environment, this meta-data is sufficient, as data is only available from the partner who generated the data. In a P2P environment, this is not enough. In this environment, the data needs to be associated with the original partner who generated that data when it’s stored at a different location. Therefore, in our system in addition to
the tag identifier we also use the unique identifier of the partner who generated that data to identify transaction data. In addition, to allow for stronger and more granular identification and filtering we also store the date on which that information was actually generated. Table 3.5 compares the identification data stored for RFID data in current systems and the proposed system.

3.5.3 P2P Data Retrieval and Storage

In the proposed system when business events are generated and stored as transaction data, the partner will also record the other identifiers listed in the above table. When a partner requests specific data concerning an RFID tag all of the above information will be transmitted for each transaction event along with the actual transaction information.

Data retrieval and formatting in the proposed architecture is quite a complex task as there are four main possible sources of data for any given company as shown in Figure 3.6. Therefore, the DMM must retrieve the data from these four different sources aggregate it and format it as required by the business application. Data lookup and retrieval is carried out in response to data requests from internal business applications or from external partners.

![Figure 3.6: RFID data retrieval, aggregation and formatting](image)

When a business application requires data concerning a specific tagged object it will relay that request to the data management module. Based on the requirements of the application and its current knowledge it can also relay additional information such as who generated that data and around when it was generated. When the data management module receives a data request it will first identify which data services of which partners might contain the information required. This is done via a P2P data lookup using data...
profiles, which is explained in Section 3.6. Once the proper locations of the required data have been located, the middleware of the system will request those partner data services for the required data.

When retrieving data the DMM will need to query each possible data source for the required data in the most effective manner. The data retrieval order we propose for this architecture is tag, private database, shared repository and finally external P2P data sources. When requesting P2P data from external data services the additional information provided by the application can be used to retrieve only a subset of the data at the external service by using them as filters. So for example rather than requesting for data concerning Tag X the system can ask the P2P data service for data concerning tag X, which was generated by partner Y between two specific dates. The additional filtering possible with the additional identifiers stored with transaction data reduce both the workload and network bandwidth required for the overall system. Once the required data has been fully retrieved by the DMM the data will be aggregated and formatted as required. It will then be sent to the application that requested that data. If the data request is from an internal business application then the middleware will use external sources if required, if it is from an external partner’s middleware only internal data sources will be used and the data will be transmitted in raw format with no aggregation or formatting.

Once data is retrieved from external partners it will be stored on the local servers and shared with other partners via the P2P network. However, before this can be done the system first needs to identify which data can be shared via P2P and which data is constantly updated by the original partner and therefore unsuitable for P2P distribution. This decision is made based on the data class the data belongs to. The portions of the retrieved data that has been classed as static data or constant transactional data (indicated by either 1 or 2 for data class) will be saved in raw form in the shared data repository. Any data classed as updatable transaction data (indicated by 3 in the data class) will be formatted and forwarded to the business applications but will not be stored and shared via the P2P network. The retrieving partner will never modify any external data shared by that partner.

Additionally the DMM will also receive transaction data that has been locally generated by business applications. In addition to the actual transaction data, the system
must also store the additional identifiers required for P2P data identification. The tag identifier, original partner identifier, date generated on and data class will be generated by the business application and sent along with the transaction data to the middleware. This data can be split into two main classes: private data and shared data. All locally generated shared transaction data, regardless of data class, will be stored and shared via the P2P network. Locally generated transaction data classed as private will be stored in the private repository and will not be available to external partners. The private data will be stored after formatting to reduce the need for repeated formatting of the data. The shared data will be stored as is to ensure that the maximum amount of information is held in it.

### 3.6 P2P Data Lookup

By sharing RFID data using P2P technology, we increase the overall system scalability and remove a number of bottlenecks in the system. However, the traditional centralized ONS based data lookup service offered by the EPCGlobal still pose serious scalability issues and create a SPOF. Therefore, a P2P data lookup is required to remove the SPOF at the ONS lookup service and further improve system scalability.

#### 3.6.1 Creating Service and Data Profiles

In P2P data lookup, each node must be able to let the other nodes know what data it is making available for retrieval. Therefore, in our system each data service will have a service profile, which contains Meta data about the service it offers. It also will generate a list known as its data profile, which contains information about the data it’s sharing, and share these two profiles with other partners.

##### 3.6.1.1 Service Profile

The service profile for each data service will contain the following information: (1) The service identifier/address, (2) the type of service it offers (e.g.: SOAP or EPCIS), (3) The partner identifier of the company that runs it (e.g.: the company prefix allocated to it by the EPCGlobal), (4) The service profile time stamp (which indicates the particular time this service profile was created and distributed) and (5) a service profile expiration time which will indicate when the service profile should be discarded and a new one retrieved.
Additionally each service profile will contain two extra data fields that are filled by the partners when they retrieve and locally store the service profiles of external services. These are the data profile time stamp and data profile expiry. These two fields allow the partner to track when he last downloaded the data profile for a particular external data service and when he should retrieve a newer data profile for that service. It is important to note that each partner can have multiple data services, and that each service will have its own service profile. In addition, we also recommend that each data service contain the service profiles for all of that partners data services. Table 3.6 shows an example service profile.

<table>
<thead>
<tr>
<th>Service Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service address 124.134.12.32</td>
</tr>
<tr>
<td>Type of service EPCIS</td>
</tr>
<tr>
<td>Partner identifier 778291</td>
</tr>
<tr>
<td>Service profile time stamp 07:24:39 01/05/2012</td>
</tr>
<tr>
<td>Service profile expiry 07:24:39 02/05/2012</td>
</tr>
<tr>
<td>Data profile time stamp 16:32:12 12/05/2012</td>
</tr>
<tr>
<td>Data profile expiry 20:32:12 12/05/2012</td>
</tr>
</tbody>
</table>

3.6.1.1 Data Profile

In RFID systems, the data that is shared is RFID event and transaction data concerning the tagged objects. Hence, the P2P nodes must have a method with which it can share the tag numbers it has data concerning and which partner originally generated that data. Therefore, in addition to the service profile each data service will also contain a data profile (shown in Table 3.7). This is a list of all the tags, which that particular service has data about. It also contains the original partner who generated that data and the last time data for that tag and partner combination was generated or updated.

<table>
<thead>
<tr>
<th>Data Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Address: 124.134.12.32</td>
</tr>
<tr>
<td>Tag identifier 265216762 Partner identifier 387287 Last update 16:32:12 12/05/2012</td>
</tr>
<tr>
<td>87162908 Partner identifier 838933 Last update 11:26:57 19/07/2011</td>
</tr>
<tr>
<td>445783754 Partner identifier 320933 Last update 07:45:17 12/03/2011</td>
</tr>
<tr>
<td>6546976967 Partner identifier 407321 Last update 16:32:14 12/05/2012</td>
</tr>
</tbody>
</table>
Both these profiles as well as the profiles of other partner’s data services and the service profiles of other company service profiles will be stored and managed by the DMM of the middleware that runs the data service. These two profiles will then be shared with other partners of the system. External profiles will be regularly updated based on expiry dates specified in the service profile. Details of the sharing method are as follows:

3.6.2 Distributing Partner Service Data Profiles

In typical P2P systems, one of the biggest challenges is locating the nodes and proper distribution of the file location details. This difficulty is mainly because of the high node churn and unstructured nature of typical P2P networks and also the fact that most nodes in the system have no prior connections or associations [42]. Fortunately for us, in networked RFID systems in supply chains both these problems are removed or greatly minimized as discussed in Section 3.2.2. Therefore, rather than using some of the more complex and resource intensive methods such as the flooding technique used in Gnutella or less scalable centralized data lookup systems used in Napster [107] we suggest a simpler and more scalable method which we have named “chain distribution”. The proposed technique is based on the fact that partner chains pre-exist in supply chain RFID systems. The proposed system leverages these existing partner chains to ensure full discovery of all possible data locations. Once the initial discovery is done, the partners use direct communication to retrieve data from partner data services. The partner data service profile distribution and lookup process is divided into two main parts: (1) partner data service discovery (2) partner data profile update.

3.6.2.1 Partner Data Service Discovery.

Partner data service discovery is the process of partners been informed about completely new data services for that RFID network. This needs to be done when a new partner joins the supply chain or an existing partner adds additional data services.

When a completely new partner joins a supply chain, the data service discovery for that partner will be done via the chain distribution method mentioned before. Because a new partner can only join a supply chain with the knowledge and approval of at least one existing supply chain partner that existing partner will be able to directly get the service profiles of the new partner. That partner will then be tasked with distributing the new partners service profiles to their up or down stream partners. When the other
partners receive the new partners service profiles they will directly contact those services and retrieve their data profiles. The existing partner who initiated the new partner will also be in charge of forwarding all services profiles it has to the new partner.

Figure 3.7: Discovering the data services of new partners

In Figure 3.7 the new partner joins an already existing supply chain with the knowledge and approval of partner 3. When this happens partner 3 receives the service profiles of the new partner’s data services directly from the new partner. He also forwards all the service profiles he has of the networks data services to the new partner. Partner 3 then forwards the service profiles of the new partner to its direct partners (partner 2) and also makes it available via general sharing. When partner 2 receives the new service profiles, he realizes that this is a new data service for which he has no data profile. Then using the service address and service type Meta data included in the service profile he directly contacts the new partner’s data services and requests them for their data profiles. He also forwards the new service profiles to his direct upstream partner or partners (partner 1). On receiving the service profiles of existing data services the new partner contacts them directly and retrieves the data profiles of those partners (messages 7 and 8). In this way the new partner’s service profiles are distributed to all the partners along the existing supply chain connections as all partners are connected by the supply chain. At the same time, the partner who invited him into the supply chain brings the new partner up to date on existing data services.
Service discovery is also required when an existing partner adds new data services. This is done slightly differently from new partner discovery. When a new data service is added that partner includes the service profile of the new service in all his existing data services. As all service-profiles have an expiry date partners need to regularly contact all the data services and refresh their service profiles. When this happens, any service profiles of new data services will be sent along with the current service profile for that particular data service. When the requesting partner receives the service-profiles, he realizes that this is a new data service for which he has no data profile. Then as before, he directly contacts the new data services and requests them for their data profiles.

Figure 3.8: Discovering new data services of existing partners

The service discovery that happens when an existing partner adds a new data service is shown in Figure 3.8. Partner A adds a new data service B to the network. He then includes the service profile of B in existing data service A. Next time the service profile of service A expires the external partner B sends a service profile refresh request to data service A of partner A. Because data service A has the service profile of a new data service it sends that service profile along with its own current service profile in response to the service-profile refresh request of partner B. When partner B receives this new
service profile, he realizes that this is a new data service for which he has no data profile. Then using the service address and service type metadata included in the service profile he directly contacts the new partner’s data service (service B) and requests them for their data profile.

3.6.2.2 Partner Data Service Update

The other important task in data lookup is ensuring that the details that partners have about external data services are correct and up-to-date. This is carried out through updates for the data profile for each data service.

When a partner first discovers a new data service in its network, it directly contacts that data service and requests for its data profile. However, overtime the data shared by a data service will change and grow as more tags pass through that partner and the system generates more and more transaction data. Whenever new transaction data is generated, an existing data entry is updated or when it retrieves new data from external partners and makes it available for sharing the data service will update its data profile. To update its profile all it does is either change the last update field for existing tag/partner combination or to add a new tag identifier, partner identifier, and last update tuple for new tags to its data profile.

To ensure that the data profiles are up to date external partners will use the data profile time stamp and the data profile expiry fields in the service profile for each data service. The data profile time stamp will store the time when the data profile for that data service was last updated and the data profile expiry will store the time when the data profile for that service needs to be updated. Please note that unlike the service profile time stamp and service profile expiry these two fields are generated and maintained by the external partner and not the actual data service. Therefore, partners are free to pick the update rate for different external data services based on personal preference and needs. When the data profile for a specific service expires the partners middleware will contact that data service and request a data profile update.

Using these up-to-date data profiles partners can request the information they require from any number of data services rather than just the data service that originally generated it. For example, a partner wants to know when the RFID tag X left the premises of partner A and he already knows that the tag arrives at partner A on 15/06/2012. Now it will check all the data profiles for data services that have data
generated by partner A about tag X and which were updated after 15/06/2012. When it identifies the possible data-services, it will then contact those in order and request for the required data till the correct data is received from one data service.

### 3.8 Comparative Analysis

The EPCGAF has done a very good job in creating a global standard for RFID systems and defining and supplying the core services required for intra organizational data lookup and sharing. However, the one size fits all approach it takes contains some weaknesses that limit organizations from harnessing the full power and functionality of their RFID system. In addition, while some of the newly proposed P2P architectures are promising they still have some weaknesses. Therefore, in the following section we will do a theoretical evaluation of the proposed architecture compared to other architectures using the basic RFID enabled supply-chain system requirements identified in Section 2 as metrics (scalability, availability/reliability, system integration, security and performance). Table 3.8 shows a summary of this analysis.

Scalability is the ability of a system to carry out its duties with minimal loss of performance when the number of active components and the interactions between them increase [108]. RFID applications in a global supply chains need to be able to scale effectively with the size of the supply chain. In the EPCGAF, there are a number of components in the system that scale badly. These are primarily the lookup services offered by the EPCGlobal itself and the EPCIS of the partners. Because these points are all centralized and based on client-server technology, the sheer number of data requests that can happen in a large-scale supply chain system can easily overwhelm them. In the more recently proposed P2P based architectures [23, 42] the bottleneck that is created by the EPCGlobal lookup services is removed as they use a more scalable decentralized P2P technique for the actual data lookup. However, the bottlenecks at the actual RFID data sources still exist because they are still a single server fulfilling the requests of numerous partner systems and business applications. In our architecture both the lookup process and the actual data distribution is done using modified P2P techniques. As P2P systems offer much more scalability [107], this means that the lookup and data services in the proposed architecture scale much better that than the client server based lookup and data services used in previously proposed architectures.
Availability of a system can be defined as the degree to which the architecture is susceptible to failure at the system level in the presence of partial failures within components, connectors, or data [108]. The structure of the EPCGAF introduces a number of Single Points of Failure (SPOF), at the lookup process and the EPCIS of partners, which is extremely bad to have in large networked systems [13]. In the more recent P2P architectures the SPOF at the data lookup is removed but the system still retains a SPOF at the data service components. In contrast, the architecture proposed by us removes these SPOF and increase system availability and reliability in a number of ways. (1) By storing static data on the RFID tag in addition to the EPCIS of the manufacturer, we reduce the dependency on external data sources and services, (2) By storing filtered, aggregated and formatted data in a local DB, we further minimize the dependence on all external services and (3) By using P2P technology, which is proven to have much better availability and reliability than client-server technology, we increase the overall reliability and availability of the networked system as a whole.

In IT, system integration is the process of combining different computing systems and software applications physically or logically to act as a coordinated whole. In the EPCGAF the business applications are in charge of carrying out the ONS lookup process, retrieving data from the partners EPCIS and then filtering, aggregating and formatting that data [96]. Therefore all business applications that use the EPC data will need to be significantly modified to carry out this additional functionality [46]. In the P2P architectures [23, 42] the authors discuss only the data lookup aspect of the system but do not discuss and other system components or how the proposed mechanism will be integrated into existing applications. In our framework, the modularized middleware does the ONS lookup, data retrieval and the filtering and aggregation of that data. Therefore, existing legacy applications only need to be modified to communicate with the middleware making system integration much easier.

Ensuring the security and privacy of the EPC data stored in the system is of utmost importance in any RFID enabled application [2]. Unfortunately, by its own admission the security features that are provided by the EPCGAF are very basic [13]. In addition, The ONS service that it offers have a number of security issues such as vulnerabilities to DDoS attacks and cache poisoning [31]. The P2P architectures in [23, 42] remove the vulnerability to cache poisoning and partially removes the vulnerabilities to DDoS
attacks. However, they do not have any security features built into them; neither do they discuss the potential threats to the proposed system. Our architecture offers a number of advantages when it comes to system security. (1) The P2P technology that our architecture employs eliminates the vulnerability to cache poisoning and reduces vulnerability to DDoS attacks and (2) Our architecture explicitly identifies the need for high level of security and has a module in the middleware framework dedicated to providing the required security functionality in one centralized place. Moving the security to the middleware and away from the readers and RFID tags allow a much more robust and thorough security solutions to be implemented. E.G:- The security module of the proposed architecture can implement the security solutions presented in chapters 5 and 6 of this thesis.

System performance is the efficiency with which the system does any given task. The EPCGAF performs certain tasks quite inefficiently. For example the static data in the EPCglobal system comes from the EPCIS of the manufacturer while the transactional data concerning the objects are held by many different partners [13]. Therefore, most of the time, the system needs to at least access the manufacturer’s EPCIS as well as the EPCIS of the partner containing the transaction data required when retrieving data. In addition the data recovered from another partner’s EPCIS needs to be filtered, aggregated and formatted before it can be used by business applications [96]. As the EPCGAF requires that the data be retrieved from the partner’s EPCIS each time it is needed, the system must filter, aggregate and format the same data whenever it is retrieved from partner’s EPCIS. This creates unnecessary duplication of work, which affects system performance negatively. Because the EPCGlobal implements the ONS as a hierarchy, the system also has to complete a large number of processes to complete a data lookup. The P2P architectures improve the performance during data lookup. By using P2P techniques, they reduce the number of steps required for the data lookup. Our architecture improves performance in a number of different ways. Due to static data being stored on the tag, our architecture has a fewer number of situations requiring a data look-up. In our architecture, once data is retrieved and formatted its stored in the local private database. By doing this, our architecture significantly reduces the amount of duplicate filtering, aggregation and formatting done by the middleware component of the system compared to the EPCGAF. This significantly reduces the load on the
middleware and therefore improves the overall system performance. We also use the chain distribution method for locating new data services and partners. This approach is a lot more efficient than typical decentralized P2P node discovery methods such flooding, because partners directly query each other. In addition the P2P data sharing balances the loads more efficiently and reduces bottlenecks therefore improving overall system performance.

This analysis shows that the architecture developed and presented in this chapter has a number of advantages over the other available architectures when used to develop large networked RFID systems for supply chain management systems.

![Table 3.8: Comparison of EPCGAF and proposed architecture](image-url)
3.9 Summary

In this chapter, we have first carried out an analysis of the nature of a networked global supply chains and from that derived the specific system requirements and features that is required of an RFID enabled global supply-chain management application. We have then developed a P2P RFID architecture that is optimized for global supply chain management applications. The main component of this architecture is a modularized middleware. We have explained the tasks carried out by each module in detail as well as described their functionality at a logical level. We have also discussed the interaction between the different modules. For our architecture, instead of using client-server technology, we propose a P2P approach for both the data service lookup and RFID data sharing. Finally, we have carried out an analysis of our proposed framework comparing it to the current RFID architecture standard: the EPCGAF and other recently proposed architectures. This analysis showed that our architecture has a number of advantages over other options specially in building large supply-chain management systems. Particularly our framework offers better scalability, performance, reliability and security as well as easier integration with existing systems compared to existing architectures.
A Security Framework for Networked RFID

In this chapter, we develop and present a conceptual framework for analysing the threats, attacks and security requirements pertaining to networked RFID systems. The vulnerabilities of, and the threats to the system are identified using the threat model. The security framework itself consists of two main parts: (1) The attack model: which identifies and classifies the possible attacks and (2) The system model which identifies the security requirements. The framework gives readers a method with which to analyse the threats any given RFID system faces. Those threats can then be used to identify the attacks possible on that system and get a better understanding of those attacks. It also allows the reader to easily identify all the security requirements of that RFID system and identify how those requirements can be met. Therefore the main research contributions of this chapter are (1) Creation of a complete and holistic networked RFID security framework which does not contain any weaknesses identified in current security frameworks (2) Evaluation of that framework by applying it to a real world RFID system.

4.1 Introduction

Even though RFID technology has proven to be an asset for tracking objects in large networked systems such as global supply chain management systems there are a number of issues that prevent its widespread adoption [28]. In RFID the main current barrier to adoption is the large number of security concerns and the additional performance
overhead placed on the system when generating and sharing such a vast quantity of data [40].

Networked RFID systems are a relatively complex type of RFID system. This complexity arises from some of its features such as the use of wireless communications and mobile data containers (RFID tags), its highly distributed nature and the presence of multiple independent entities that are authorized to access the system. Due to its wireless communication method and distributed nature networked RFID systems are vulnerable to a great number of malicious attacks at the edge of the system (tags, readers and wireless communications). These attacks can range from simple ones such as passive jamming and eavesdropping to more sophisticated attacks such as physical cloning of tags, man-in-the-middle attacks and even RFID malware [109]. In RFID systems these threats can be mounted either through physical or logical access to system components. In addition, networked RFID systems can be attacked by internal partners as well as external attackers. Therefore the security threats and attacks that are faced by RFID networks are both numerous and extremely diverse. To successfully manage and eliminate all these different types of threats a large number of security requirements must be implemented [17].

Due to the large number and different types of attacks and threats facing an RFID system, fully securing one is a very complex task. This task is made even more difficult by the number of different components that must be protected and the large number of security concepts that must be upheld. Currently one of the biggest barriers to the widespread adoption of networked RFID systems is the unresolved security issues inherent in them [21]. Without a proper security framework to reference most companies have no method with which to reliably assess the vulnerabilities of their proposed or developed system. Nor do they have a method with which to decide how best they can remove those vulnerabilities and fully secure their RFID systems. Due to this problem most companies are still reluctant to implement RFID based solutions, even though the benefits they pose are great. Therefore the need for a networked RFID security framework that will allow developers to successfully identify, manage and secure against the threats and attacks faced by RFID systems is currently very acute. But if such a framework is to be successfully developed a few challenges must first be overcome. Networked RFID systems, while seemingly similar to normal networked
systems, differ quite significantly from them [33]. Therefore the most important challenge is analysing how the security requirements of networked RFID differ from the security requirements of typical networked systems.

If a security framework for networked RFID was successfully developed it would ensure that the companies that are implementing RFID solutions could easily analyse and verify the security of those systems leading to higher adoption rates for networked RFID. Therefore in this chapter we develop and present “A networked RFID security framework”. Before developing the security framework we develop a threat model which analyses the threats faced by networked RFID systems, the vulnerabilities they exploit and the attacks that result from those threats. The actual framework is composed of two main components (1) The attack model: which identifies and classifies all possible attacks on networked RFID systems, and (2) The system model which identifies all security requirements needed to protect a networked RFID system. The developed security framework will create a systematic path to identifying all the potential threats to any given RFID application, better understanding the attacks that can be mounted on the system and also identifying the security requirements for securing the system.

4.2 Threat Model for Networked RFID

The distributed and collaborative nature of networked RFID along with the use of low cost RFID tags which employ wireless communications mean that there are a large number of threats faced by these types of systems [17]. These threats exploit vulnerabilities in the system to become attacks. The threat model we develop (shown in Figure 4.1 and Table 4.1) and present in this section will identify and discuss the common threats faced by networked RFID systems. It will then analyse how those threats exploit certain vulnerabilities that can exist in the system to become specific attacks that compromise the security of the system.

![Figure 4.1: Threat model for networked RFID](image-url)
Table 4.1: Networked RFID threat model

<table>
<thead>
<tr>
<th>Threat</th>
<th>Exploits weaknesses:</th>
<th>And results in</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Interception or</strong></td>
<td>Lack of secure mutual authentication</td>
<td>Replay attacks</td>
</tr>
<tr>
<td><strong>modification of</strong></td>
<td>Lack of secure mutual authentication and confidentiality</td>
<td>Eavesdropping attacks</td>
</tr>
<tr>
<td><strong>system data and communications</strong></td>
<td>Lack of secure mutual authentication, integrity verification and confidentiality</td>
<td>Man-in-the-middle,</td>
</tr>
<tr>
<td></td>
<td>Lack of sufficiently strong encryption</td>
<td>Crypto attacks,</td>
</tr>
<tr>
<td></td>
<td>Lack of storage confidentiality</td>
<td>Physical attacks</td>
</tr>
<tr>
<td></td>
<td>Poor physical security of tags</td>
<td></td>
</tr>
<tr>
<td><strong>Introduction of</strong></td>
<td>Lack of strong and secure mutual authentication</td>
<td>Tag cloning</td>
</tr>
<tr>
<td><strong>false objects into system</strong></td>
<td>Mobility of tags</td>
<td>Tag spoofing</td>
</tr>
<tr>
<td></td>
<td>Lack of proper mutual authentication</td>
<td>Reader masquerading</td>
</tr>
<tr>
<td></td>
<td>The mobility of the tags and the easily identifiable radio fingerprint on low cost tags</td>
<td>Radio fingerprint tracking</td>
</tr>
<tr>
<td><strong>Invasion of privacy</strong></td>
<td>Lack of physical security</td>
<td>Physical destruction of components</td>
</tr>
<tr>
<td></td>
<td>Low resources available on tags</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Broadcast mechanism of communications</td>
<td>Active jamming</td>
</tr>
<tr>
<td></td>
<td>Use of pseudonyms of some security protocols</td>
<td>Passive jamming</td>
</tr>
<tr>
<td></td>
<td>Built in lock and kill commands and lack of</td>
<td>De-synchronization of tags</td>
</tr>
<tr>
<td></td>
<td>mutual authentication</td>
<td></td>
</tr>
<tr>
<td><strong>Denial of service</strong></td>
<td>Lack of storage integrity</td>
<td>Unauthorized tag locking or killing</td>
</tr>
<tr>
<td><strong>RFID malware</strong></td>
<td>Lack of strong and secure mutual</td>
<td>RFID malware (worms, viruses, SQL and Script injection)</td>
</tr>
<tr>
<td></td>
<td>authentication lack of storage integrity.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Weak anti-malware protection on backend servers</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Lack of strong and secure mutual</td>
<td>Buffer overflow</td>
</tr>
<tr>
<td></td>
<td>authentication Lack of proper buffer control in readers</td>
<td></td>
</tr>
<tr>
<td><strong>Attacks by</strong></td>
<td>Lack of access control</td>
<td>Elevation of privileges (reading)</td>
</tr>
<tr>
<td><strong>internal partners</strong></td>
<td>Lack of data ownership</td>
<td>Elevation of privileges (writing)</td>
</tr>
<tr>
<td></td>
<td>Lack of non-repudiation</td>
<td>Repudiation of actions</td>
</tr>
<tr>
<td></td>
<td>Lack of access controls</td>
<td>Partner de-synch, killing or locking tags for partners</td>
</tr>
<tr>
<td></td>
<td>Lack of non-repudiation</td>
<td></td>
</tr>
</tbody>
</table>

The two most common threats faced by RFID systems are the possibility of an attacker intercepting or changing the wireless communications between tags and readers. Because low cost RFID tags do not contain sufficient resources for standard security functionality most networked RFID systems cannot implement strong authentication, confidentiality or integrity verification [56]. Therefore potential attackers can exploit the lack of these security mechanisms to mount a number of attacks on the system. These attacks allow them to either gain access to confidential information or allow them to exchange sensitive data so as to harm the system. In eavesdropping the attacker
exploits the lack of confidentiality in the system to listen to a legitimate conversation between readers and tags. This allows the attacker to gain access to confidential information. Data leakage attacks: a more complex form of eavesdropping, are mounted by eavesdropping on a large number of authenticated communications between a tag and a reader and using that data to gain confidential information [51]. Another common attack, which exploits lack of proper mutual authentication in RFID communications, is the man-in-the-middle attack. This attack is a form of active eavesdropping in which the attacker makes independent connections with a reader and tag that is communicating while making them believe that they are talking directly to each other. The attacker then proceeds to change valuable data or steal confidential information as it’s transmitted through him between the reader and the tags [50]. Lack of strong mutual authentication is also exploited to mount replay attacks. Here the attacker uses previously used responses by a tag or a reader in a challenge-response protocol to initiate a new session with the tags or readers of the system. This allows the attacker to access either the reader or tags as a legitimate component and steal information or wrongly update data stored on the system. Attackers can also exploit the weak encryption techniques used in RFID systems using low cost tags to mount crypto attacks on those systems. Crypto attacks use various mathematical methods to break through the weak encryption in communications and gain access to the information that’s being communicated [72].

Another major threat faced by networked RFID systems is the attacker introducing false objects into the system. These types of threats primarily exploit the lack of proper mutual authentication between tags and readers. Tag cloning, tag spoofing and reader impersonation are all attacks that result from this type of threat being successfully leveraged into an attack. In tag cloning the attacker replicates all the identification details of a legitimate tag on to a forged tag and introduces it in to the system [110]. In tag spoofing, rather than creating a new tag, the attacker just transmits the identification information of legitimate tags in the vicinity of readers using a transmitting device. In reader impersonation the attacker impersonates a reader of the system, rather than a tag, and tries to access tags by initiating a conversation with them [50]. All three of these attacks enable the attacker to either feed false data to the system or retrieve confidential data from the backend database while posing as a legitimate component of the system.
The threat of RFID malware has only been recently brought to the attention of RFID researchers [111]. These attacks are mounted by exploiting poor mutual authentication or storage integrity checks to store malicious code on the tags or to create cloned tags with malicious data and introduce them into the system. When these tags are read by readers the malware either corrupts the data in the backend databases or compromises the middleware of the system by infecting it. In buffer overflow attacks, which are a simpler version of malware attacks, the attacker makes a tag try and send the same block of data repeatedly till it overflows a memory buffer in either the readers or the middleware of the system thereby corrupting data or even crashing that component or even the whole system [112]. The threat of RFID malware is very severe because it not only corrupts data but it can also spread from tag to backend database to tag and affect a very large amount of tags and backend databases very quickly. More complex RFID malware can even infect the business applications or open breaches in the firewalls protecting the internal system allowing attackers access the internal components of the system directly [26].

Another type of threat to networked RFID systems is the invasion of privacy enabled by tracking tags. Here the attacker sets up a network of RFID readers and exploits the mobility of the tags and the fact that most tags reply with their unique identification number on being queried by any reader. By identifying the tag at regular intervals they can then build a map of their movement over time thereby tracking either the person or the object the tag is attached to [40]. Tag constellation tracking is a more complex form of tag tracking where the attacker tracks a combination of tags rather than a single tag [51]. Additionally most radio transmitting devices have what is known as a radio fingerprint which is created at manufacture and is unique to each tag. By exploiting this attackers can sometimes track individual tags even if they don’t have access to its identification number.

Another threat faced by RFID systems is the attacker rendering components or even the whole system unavailable by various means. The successful completion of such an attack can cause part of or even the whole system to become unavailable. This in turn affects the performance of not only the RFID system but also that of the business applications of both the company in question and those of the external partners that rely on the RFID system for information. The easiest Denial of Service (DoS) attacks to
mount on RFID systems are signal jamming and physical destruction of system components. Signal jamming takes advantage of the fact that wireless communications use a broadcast medium and floods the channel with powerful signals using the same frequency. This makes it impossible for the relatively weak RFID signals to propagate through thereby effectively rendering the tags unavailable to be read by the readers [50]. The attacker can also exploit the limited amount of resources available on readers and tags and bombard a specific reader or tag of the system with data requests thereby overloading that tags or the reader’s capability to reply. Attackers can also exploit the relatively lower physical security available to RFID tags to just physically damage or destroy the tag thereby shutting down the system [51]. A more complex DoS threat that can be mounted on some RFID systems is the de-synchronization of tags with the backend components. Here the attacker takes advantage of the temporary pseudonyms used by certain RFID protocols to de-synchronize the tags next response from the response expected by the rest of the system. This makes it impossible for the tag and reader to communicate till they are manually re-synched [74]. Attackers can also exploit the built-in KILL or LOCK commands on certain RFID tags to disable those RFID tags thereby rendering them useless till they are reactivated [21].

In RFID systems with multiple partners the possibility of a partner compromising the overall system for his own profit is an ever-present threat. The possibility of attacks by partners jeopardizes the trust the users have in the system thereby reducing the overall advantages that can be gained by implementing a networked RFID system. Here the partners take advantage of either their authorized access to the system or the lack of proper access controls to mount attacks which compromise the system for other users. Repudiation attacks happen when an entity sends a communication or changes system data but later denies doing so [2]. In RFID this can be in the form of changed tag data to forged tag broadcasts. Another threat in this environment is partner de-synchronization. This attack is a carried out by an authorized independent partner of the system and would typically de-synchronize the RFID tag with the backend databases and readers of the other partners in the network. In a multi-entity RFID network some companies may want to use the RFID tags to store data that is confidential or the capabilities of some companies may be limited (e.g.: can read tag data but cannot update it or can read just some of the tag data and not all of it). By exploiting the lack of proper access controls a
partner can mount an elevation of privileges attack and increase the access he has to the tags of the system to gain confidential information on the partners business processes or to even update or delete RFID data without proper authorization [51].

4.3 Security Framework

When developing a security framework for any system there are two main areas that have to be explored and analysed: (1) The possible attacks to the system and their features and (2) The important system components that must be protected. Therefore our security framework is composed of two main parts: the attack model and the system model.

4.3.1 Attack Model

The attack model analyses the possible attacks on the system and classifies them based on various different criteria. The attack model (Figure 4.2) allows readers to get a better understanding of all the attacks that can possibly be mounted on a networked RFID system. The attack model analyses the attacks in three ways. It looks at the source of the attack, the negative impact the attack will have on the system if successful and the method by which the attack access the system components it’s attacking.

4.3.1.1 Access Methods

One of the most important aspects of any attack is how that attack is actually mounted on the system. For networked RFID applications we identify two main access methods with which attackers can attack the system: logical access and physical access.

The main method of communication between tags and readers in an RFID system is wireless communication. Therefore potential attackers can exploit this
communication method to gain logical access to either the memory modules of the system or even the information that is being remotely communicated between the tags and readers. Unauthorized logical access to the system can be gained in a number of different ways [51]. The attacker can pretend to be an authorized tag or reader to gain access to the system. They can also intercept the wireless transmissions between the readers and tags and decrypt them to gain confidential information. Eavesdropping, tag tracking and replay attacks are some common types of logical access attacks.

In normal IT systems the data is stored in a physically secure location such as a data server in a server room. Whereas RFID systems store some of its sensitive data on the RFID tag itself. These tags are affixed to physical objects that travel along the physical network [11]. In addition, the RFID readers may also be mounted in relatively unsecure locations such as warehouses and transport vehicles. Therefore some of the components in networked RFID systems have relatively low physical security compared to the components of a normal network. Physical access attacks are mounted by attackers who gain physical access to the tags or the readers of the system. Physical destruction of tags and physical reading and writing of tags are some common physical access attacks. (See Table 4.4 for details of access methods of all identified attacks)

4.3.1.2 Attack Impact

Another important aspect of any attack is the impact it will have on the system. We categorize the attacks possible on networked RFID systems into 5 main groups based on the negative impact they will have on the system: (1) Modification, (2) Interception, (3) Interruption, (4) Fabrication and (5) Tracking. The impacts we have identified are slightly different from the STRIDE developed by Microsoft. Table 4.2 maps our impacts to the STRIDE model for comparison while Table 4.3 shows the impact each identified attack can have on the system.

Table 4.2: Mapping of attack classification to Microsoft STRIDE model

<table>
<thead>
<tr>
<th></th>
<th>Spoofing</th>
<th>Tampering</th>
<th>Repudiation</th>
<th>Information Disclosure</th>
<th>Denial of Service</th>
<th>Elevation of Privilege</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modification</td>
<td></td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Interception</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Interruption</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Fabrication</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tracking</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>

4.3.1.2.1 Attack Impact Analysis

We analyze the impact of each identified attack on the system. The impacts we have identified are slightly different from the STRIDE model developed by Microsoft. Table 4.2 maps our impacts to the STRIDE model for comparison while Table 4.3 shows the impact each identified attack can have on the system.
Interception and modification attacks are the most common attacks possible on any IT system. In interception attacks the attacker intercepts data while it’s stored or being communicated and gains access to confidential information. In modification attacks the attacker changes, deletes or creates data in the system without authorization [22]. In RFID systems these attacks can be carried out with either remote or physical access. The wireless nature of RFID means communications can be easily intercepted as they are travelling between tags and readers allowing for remote modification and interception. The storage of sensitive data on a mobile RFID tags means the system can also be subject to modification and interception attacks via physical access to the tags [20]. Replay-attacks, man-in-the-middle attacks, eavesdropping, data leakage and crypto attacks are all common interception attacks on RFID systems. Modification attacks while harder to mount also have a much greater impact on the system if they are successful. If a modification attack is successful then critical data that’s not available elsewhere may be lost or corrupted in the process. If the system is to continue working there must be a way in which the system can identify and recover from these attacks. Successful modification attacks mounted via replay attacks or man-in-the-middle attacks can be further leveraged to carryout RFID malware or buffer overflow attacks [113]. Data integrity of the tag can also be compromised by natural causes such as electromagnetic fields and physical shocks [2].

Ensuring the availability of any IT system is of paramount importance. An interruption attack renders the system unusable by blocking access to some or all parts of the system or by ensuring that different parts of the system can’t properly identify or communicate with each other [2]. The availability of RFID systems are of vital importance to corporations using them as unavailability of the RFID systems leads to the unavailability of all the applications that rely on it. In RFID systems interruption attacks can vary from simple active radio jamming attacks to complex attacks that desynchronize tags with the central database in RFID systems using pseudonyms [109]. The availability of RFID systems can also be compromised by physical or logical destruction of tags or their data and the use on unauthorized kill/lock attacks to stop the functionality of the tags.
Table 4.3: RFID attacks by their possible impact on system

<table>
<thead>
<tr>
<th>Attack</th>
<th>Modification</th>
<th>Interruption</th>
<th>Fabrication</th>
<th>Tracking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replay attacks</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Eavesdropping attacks</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data leakage</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Man-in-the-middle,</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Crypto attacks</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Physical reading of tags</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Physical writing to tags</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tag cloning</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Tag spoofing</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Reader masquerading</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Tracking (Forward and Backward)</td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Tag constellation tracking</td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Radio fingerprint tracking</td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Physical destruction of components</td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Active jamming</td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Passive jamming,</td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>De-synchronization of tags</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unauthorized tag locking or killing</td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>RFID Malware (worms, viruses, SQL and Script injection)</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Buffer overflow</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elevation of privileges (reading)</td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Elevation of privileges (writing)</td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Repudiation of actions</td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Partner de-synch,</td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Killing or locking tags for partners</td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
</tbody>
</table>

In addition to the above common types of attacks RFID systems are subject to two more types of attacks: fabrication attacks and tracking attacks. Fabrication happens when the attacker inserts new messages or items into the system without the knowledge or authorization of the system owners [110]. In RFID systems these attacks mainly manifest as cloning or spoofing attacks where the attacker inserts fabricated tags into the system. The attacker may also try to carry out a fabrication attack by pretending to be an authorized reader and querying tags for their information as well. Like with modification attacks successful fabrication attacks can be further leveraged to mount malware attacks on the system or in some cases cloned tags can be used to authenticate
false object as their real counterparts (medicines and other designer consumer goods).

Tracking attacks are possible on networked RFID systems due to the mobile nature of the tags. By tracking the movement of individual tags along the physical network the attacker can gain insight into the structure of the network as well other information such as location of specific vehicles or people and the efficiency of physical network and the business processes that support it. Tracking can also carried out my physically or logically compromising a tag and then using the information gained to identify the past or future transmissions of that tag (forward tracking and backward tracking) [51].

### 4.3.1.3 Source of Attack

Another important part of any attack is the source of that attack. In networked RFID systems, unlike in standalone RFID systems, the attack can originate from one of two different sources: External attackers and internal attackers. See Table 4.4 for details.

External attackers are persons or organizations that have no authorized access to the system but try and gain some or complete access to the system by various means. These attackers have a number of motivations for these attacks including stealing restricted information, changing system data, stealing the goods the tags are affixed to, tracking the tags as they travel along the physical network, cloning the RFID tags for counterfeiting and spreading malicious malware into the system thereby disrupting its performance [50]. Internal attackers on the other hand are authorized users of the system who try and gain more access than they are entitled to (a partner who is only allowed to read tag data updates it) or try to disrupt the system in such a way as to harm other users of the system. Because networked RFID systems are typically used by a number of independent entities who are simultaneously partners and competitors, internal attacks are a major concern for these types of systems [6]. There are a number of attacks that internal attackers can carry out on networked RFID systems including creating data/updating tags and then denying they made those changes (repudiation) and gaining access to private data stored on the tags by other partners in the supply chain (Elevation of privileges (reading)) as well as changing data they are not authorized to change (Elevation of privileges (writing)).
Table 4.4: RFID attacks by access method and attack source

<table>
<thead>
<tr>
<th>Access Method Attack Source</th>
<th>Logical</th>
<th>Physical</th>
<th>External</th>
<th>Internal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replay attacks</td>
<td>Y</td>
<td></td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Eavesdropping attacks</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data leakage</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Man-in-the-middle</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Crypto attacks</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Physical reading of tags</td>
<td></td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Physical writing to tags</td>
<td></td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tag cloning</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tag spoofing</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reader masquerading</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tracking (Forward and Backward)</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tag constellation tracking</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Radio fingerprint tracking</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Physical destruction of components</td>
<td></td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Active jamming</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Passive jamming</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>De-synchronization of tags</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unauthorised tag locking or killing</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RFID Malware (worms, viruses, SQL injection)</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Buffer overflow</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elevation of privileges (reading)</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elevation of privileges (writing)</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Repudiation of actions</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Partner de-synch</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Killing or locking tags for partners</td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
</tbody>
</table>

4.3.2 System Model

In the previous section we discussed the attack model component of the security framework. In this section we will discuss the system model component of the security framework. The system model identifies the key system components that need to be protected and analyses the security concepts that must be preserved to fully secure the system and the security requirements that result. The final security framework is shown in Figure 4.3 while the development process behind will be explained in detail in the following sections.
4.3.2.1 System Components

In a networked RFID system the system components include RFID tags, RFID communications, readers, middleware, and backend data storage and business applications. In RFID systems the edge components: namely the readers and tags are implemented on non-standard devices and they communicate using a non-secure channel. The other components on the other hand are all implemented and connected by standard IT infrastructure. Therefore in our framework we separate the components into three groups: The tags, the wireless communications and the back-end components as shown in Figure 4.4.

Tags in standard large scale networked RFID applications typically comprise an electronic memory module, a logic module and a communication antenna. The memory module of the tag can consist of memory which can be read only, write once or updatable. The logic module of low cost RFID tags consists of between 500 to 10000 gates. The tags can be powered with either passive a semi-passive power source or an active power source [33]. In any RFID system the tags hold the identifier of the object it’s attached to as well as additional data such as brand, product, expiry date and price of
the object it’s attached to. Unlike other IT components RFID tags are extremely mobile and have very low physical security.

![RFID System Diagram](image)

**Figure 4.4: Components of networked RFID**

Wireless communications are how tags and readers of a networked RFID system communicate. The wireless signals are communicated via an insecure channel and is transmitted using a broadcast mechanism. The range of an RFID transmission depends on the frequency of the radio wave used and the power applied. One of the main concerns in securing RFID systems is providing adequate security to the wireless communications given the limited power and computing capabilities available on low cost tags and the throughput required. Currently most RFID communications are secured using relatively weak security protocols. These protocols normally employ low resource operations such as bitwise XOR and one-way hashing to try and secure the communications.

The backend components include the readers, middleware, an RFID repository and business applications. RFID readers are used to read and write data to and from RFID tags. RFID readers do not normally store any data or translate any data. They just act as a messenger between the RFID middleware and the RFID tags and relay raw binary information from the middleware to the tags and vice versa. The middleware filters,
translates and formats the data to the RFID repository that then stores that data for future retrieval from either the middleware or by business applications [11].

4.3.2.2 Security Concepts

Security concepts are core requirements that must be met to ensure the security of any given application. The security concepts for an application depend on the system architecture and its functionality. In networked RFID systems we identify five key security concepts that must be met to ensure the security of the system: authenticity, integrity, confidentiality, privacy and availability [17] as shown in Figure 4.5.

![Figure 4.5: Applying the security concepts to the components](image)

Figure 4.5: Applying the security concepts to the components

Authenticity ensures the validity of the claimed identities of the components participating in a communication. It ensures that no entity is actually trying to masquerade as someone or something they are not [48]. In networked RFID this means that all readers must be able to identify each tag as being who it claims to be and that all tags can identify each reader as being part of the system.

Integrity ensures the correctness and accuracy of data, whether its stored data or data being communicated, against unauthorized creation, modification and deletion. It
also provides an indication if the data has been compromised in this manner and in some cases allows for the retrieval of the correct data from the corrupted version [48].

For networked RFID this requires the integrity of the data stored on the RFID tag and the backend database as well as the data that’s being communicated between readers and tags. Complete implementation of integrity also ensures that once data has been created, modified or deleted by an authorized party those changes cannot be denied by that party.

Confidentiality protects the data of the system from unauthorized disclosure to parties who are not entitled to access that data. In networked RFID systems confidentiality requires that both the data stored on the tag itself and the data that is being communicated between readers and tags is secure from unauthorized access by outside parties. It also requires that the authorized partners do not gain access to data that they are not allowed to access [17].

In addition to the above mentioned requirements, full security of an RFID system requires that the concepts of privacy and availability be assured. Privacy provides for the protection of information that might be derived from observing system activities [50]. In networked RFID systems this protects against attacks such as tracking of tag movements and the use of traffic analysis to derive information about the data that’s being communicated. The final security concept, availability ensures that there is no denial of access or service by any component of the system to other authorized parties or components of the system [48]. For networked RFID applications this requires that all readers can access all the tags of the system and vice versa. It also requires that the overall RFID system be available when other business applications require data from it.

For a networked RFID system to be fully secured all of the above mentioned security concepts must be fully assured for each group of components in the system. Therefore the five core security concepts of authenticity, integrity, confidentiality, privacy and availability must be separately assured for the tags, communications and backend components of the RFID system.

4.3.2.3 Networked RFID Security Requirements

The security requirements are the security functionality that needs to be implemented to secure the system. As explained above, each security concept for each component must be separately secured before the overall system can be considered secure. Therefore we
identify that the all following security requirements must be implemented to ensure that all the security concepts for each component is fully secured. Table 4.5 maps the security requirements to the security concepts and system components that they protect.

Table 4.5: Networked RFID security requirements

<table>
<thead>
<tr>
<th></th>
<th>Integrity</th>
<th>Authenticity</th>
<th>Confidentiality</th>
<th>Privacy</th>
<th>Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Tags</strong></td>
<td>Mutual Authentication, Storage Integrity, Non-repudiation, Data Ownership</td>
<td>Mutual Authentication</td>
<td>Mutual Authentication, Storage Confidentiality, Access Control</td>
<td>Anonymity</td>
<td>Physical Protection, Electronic Protection</td>
</tr>
<tr>
<td><strong>Communications</strong></td>
<td>Transmission Integrity</td>
<td>Transmission Integrity</td>
<td>Transmission Confidentiality</td>
<td>Data Leakage Protection</td>
<td>Electronic Protection</td>
</tr>
<tr>
<td><strong>Backend Components</strong></td>
<td>Mutual Authentication, Non-repudiation,</td>
<td>Mutual Authentication</td>
<td>Mutual Authentication, N/A</td>
<td>N/A</td>
<td>Electronic Protection</td>
</tr>
</tbody>
</table>

Mutual authentication allows the two communicating entities to verify the identity of the entity they will be communicating with [20]. In RFID systems mutual authentication is required to ensure that the readers and tags that are communicating are of the same network and have access to that information and are authorized to do the data modifications they request. Proper mutual authentication not only ensures the authenticity of the components but also provides confidentiality and integrity for the data stored on the backend components by ensuring that only authorized entities can access that data. Secure mutual authentication is also required as a base to implement a majority of the remaining security requirements and is therefore the most important security requirement in networked RFID. Most RFID security protocols implement mutual authentication as part of its security features.

Because RFID systems use wireless communications it is easy for potential attackers to intercept them. Transmission confidentiality ensures the security of data while it’s broadcast by ensuring that it cannot be understood by an attacker who intercepts it [48]. Transmission confidentiality typically uses various encryption methods to ensure that attackers cannot understand data intercepted in this manner. The protocols in [58, 62, 114] all provide for transmission confidentiality. Storage confidentiality ensures the security of data while it’s stored on the tag or the backend.
components. In networked RFID systems the tag is easily accessible by attackers. Hence it is important that tag data be secure in case of physical or logical compromise of the tag [17]. Unfortunately storage confidentiality is an area that has received very little research attention till now.

Complete data integrity requires the two security requirements of transmission integrity and storage integrity to be implemented. In RFID not only can potential attackers intercept wireless communications but they can also modify those communications compromising their integrity. Ensuring that communications that have been illegally modified can be identified and the original data recovered is done by transmission integrity. Transmission integrity in RFID typically employ various low cost encryption methods and message digests/hashe to enable the detection and recovery of RFID communications that have been externally modified [70]. The protocols presented in [58, 62, 114] all provide communication integrity but not data recovery. RFID broadcast data recovery still remains an area open for research. Storage integrity ensures the integrity of data while it’s stored on the tag. The relative physical and logical accessibility of RFID tags on which sensitive data is stored dictates that storage integrity is a high priority, especially in systems which store additional sensitive data on the tag. Both encryption methods and journaling systems such as the ones presented in [67, 69] can be used to ensure some storage integrity but currently they are not sufficiently strong or complete enough. Currently storage integrity of RFID tags is an area with very little research contributions. Because current applications store a majority of data on the backend database rather than the tag this is not currently an issue. But in the future as more and more systems store data on the tag itself this will become a high priority security requirement.

In a multi-entity RFID system some partners may like to store data on the tags that is accessible only to them [17]. Therefore the security functionality of the system must protect that data from unauthorized access by internal partners. If other partners gain access to private data this can lead to confidentiality issues as well as create trust problems among them. The security requirement of access control ensures that partners can only gain access to data that they are authorized to do so. While the confidentiality of private partner data is assured by access control its integrity is assured by data ownership. Data ownership guarantees that entities can only modify data that they are
authorized to modify or delete. Both access control and data ownership require granular data storage based on the network partner who owns it. They also require that finely grained controls are implemented to ensure that data on the tags can only be accessed or modified by partners based on a predefined set of access and update rules. If done successfully multiple independent partners can have read and write access to the RFID tags while ensuring that access is limited to the data they are actually authorized to access or modify. RFID tag access control and data ownership are research areas with minimal current research.

Non-repudiation is a security requirement that ensures the identification of the origin of data and the assurance of the genuineness of that data [115]. For example a partner, who is authorized to do so, may change the price stored on the RFID tag then deny doing so. Non-repudiation ensures that this cannot happen. By ensuring that a record of what data is modified by which entity is kept, non-repudiation guarantees accountability of those partners increasing overall trust in the system [69]. Non-repudiation techniques for backend database exist but no non-repudiation techniques have been developed for securing the data stored on RFID tags.

In networked RFID systems attackers are able to invade the privacy of the tag holder by tracking their movement. Anonymity ensures that tracking attacks cannot be mounted on the tag. There are a number of tracking attacks possible on RFID systems including tag tracking, forward and backward tracing, and radio fingerprint tracking as well as tag constellation tracking [21]. The protocols presented in [58, 62, 114] all use a system of pseudonyms to ensure the privacy of the tags. This system also provides a limited form of protection against forward and backward tracking as well. The privacy of RFID systems can also be compromised by data leakage. Data leakage happens when attackers intercept communications or other information about the system over a long period of time and gradually use that data to derive information about the system and the data being communicated by it [51]. Common data leakage attacks include traffic analysis as well as attacks on confidentiality of the system via a large number of eavesdropping attacks on weakly encrypted communications. Data leakage protection requires that the attacker is unable to derive any information about system data or system security over the course of multiple eavesdropping sessions. Data leakage protection can be implemented by using techniques such as implementation of more
advanced hardware configurations and ensuring that all communications are strongly encrypted. The protocols presented in [58, 62, 114] all also claim strong and secure encryption which makes data leakage attacks impossible.

Networked RFID systems availability can be disrupted through physical or electronic means. The security requirements of physical protection and electronic protection secure the system against these types of attacks. Physical protection will protect the system against attacks such as physical tag destruction, removal of tags from the tagged items or the use of items such as aluminium foil to mask tag signals. Physical protection is easily provided by ensuring that physical access to RFID tags is limited to authorized parties when possible and by having deterrents such as electronic surveillance present when unauthorized parties can access the tagged items. Electronic protection is required to protect the system from denial of service attacks mounted through electronic means. These attacks include overwriting/destroying tag data through strong electronic magnetic pulses, disruption of communications through radio frequency jamming, overloading of the system capability with repeated requests or by de-synching the tags with the authorized readers through message blocking. To combat these types of threats, functionality such as filtering of tags or readers which repeatedly send the same request and multiple frequency transmissions can be implemented. Also the use of RFID tags that can simultaneously transmit in several different wireless frequencies or move between a few preset frequencies can make passive jamming attacks much more difficult.

4.4 Application of the Framework

In this section we will evaluate the developed framework by applying it to a real world networked RFID application. We will demonstrate that our framework allows for the methodical identification of the threats faced by the system in question and that it also allows the user to easily identify all the possible attacks on the system and analyse their impact, point of origin and access method. We will also demonstrate how the framework makes identifying the security requirements that need to be implemented to protect against those threats easy.
4.4.1 RFID in E-Passports

An area in which the use of networked RFID is rapidly becoming more common is the use of RFID tags in passports. Many countries—including USA, Australia and all European Union members—recently introduced e-passports containing RFID chips. The tags are normally embedded in the photo page of the passport and also contain all the information displayed on the photo page. When interrogated by authorized readers, the tags transmit personal and biometric data of the holder to the reader. For example the Australian e-passports contain all the data that’s displayed on the photo page of the passport: namely the digitized photograph, name, gender, date of birth, nationality, passport number, and the passport expiry date. The stored information is only broadcast when an authorized reader requests for that information, requests by unknown readers are ignored. One key feature of this system is that a majority of the static data of the object is stored on the tag itself and not on a back end database. Because the data stored on the RFID tag is unchanging the readers and tags are typically only read enabled and that data cannot be updated. In addition data is communicated in one-way with no data being broadcast from the reader to the tag other than its authentication info. The main differences between the RFID tags used in RFID enabled passports and other common RFID applications such as patient tracking and global supply chains is that the tags used are typically a lot more expensive [116]. Because personal and biometric data are particularly sensitive, attackers might be highly motivated to copy e-passports or use their data for identity theft. The consequences of an attack could be serious, including personal and biometric data theft, tracking of the e-passport’s owner, illegal border crossings or even detonating a bomb designed for a specific country of origin or for a specific individual, based on information emitted by the chip in his or her passport [48].

4.4.2 Applying the Threat Model to Scenario

Let’s consider the threat model for this type of system: Out of the 6 types of threats possible on networked RFID systems only 4 are of concern to this particular system. The threats (1) interception and modification of system data and communications (2) introduction of false objects into system (3) invasion of privacy and (4) denial of service still remain. But the threat of (5) RFID malware and (6) attacks by internal partners can be disregarded. The threat of RFID malware can be disregarded because these systems
use read-only RFID tags which make storing malware on them impossible. Even if an attacker managed to use a cloned or spoofed RFID tag to feed malware to a reader it would only affect that reader and terminal and would not be further propagated to other systems or tags as the tags used in the system are not updatable.

Table 4.6: Threats, weaknesses and attacks applicable to E-Passport systems

<table>
<thead>
<tr>
<th>Threat</th>
<th>Exploits weaknesses:</th>
<th>And results in</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Interception or modification</strong></td>
<td>Lack of secure mutual authentication</td>
<td>Replay attacks</td>
</tr>
<tr>
<td><strong>system data and communications</strong></td>
<td>Lack of secure mutual authentication and confidentiality</td>
<td>Eavesdropping attacks</td>
</tr>
<tr>
<td></td>
<td>Lack of secure mutual authentication,</td>
<td>Data leakage</td>
</tr>
<tr>
<td></td>
<td>integrity verification and confidentiality</td>
<td>Man-in-the-middle</td>
</tr>
<tr>
<td></td>
<td>Lack of sufficiently strong encryption</td>
<td>Crypto attacks</td>
</tr>
<tr>
<td></td>
<td>Lack of storage confidentiality</td>
<td>Physical reading of tags</td>
</tr>
<tr>
<td></td>
<td>Poor physical security of tags</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Lack of storage integrity</td>
<td>Physical writing to tags</td>
</tr>
<tr>
<td></td>
<td>Poor physical security of tags</td>
<td></td>
</tr>
<tr>
<td><strong>Introduction of false objects</strong></td>
<td>Lack of strong and secure mutual authentication</td>
<td>Tag cloning</td>
</tr>
<tr>
<td><strong>into</strong></td>
<td></td>
<td>Tag spoofing</td>
</tr>
<tr>
<td><strong>system</strong></td>
<td></td>
<td>Reader masquerading</td>
</tr>
<tr>
<td><strong>Invasion of privacy</strong></td>
<td>Mobility of tags</td>
<td>Tracking (Forward and Backward)</td>
</tr>
<tr>
<td></td>
<td>lack of proper mutual authentication</td>
<td>Tag constellation tracking</td>
</tr>
<tr>
<td></td>
<td>The mobility of the tags and the easily identifiable</td>
<td>Radio fingerprint tracking</td>
</tr>
<tr>
<td></td>
<td>radio fingerprint on low cost tags</td>
<td></td>
</tr>
<tr>
<td><strong>Denial of service</strong></td>
<td>Lack of physical security</td>
<td>Physical destruction of components</td>
</tr>
<tr>
<td></td>
<td>Low resources available on tags</td>
<td>Active jamming</td>
</tr>
<tr>
<td></td>
<td>Broadcast mechanism of communications</td>
<td>Passive jamming</td>
</tr>
<tr>
<td></td>
<td>Use of pseudonyms of some security protocols</td>
<td>De-synchronization of tags</td>
</tr>
<tr>
<td></td>
<td>Built in lock and kill commands</td>
<td>Unauthorised tag locking or killing</td>
</tr>
</tbody>
</table>

The threat posed by internal partners can be disregarded for a number of reasons. All authorized partners are allowed to read all the data stored on the system and none of the partners can update the data stored on the tag making non-repudiation and elevation of privileges impossible. In addition only reliable and trustworthy government law enforcement agencies are authorized to read the tags and there is no competition between them like between certain supply chain partners. Therefore, this removes any trust issues between the partners. By applying the threat model to the system in question
we can determine the threats, exploits and the attacks applying to this particular system as shown in Table 4.6.

### 4.4.3 Understanding Potential Attacks

Now that we have identified all possible attacks on the system it’s time to understand how those attacks can affect this particular system. This is done by applying the attack model tables to the attacks that can affect this particular system. To do this we first disregard any attacks that were eliminated when we analysed the threats to the system. Then we look at the remaining attacks and identify which of their possible impacts are applicable for the system in question. This is shown in Table 4.7.

<table>
<thead>
<tr>
<th>Attack</th>
<th>Modification</th>
<th>Interception</th>
<th>Interruption</th>
<th>Fabrication</th>
<th>Tracking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replay attacks</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Eavesdropping attacks</td>
<td></td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data leakage</td>
<td></td>
<td></td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Man-in-the-middle</td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Crypto attacks</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Physical reading of tags</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Physical writing to tags</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tag cloning</td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Tag spoofing</td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Reader masquerading</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Tracking (Forward and Backward)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Tag constellation tracking</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Radio fingerprint tracking</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Physical destruction of components</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Active jamming</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Passive jamming</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>De-synchronization of tags</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Unauthorised tag locking or killing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
</tbody>
</table>

1 9 5 3 3

In the table above the Ys represent impacts that can affect the system. An impact that cannot affect the system in question has been removed. e.g.- possibility of data modification from attacks other than man-in-the-middle can be disregarded for this
system as tags data cannot be updated. The final row of the table shows the total number of attacks that have each type of impact on the system.

By comparing the possible attacks on the system to Table 4.4 we identify the number of attacks that use each access method and the number of attacks that originate from each source as depicted in Table 4.8.

Table 4.8: Access methods and attack sources of attacks on system

<table>
<thead>
<tr>
<th>Access Method</th>
<th>Logical</th>
<th>Physical</th>
<th>External</th>
<th>Internal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replay attacks</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Eavesdropping attacks</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data leakage</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Man-in-the-middle</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Crypto attacks</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Physical reading of tags</td>
<td></td>
<td></td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Physical writing to tags</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tag cloning</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tag spoofing</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reader masquerading</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tracking (Forward and Backward)</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tag constellation tracking</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Radio fingerprint tracking</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Physical destruction of components</td>
<td></td>
<td></td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Active jamming</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Passive jamming</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>De-synchronization of tags</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Unauthorised tag locking or killing</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
</tbody>
</table>

14 5 16 0

4.4.4 Identifying Security Requirements

Finally we will use the system model of the security framework to identify the required security requirements and the possible methods with which to implement them based on the system's architecture and use. As can be seen in Table 4.9 a number of the identified security requirements for networked RFID systems is not required for this system due to some of its architectural and system features.

Next we decide how each of the security requirements identified above can be implemented for the system under review taking into consideration the system...
architecture and the attacks that the system is vulnerable to. Therefore we first look at the system architecture and identify any important and unique features of the system that may affect how the security requirements may be implemented. In this system we notice a few important features: (1) the tags are only contacted by authorized readers at very specific points (airport immigration areas) and the owner of the tag is well aware of these areas. (2) The tags need only contain static non changing data (3) the data flow in the system is only from tags to readers and (4) the tags are embedded in expensive passports and therefore can be high cost tags with more resources available on them. Based on the vulnerabilities and the identified features of the application the steps shown in Table 4.10 can be taken to implement the required security requirements.

Table 4.9: E-Passport security requirements

<table>
<thead>
<tr>
<th></th>
<th>Integrity</th>
<th>Authenticity</th>
<th>Confidentiality</th>
<th>Privacy</th>
<th>Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Tags</strong></td>
<td>N/A – Tag data cannot be modified in a meaningful manner</td>
<td>Mutual Authentication</td>
<td>Mutual Authentication, Storage Confidentiality, Access Control (not required as all partners have access to all data)</td>
<td>Anonymity</td>
<td>Physical Protection, Electronic Protection</td>
</tr>
<tr>
<td><strong>Communications</strong></td>
<td>Transmission Integrity</td>
<td>Transmission Integrity</td>
<td>Transmission Confidentiality</td>
<td>Data Leakage Protection</td>
<td>Electronic Protection</td>
</tr>
<tr>
<td><strong>Backend Components</strong></td>
<td>Mutual Authentication</td>
<td>Mutual Authentication</td>
<td>N/A – Backend data is not transmitted to the tags and therefore cannot be requested by attackers</td>
<td>N/A</td>
<td>Electronic Protection</td>
</tr>
</tbody>
</table>

The analysis done using the framework suggests that non electronic methods may allow for a much greater increase to security at a lower price. The use of a simple sleeve to ensure that the tag cannot communicate will easily eliminate most tracking, reader impersonation and replay attacks. The use of tags with very short transmission ranges will make attacks such as eavesdropping, man-in-the-middle and replay much more difficult. The more expensive nature of the tags employed allow for the use of more traditional and power security primitives such as tag side PRNG and encryption algorithms that make the implementation of secure and strong mutual authentication and transmission confidentiality and integrity relatively straightforward. The use of write
only tags will ensure storage integrity and storing the data on the tag in encrypted form will ensure storage confidentiality.

Table 4.10: Defences for RFID enabled passport systems

<table>
<thead>
<tr>
<th>Security requirement</th>
<th>Recommendation on how to implement</th>
<th>Removes or reduces vulnerability to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transmission Integrity</td>
<td>Use data integrity verification techniques (simple one-way hashing is a good recommendation) to verify the integrity of broadcasts. If any corruption is detected request a retransmission</td>
<td>Man-in-the-Middle, Desynch Attacks,</td>
</tr>
<tr>
<td>Mutual Authentication</td>
<td>Requires a secure mutual authentication mechanism that allows readers and tags to reliably authenticate each other. Can be implemented using mechanisms such as BAC (Basic Access Control) or EAC (Extended Access Control) which have been standardised for passport security. Lock commands using the MRZ (Machine Readable Zone) is also a possibility but not recommended. The use of plastic sleeves to prevent unauthorised remote access is also possible</td>
<td>Reader Masquerading, Cloning, Spoofing, Data Leakage, Man-in-the-Middle, Replay Attack, Eavesdropping, Kill/Lock Commands, Desynch Attacks, Active jamming</td>
</tr>
<tr>
<td>Storage Confidentiality</td>
<td>Because the owner of the passport may try and access the data stored on the tag to clone it or spoof the system storage confidentiality must be implemented - Storing the data in encrypted form on the tag and allowing only secured access to the keys required to decrypt that data is a option</td>
<td>Physical reading of tags, Crypto attacks</td>
</tr>
<tr>
<td>Transmission Confidentiality</td>
<td>Extremely important – needs to implement some method of secure encryption. Can be implemented using traditional security primitives due to the high cost of the tags employed.</td>
<td>Man-in-the-Middle, Data Leakage, Replay Attack, Eavesdropping, Crypto Attacks</td>
</tr>
<tr>
<td>Anonymity</td>
<td>Extremely important because passports allows the holder to be tracked. While strong mutual authentication and rotating pseudonyms can make tracking harder the nature of the system allows for a much simpler system. Using a plastic sleeve that blocks all communications to protect the passport till the owner arrives at an area where the tag needs to be read is possible. Lock commands using the MRZ (machine readable zone) is also a possibility</td>
<td>Tag tracking, Forward tracing, Backward tracing, Radio fingerprint tracking</td>
</tr>
<tr>
<td>Data Leakage Protection</td>
<td>Not really required as nature of the application means tags are polled only very infrequently making data leakage a very low threat. Also the tags can be built to be pretty sophisticated making radio fingerprint tracking harder</td>
<td>Radio fingerprint tracking, Data leakage</td>
</tr>
<tr>
<td>Physical Protection</td>
<td>Already at an acceptable level due to readers being in secure areas and the tags being attached to important objects (passports) which are secured by the owner. Physical protection from owner cannot be implemented.</td>
<td>Physical destruction of components</td>
</tr>
<tr>
<td>Electronic Protection</td>
<td>Use multi frequency RFID tags and short range transmissions to reduce the chance of jamming. Use lockout mechanisms to ensure active jamming is not possible</td>
<td>Passive jamming, Active jamming</td>
</tr>
</tbody>
</table>
4.5 Summary

Even though the deployment of networked RFID systems has greatly accelerated in the last few years there are still major concerns about the security available in systems using low cost tags. These concerns arise because of the inherent differences in various types of RFID systems and the large amount of security threats they are subject to. While there are a large number of security frameworks focused on how best to assess and secure IT systems implemented using typical infrastructure, next to no work has been done in developing a comprehensive security framework for networked RFID systems.

In this chapter we develop and present a conceptual security framework that can be used for (1) assessing the vulnerabilities of RFID systems, (2) identifying the attacks possible on them and (3) identifying the security requirements to fully secure that system. Our framework is composed of two main parts: the attack model and system model. There is also a threat model which is used to identify the vulnerabilities of the RFID system. Overall the framework developed provides a methodical manner in which possible attacks and threats on a given RFID system can be analysed and allows the user to easily identify the manner in which those threats can be removed for that particular system. The presented framework is applied to real world networked RFID system. The application of the framework illustrates how the framework can be used to assess and improve the security of networked RFID systems. It also shows how using this framework allows users and developers to identify unique mechanisms by which the security of any given system can be improved based on characteristic that are unique to that system and its architecture.
Chapter 5

A Secure Tag Authentication Protocol for RFID

As discussed in chapter four of this thesis RFID communications face a lot of threats and can be compromised by a large number of different attacks. The security of the overall RFID system, and also of all systems that depend on the data it supplies is therefore dependent on secure communications between readers and tags. Hence, in RFID systems, ensuring the security of tag-reader communications is of utmost importance. To this end, in this chapter, we develop and present an RFID security protocol that allows mutual authentication between the reader and tag as well as enabling the secure communication of tag data. The protocol presented uses a hybrid technique to provide strong security while also ensuring that the resource requirements are low. To do this it employs a mix of simple one-way hashing and low-cost bitwise operations. Out protocol ensures the confidentiality and integrity of all data being communicated and allows for reliable mutual authentication between tags and readers. The security analysis carried out also indicates that the presented protocol is also resistant to a large number of common attacks. Therefore the main research contributions of this chapter are (1) Development of a security protocol for RFID systems and (2) Analysis of the security and performance of the developed protocol to show its advantages compared to existing protocols.
5.1 Introduction

Even though the use of RFID can lead to increases in productivity and automation the nature of RFID technology dictates that this enhanced automation and productivity come at the price of an increase in security threats to that system. RFID systems use an unsecured channel to communicate between tags and readers. Consequently, RFID applications require a large number of security functionality including mutual authentication, transmission confidentiality and integrity, anonymity and availability [20]. Therefore, for RFID technology to be deployed in large scale supply chains the security issues arising from its use must be fully addressed and the required security functionality fully implemented [40]. If a provably secure RFID protocol is developed it would allow the use of RFID tags in global supply chain management leading to major advances in automation and supply chain visibility. As a result, research in developing RFID security protocols has received widespread attention in current research circles.

Unfortunately most networked RFID applications use low cost passive RFID tags that are extremely constrained resource-wise. This lack of resources on the tags mean that most tried and tested security methods such as public key encryption, cryptographic hashing or PRNG cannot be deployed when developing security protocols for RFID systems. The combination of the large number of security functionality required by the system and low resources available on the tags means that securing an RFID enabled supply chain is extremely challenging problem. While a very large number of security solutions have been proposed over the last decade none have proven to be truly effective [117]. The early RFID security protocols were developed using tried and tested security techniques and could only be successfully implemented in systems using expensive active tags. The lack of resources on low cost tags dictate that these protocols were too resource intensive to be used in most networked RFID applications that employed such tags. Therefore a new method, known as ultra-light-weight cryptography was first suggested by Juels in [56]. The basic concept behind this method was the use of simple bitwise operations to enable secure communication between tags and readers. Because of the lightweight nature of these bitwise operations, protocols developed using this technique could be implemented even on the cheapest of tags, but it also meant the strength of the security afforded was also comparatively lower. Over the next few years large numbers of RFID security protocols were developed using ultra-light-weight
cryptography. But a majority of them were later shown to be susceptible to one or more attacks rendering them unsuitable for use in securing RFID systems. Therefore, the creation of secure RFID protocols that can be implemented on low cost RFID tags still remain an open problem.

In this chapter, we develop and present a mutual authentication and tag data transmission protocol for networked RFID systems. The protocol presented uses a mix of traditional methods (simple one-way hash functions and reader-side PRNG) and ultra-light-weight methods (simple bitwise operations). Using this hybrid technique we develop a security protocol that not only supports a large number of security functions but also whose resource requirements are within the resources found on low cost passive tags. The protocol presented in this chapter enforces a number of core security concepts such as mutual authentication, communication integrity, communication confidentiality and anonymity. It is also resistant to a large number of common RFID attacks such as man-in-the-middle attacks, replay attacks, eavesdropping, tag tracing and de-synch attacks. Because we use one-way hashes for integrity verification purposes, rather than a number of weakly encrypted messages, our protocol shows a much larger resistance to crypto attacks. The contributions of this chapter include the development of new secure hybrid RFID security protocol and the primary steps in the creation of a data transmission protocol which allows the secure transmission of data other than the identification data associated with the tag. We also carry out an extensive security analysis of the proposed protocol identifying the security functionality it has and illustrating its security against a number of common attacks. The performance comparison shows that its resource requirements are equal or less than those of other RFID protocols.

5.2 RFID Attack Environment

In this section we analyse the specific threat model that applies to a networked RFID system and how that affects any security protocols that are developed for them.

5.2.1 Threat Model

The threat model for networked RFID applications is significantly different from the threat model for typical IT networks. The most significant attack vector for typical IT networks is the wired or unwired connection to the internet and therefore the external IT
infrastructure that is not controlled by the company in question. While this attack vector still exists for RFID systems, the biggest threat for networked RFID systems is posed by attacks that target the wireless communications between the readers and tags. This means that there are a number of security requirements that must be met for tag-reader communications to be considered secure. Because attackers can pretend to be authorised parties and try and communicate with tags or readers both the tag and the reader must be able to mutually authenticate each other so as to ensure that they are actually communicating with a trusted party [17]. In addition because attackers can intercept and change messages as they are travelling between readers and tags the system must be able to verify that the data received at either end has not been tampered with by unauthorised parties by providing integrity verification [31]. Finally the system must also be able to guarantee the security and privacy of the communications by ensuring data transmission confidentiality as it’s normally very easy to eavesdrop on unsecured wireless communications. Additionally the attackers can try and track tags as they travel and thereby invade the privacy of the person/company transporting that tag. To avoid this possibility the system must be able to provide tag anonymity [117]. Finally because the availability of the tags and readers are of paramount importance the security system in place must provide a guarantee of the availability of all tags and readers whenever they are required to function.

While some features of RFID systems make them more vulnerable to attacks other features have the opposite effect. In traditional attack models it is assumed that the attackers have access to the system it's attacking at all times. But for RFID systems that are being attacked at the tag-reader transmissions this is not the case. Because the RFID transmission range is limited a rogue reader has to be in close physical proximity of the readers or tags it’s trying to attack [34]. Because close physical proximity is not always possible there will always be occasions on a regular and consistent basis where tag will be able to communicate with an authorised reader in a safe environment. Additionally in supply chains most RFID tags are also in close proximity to lots of other tagged objects and tag reading is normally done via a polling transmission broadcast by a reader. Hence a majority of the time RFID communications will take place simultaneously with hundreds if not thousands of other similar RFID conversations [27]. Therefore if the attacker cannot uniquely identify the communications of a specific tag while they take
place its extremely difficult if not impossible to identify them later. Trying to identify a communication later will require the storing of hundreds of thousands of individual transmissions for review at a later date which is not possible. It’s also almost impossible for an attacker to selectively block only certain communications of an ongoing transmission as there are thousand taking place simultaneously. Therefore DoS attacks that are mounted by blocking only select transmissions of a single tag are incredibly hard to mount on large scale networked RFID systems.

5.3 A Hybrid RFID Security Protocol

In this section we will describe in detail the Hybrid RFID Security Protocol (HRSP). We will first present a high level overview of the protocol explaining what data each component must store/remember, what functionality must be implemented on the readers and tags, the overall architecture of the system and the process carried out by each component during each phase. In Section 5.3 we will present the protocol itself including details of each computation done by each component and the messages exchanged.

5.3.1 Overview of Protocol

Each tag stores a static identifier (EPC), a temporary index-pseudonym (IDS) and two keys $K_1$ and $K_2$ all of which are of 96 bit length to ensure compatibility with EPCGlobal encoding schemes. The EPC is set and stored when the tag is created and never changes. The IDS and the two keys $K_1$ and $K_2$ will change and be updated each time the tag communicates with a valid reader. In addition, if required the developers can use to store additional information about the item the tag is attached to on the tag itself as well. The database needs to store the current IDS and $K_1$ and $K_2$ along with the associated EPC for each tag. The backend database will also store the last known valid IDS and $K_1$ and $K_2$, shown by $IDS_{Old}$, $K_{1Old}$ and $K_{2Old}$. These details need to be made available to all readers that belong to any business entity in the network. Table 5.1 shows the information that needs to be stored on each component.
Table 5.1: Data storage on key components

<table>
<thead>
<tr>
<th>Data to be stored</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Readers</strong></td>
</tr>
</tbody>
</table>
| **Tags**          | EPC – Static 96 bit identifier  
                     IDS – 96 bit temporary pseudonym  
                     K₁, K₂ – two 96 bit keys that constantly change  
                     Any additional information about the item its affixed to |
| **Database (for each tag in system)** | EPC - Static 96 bit identifier  
                     IDS – Current temporary pseudonym  
                     IDS_{old} – last know valid IDS  
                     K₁, K₂ - Two 96 bit keys that constantly change  
                     K_{1old}, K_{2old} – The two last known valid 96 bit keys |

The proposed protocol employs a mix of standard security primitives (one-way hash, PRNG,) and simple bit-wise operations (XOR, bit string concatenation and splitting). Table 5.2 shows the functionality required by the three key components.

Table 5.2: Functionality of key components

<table>
<thead>
<tr>
<th>Required Functionality</th>
</tr>
</thead>
</table>
| **Readers**            | 96 bit PRNG  
                          Bitwise XOR  
                          Simple one-way hash function  
                          Bit string concatenation  
                          Bit string splitting  
                          Mutual authentication with database using standard methods |
| **Tags**               | Bitwise XOR  
                          Simple one-way hash function  
                          Bit string concatenation  
                          Bit string splitting |
| **Database**           | Data storage  
                          Location and retrieving of data as required  
                          Mutual authentication with readers using standard methods |

Figure 5.1 shows the overall architecture of the system components to run the protocol. The tags are fixed on the actual physical item and that item along with the tag is passed from partner to partner and moves along the physical RFID network. Therefore, the central key repository needs to be available to all the partners and it needs to be able to authenticate each reader as an authorised entity to access that data. Because this mutual authentication is done between a standard database server and RFID readers over a normal wired connection tried and tested authentication techniques can be used for it.
Additionally, because the requested information must travel from the key repository to the requesting reader over a normal secured channel, and that information must be protected, standard confidentiality and integrity security mechanism can be employed.

![Figure 5.1: Protocol architecture](image)

### 5.3.2 Protocol Algorithm

The protocol consists of four key stages: Initialization Phase, Mutual Authentication Phase, Data Transmission Phase and Pseudonym Update Phase which are described in further detail below. These four phases are shown in the four boxes in Figure 5.2.

#### 5.3.2.1 Initialization Phase

The initialization phase is when the reader and tag make first contact and the reader retrieves the required data from the backend database. Therefore on coming into contact with the radio wave field of an RFID reader a tag responds with its current IDS. The reader contacts the backend database with the received IDS (via the RFID middleware) and requests the key information associated with that IDS. The database locates the required key information (EPC, K₁ and K₂) based on the IDS and forwards it to the reader. If the received IDS is not recognised (i.e.: does not exists in either of EPC details tables) the database sends a terminate message to the reader which then stops communicating with that tag. The following functions and notation will be used in the proposed algorithm

\[ H(x) \] – One-way hash function on input x  
\[ PRNG(x) \] – Generation of 96 bit PRNG x  
\[ X + Y \] – XOR Function on X and Y  
\[ X \| Y \] – Concatenation of X and Y  
\[ X == Y \] - Comparison operation on X and Y
5.3.2.2 Mutual Authentication Phase

Once the reader receives the EPC, K₁ and K₂ of tag from the backend database the mutual authentication phase begins. Here the reader uses the data retrieved from the backend database to prove to the tag it is authorised to access the tags details. Then the tag uses the data stored on it to prove it is actually the tag that the current IDS correspond to. Hence, the mutual authentication phase is subdivided into two phases: reader authentication and tag authentication. First the tag needs to authenticate the reader as being a legitimate reader from the network. To enable this reader first generates the PRNG R. It then calculates M₁ = H(EPC+K₁). It also calculates M₂ = (K₁+R). Finally, it concatenates M₁ and M₂ and transmits it to the tag. On receiving M₁ and M₂, the tag uses the K₁ and EPC it has on its memory to calculate C₁ = H(EPC+K₁)
and compares it to the received $M_1$. If the two values match it means that the reader has authenticated access to the tags information (the tags EPC and $K_1$) via the backend database. Therefore, the reader is authenticated by the tag and the protocol moves on the tag authentication sub phase. If the values don’t match the protocol terminates.

During tag authentication, the tag uses the $K_1$ saved on its memory and retrieves $R$ from $M_2$ as $R = (M_2 + K_1)$. The tag then calculates $M_3 = H(EPC+K_2+R)$ and transmits it to the reader. On receiving $M_3$, the reader uses the EPC and $K_2$ it received from the backend database to calculate $C_2 = H(EPC+K_2+R)$ and compares it to $M_3$. If they match it means that the tag has the EPC ($M_1$ is a one-way hash and therefore the tag cannot retrieve the EPC from it) and $K_2$ that matches the IDS it broadcast during the initialization phase. Therefore the reader authenticates the tag and the protocol moves on to the data transmission phase. If the $M_3$ received by the reader does not match $C_2$ then the protocol terminates.

5.3.2.3 Tag Data Transmission Phase

Once mutual authentication has been successfully concluded the protocol moves on to the data transmission phase. During this phase the tag splits the tag data into separate blocks and transmits them as 96 bit packets. Once all packets have been received and the reader receives the end indicator packet it will then concatenate the data back into one block and forward it to the middleware for decoding. We have proposed three different algorithms that can be used for this phase. One has low security and high performance, one has high security and low performance and one is balanced with medium security and medium performance. In this section we present the balanced one. The other two options are presented in the next section.

In this phase the reader generates PRNG $r_1$ and calculates $E_1 = (K_1+r_1)$ and broadcasts it. Upon receiving $E_1$ the tag breaks down the data (D) to be transmitted into n parts of 96 bit length ($D = d_1||d_2||...||d_n$). Then the tag retrieves $r_1$ from $E_1$ as $r_1 = (E_1+K_1)$. It then calculates $D_1 = (d_1+r_1)$ and $H_1 = H(EPC+d_1+r_1)$ and transmits them. On receiving $D_1$ and $H_1$ the reader retrieves $d_1$ from $D_1$ as $d_1 = (D_1+r_1)$ and using that $d_1$ it calculates $DH_1 = H(EPC+d_1+r_1)$. If $DH_1$ matches the received $H_1$ the reader accepts the data if it doesn’t it requests a retransmit. If the data is accepted the protocol continues with the reader generating a PRNG ($r_2$) and the functionality explained above repeats till all n data blocks are received by the reader. On receiving the data transmission end
signal from the tag the reader concatenates all the n data blocks it received and retrieves
the complete data block.

5.3.2.4 Pseudonym Update Phase
When the reader receives the final data packet during the data transmission the protocol
moves forward into the key updating phase. In this phase the keys and IDS on both the
tag and the backend database is updated to ensure security against tracking attacks. First
the reader generates PRNG R2. It then calculates M4 = (K1+R2) and M5 = H(EPC+R2). It
then concatenates and broadcasts M4 and M5. On receiving M4 and M5, the tag retrieves
R2 from M4 as R2 = (M4+K1), it then calculates C2= H(EPC+R2) and compares with M5
received from the reader. If C2 == M5 then the tag accepts the R2 else it discards and
requests a new R2. Once a secure R2 is received the tag starts its key updating. To do
this it updates its IDS as IDS_{new} = IDS + [(R_{2left} || K_{1right}) + (K_{2left}||R_{2right})], K_{1new} =
(K_{1Right}||K_{2Left}) + R2 and K_{2new} = (K_{2Right}||K_{1Left}) + R2. Here the left and right sub texts
indicate the left half and right half of K1 and K2. It then overwrites IDS, K1 and K2 with
IDS_{new}, K_{1new} and K_{2new}. Simultaneously the reader does the same calculations and
transmits the IDS_{new}, K_{1new} and K_{2new} along with the EPC to the back end database. The
backend database overwrites the entry for that EPC in the last approved details table
with the values from the current table. It then overwrites the current table values with
the values received by the reader.

5.3.3 Data Packet Structure
All individual message packets (d_1…d_n) are 96 bits each. Each packet is composed as
follows: a header of total length 10 bits and a data for the remaining 86 bits. The first 9
bits of the 10 bit header indicate the packet number. This allows a total of 511 packets
or 44KB (511*86 bits) of data to be transmitted. If more data needs to be transmitted the
header can be lengthened to accommodate a large number of packets and therefore a
higher data size. The final bit of the header indicates if this is the last packet or not (with
1 indicating the last packet). If the final packet has less than 86 bits of data the PRNG
received will be truncated to match the length of the final packet (10+ remaining
number of data bits) and then + with the d_n to generate the final data packet of this
phase. The data packet structure is shown in Figure 5.3.
5.3.4 Data Transmission Phase Modifications

There are a number of modifications that can be applied during the data transmission phase which either increase performance or security. In this section we discuss two possible modifications. The first modification enhances security at the cost of performance while the second modification enhances performance at the cost of security. Please note that the following two sections only discuss the modified Data Transmission Phase for the algorithm. The other three phases remain exactly the same as shown in Figure 5.2 and discussed in the previous section.

5.3.4.1 Security Enhancement

One security issue present in the default data transmissions phase algorithm is that even if an attacker changes any of the messages E₁…Eₙ the tag will still send a reply. While the attacker cannot use the reply D₁…Dₙ to gain any information this still is not optimal from a security standpoint. Therefore to ensure that the tag does not reply if they receive a corrupted or modified E₁…Eₙ we propose the following modified data transmission algorithm. Here for added security each E₁ to Eₙ transmitted during the data transmission phase will be accompanied by HR₁ to HRₙ, and is calculated as H(EPC+rₙ). Now when the tag receives E₁ and retrieves r₁ it will first calculate H(EPC+r₁) and match it with the HR₁ received from the reader. If the received HR₁ does not match the calculated H(EPC+r₁) it means that E₁ was changed during transmission and therefore nothing will be transmitted. This process is shown in Figure 5.4.
Figure 5.4: Security enhancement for data transmission

Please note that while this process increases the security of the protocol (no data is transmitted if the received $E_1$ is found to be modified or from an unauthorised reader) it also increases the overhead placed on the tag and is therefore not recommended for performance sensitive applications.

5.3.4.2 Performance Enhancement

The default algorithm for data transmission presented in Figure 5.3 can be made more performance efficient if security is not a very high priority. In the original algorithm a new random number is generated and transmitted for each data packet and the tag must use the received message to retrieve the correct random number before it can transmit each data packet. Therefore if the number of computations during data transmission is a problem, or if time (through-put) is a factor, a single PRNG can be used for the whole data block $D$. This which will decrease the computations required by the reader and the tag but it will also increase the vulnerability the protocol has to brute force attacks due to the longer time the PRNG is valid. Figure 5.5 shows the modified version of the algorithm which can be used for data transmissions.
Figure 5.5: Performance enhancement for data transmission

This enhancement to the data transmission phase greatly reduces the performance overhead as well the number of messages that must transmitted by the reader. Here the reader generates and transmits a single PRNG \( r_1 \) at the start of the phase and that \( R_1 \) is used for all data packets that are transmitted.

5.5 Security Analysis and Comparison

In this section we identify some of the common weaknesses found in RFID security protocols and analyse how our protocol avoids those weaknesses. We also look at some of the main security concepts that the protocol enforces. Finally we analyse the security of the proposed protocol against a number of common RFID attacks.

5.5.2 Protocol Security Properties

The hybrid mutual authentication protocol proposed in this paper supports a large number of commonly required security features such as anonymity, mutual authentication, confidentiality etc [118]. To achieve this security we have used a number of security techniques which include (1) random numbers to ensure freshness of messages and ensure confidentiality (2) a simple one-way hash function to ensure strong integrity verification at minimal performance overhead (3) bitwise encryption
using the XOR operator on data and secret keys or random numbers to ensure confidentiality and (4) temporary pseudonyms and keys to allow for secure mutual authentication of tags and readers while protecting against tracking attacks.

5.5.2.1 Mutual Authentication

Our protocol ensures the tag and readers can mutually and reliably authenticate each other. The tag is able to authenticate the reader based on $M_1$. Because the tag never transmits its EPC and $K_1$ only readers which really belong to the system will have access (via the backend database) to the EPC and current $K_1$ of the tag. Therefore, if $M_1$ matches the $H(EPC+K_1)$ calculated by the tag it can be sure that the reader retrieved the correct EPC and $K_1$ from the backend database and that therefore it’s authorised to access the tag and its data. The reader is able to authenticate the tag using $M_3$. If $M_3$ matches the $H(EPC+K_2+R)$ it means the tag knows the EPC (it can’t retrieve the EPC using $M_1$ as that’s a one-way hash) and $K_2$ matching the IDS it transmitted earlier. It also ensures that $M_3$ is not a replay of a previous transmission as $R$ ensures the message is new. Therefore, both the tag and the reader can mutually authenticate each other.

5.5.2.2 Data Confidentiality

In the proposed protocol all public messages are either hashed using an one-way hash algorithm (such as the one proposed in [77]) or composed by using bitwise operations employing a random number ($R_1, R_2, r_1, r_2, ..., r_n$) or bitwise operations employing a secret key known only by legitimate tags and readers ($k_1, k_2$). Therefore it’s nearly impossible to obtain the data that’s being transmitted (only 1 message is transmitted when communicating any given data making crypto attacks impossible and the chances of a brute force attack working on any of the messages where a bitwise operation has been employed using either a random number of an unknown key is $1/2^{96}$). Also because the hashes that are used to verify the integrity of the data always contain the EPC as well the attacker cannot use the hash to verify that the data he retrieved using a brute force attack is valid. In addition, because the random numbers change after each block of data and the keys ($k_1$ and $k_2$) are updated after each communication session, the brute force attack will have to be carried out a large number of times before the attacker can gain any significant knowledge of the data.
5.5.2.3 Data Integrity
Whenever a message containing data such as M₂, D₁ or D₂ is transmitted another hashed message M₁, H₁ or H₂ etc. is also transmitted. This hashed message allows the protocol to ensure the integrity of received data. In the data transmission phase the method in which PRNGs are used makes it impossible for an attacker to compromise the integrity of data without the reader realizing it. Imagine that an attacker changes the message E₁ during the data transmission phase. Then the random number retrieved by the tag is not the same as the random number that the reader generated and transmitted. Therefore, when the tag generates and sends back D₁ the data the reader retrieves from D₁ will be different from the data that the tag encrypted (as the r₁ used by the tag is different from the r₁ used by the reader). Therefore when the reader generates H(d₁) it will not match the H(d₁) transmitted by the tag and therefore the reader will discard that data and request a retransmission.

5.5.2.4 Tag Anonymity
Each tag updates its IDS, K₁ and K₂ after each authorised communication session. This update also involves the random number R₂ generated by the reader and securely communicated to the tag. Therefore, the IDS that are transmitted by a tag are constantly changing making it impossible to track it for any significant length of time. Also as previously mentioned the only piece of tag data that never changed (the EPC) is never transmitted without being hashed first using either a random nonce or a constantly changing k₁ making it impossible to use data leakage or any other types of repeated attacks to try and infer the EPC of the tag. Also any public messages are always encrypted by using bitwise operations and a random nonce which further makes impossible to track the tag using any other data that it transmits. Of course the attacker can track the tag between two successful authentication and update transmissions between the tag and authorised readers but this is a very limited form of tracking.

5.5.2.5 Backward Security
Backward security is a property that guarantees the security of future communications even when a tag is compromised. Imagine an attacker steals a tag manually reads its data (IDS, EPC, K₁, K₂) and then reintroduces it into the network. Because the attacker has access to the tags current (IDS, EPC, K₁, K₂) he will be able to track the tag from
there on and also understand the data that’s been communicated between tags and readers. But as stated in the attacker model it’s highly unlikely that a single attacker would have constant uninterrupted access to any given tag. Therefore when the tag is finally able to have a conversation with a reader where the attacker cannot access the communication (inside a large warehouse for example) the tag will update its current information and the attacker will lose access to the tag and its communications from there on.

5.5.2.6 Forward Security
Forward security is a property that guarantees the security of past communications even when a tag is compromised. Because the tags IDS, K₁ and K₂ are updated after each authenticated communication session using random numbers, even if an attacker manages to gain the current values of IDS, K₁ and K₂, there is no way he can derive the past values of IDS, K₁ and K₂ using the current values. And without those past values of IDS, K₁ and K₂ he cannot gain access to the data communicated in past sessions. The only non-changing value stored on the tag, the EPC, is never transmitted without being hashed with another random number or constantly changing key first which means it can never be identified based on tag reader communications. Also unlike some other protocols we only store the current value of the IDS on the tag therefore even if the tag is physically compromised the attacker gains access to only 1 set of values and keys not two.

5.5.3 Protection from Common RFID Attacks
The protocol also displays immunity to a large number of common attacks. In particular we claim that our scheme achieves the following security properties:

5.5.3.1 Protection against Replay Attacks
There are a number of points in the protocol where the attacker may try to use a replay attack. The attacker may try and store the IDS transmitted by the tag or the message M₁ transmitted by the reader and replay them. But because the IDS, K₁ and K₂ are updated using a random number (R₂) after each authenticated transmission sessions that would not work. The attacker may also try and block the updating phase and try and use the old IDS or recorded M₁. If the attacker tries to block the update phase and replay the
IDS to a reader of the network, this is equivalent to the attacker transmitting a random
IDS which happens to be part of the network: therefore because the attacker does not
have the EPC or the K₁ or K₂ associated with that IDS they will not be able to make any
sense of the received messages M₁ and M₂ and the attack would fail. If the attacker tries
to block the update phase and replay M₁ and M₂ to a tag the tag would authenticate the
reader and transmit M₃ but as the attacker cannot gain again any information from this
(M₃ is a hashed message) and therefore, he cannot continue the attack to try and steal
information during the data transmission phase (K₁ and r₁ are required to understand the
data transmitted during this phase and attacker does not gain access to K₁ at any point
during the replay attack). Hence, the attacker gains nothing from this attack. Then
because he is now authenticated by the tag the reader may try and mount a de-synch
attack on the tag. To do this he will now replay M₄ and M₅. But keep in mind at this
point the back end database has already been updated to the new values of IDS, K₁ and
K₂ based on the random number value contained in M₄ and M₅. Therefore this attack
will only serve to update the tag to the latest values of IDS, K₁ and K₂ as stored by the
database rather than de-synch it.

5.5.3.2 Protection against De-synch Attacks

There are two main ways in which a de-synch attack can be mounted on a networked
RFID system. The attacker can either block a key update confirmation between a tag
and reader and force only the tag or the reader to update thereby de-synching them or he
can send a false update message to either the tag or reader making them update while
the other side doesn’t. The protocol proposed by us is secure against both these types of
attacks. If the attacker blocks the update messages M₄ and M₅ the backend would be
updated while the tag would not. But the backend always stores the current IDS, K₁ and
K₂ and the last verified IDS, K₁, K₂ in two separate tables. Therefore if it doesn’t
receive an IDS it recognises in the current IDS table it will then look in the old IDS
table for a matching entry. Because the sent IDS and the K₁ and K₂ stored on the tag
matches the details in the old IDS table the system would recognise and authenticate the
tag. It will then overwrite the details of that tag in the current IDS table with the data
from the old IDS table and then continue as normal. Therefore a de-synch attack based
on blocking key messages will not work on the system. Now let’s consider an attacker
who tries to de-synch the system by transmitting false update messages M₄ and M₅.
is used to transmit a PRNG (R₂) securely to the tag and M₅ is used to verify the integrity of that R₂. Because an attacker does not have access to K₁ the M₄ they transmit will be \(X + R₂\) (where X is random 96 bit string which is not K₁) therefore when the tag retrieves R₂ from the false M₄ it will not match the R₂ that the attacker generated. Hence the M₅ transmitted by the attacker will not match the H(R₂) generated by the tag (as the R₂ used by the attacker in generating M₅ is not the same as the R₂ retrieved by the tag from the false M₄ and used to generate H(R₂)) and the tag will not go ahead with the update and will request for another update variable (R₂). Also a combination attack: blocking the messages and replaying them later will not work as discussed in the replay attack section.

### 5.5.3.3 Protection against Man-in-the-middle

The proposed protocol provides strong mutual authentication, transmission confidentiality and transmission integrity. Therefore man-in-the-middle attacks do not work against it. Any changes to the data will be picked up the integrity checks or they will change the random numbers in such a way as to make the attacker and tag have different random numbers thereby making it impossible for the attacker to “decrypt” the messages. And because the attacker cannot gain access to K₁ and K₂ or even the random numbers used he will be unable to decrypt normal messages that pass through him.

### 5.5.3.4 Protection against Eavesdropping Attacks

As explained in the data confidentiality section above only readers and tag with access to the current keys of the tag can decipher the messages broadcast by the system. Our system ensures that only the tag and authorised readers will have access to those keys. Therefore a potential attacker who eavesdrops on the broadcasts will be unable to decipher them making this type of attack futile.

### 5.5.3.5 Protection against Crypto Attacks

By ensuring that all public messages are a XOR of the data and either a PRNG or a secret key we ensure that the only way an attacker can decipher the data is through brute force attack. And even if a brute force attack is to work the use of the unknown EPC (which is only broadcast in one-way hashed messages) in all hash messages used for integrity verification ensure that the attacker cannot use the hash he intercepted to check the correctness of the data he got through brute force. As an example imagine the
attacker intercepts $D_1||H_1$ during the data transmission phase. He then mounts a brute force attacker to retrieve $d_1$. Once he has retrieved a possible value he has to check that against something to ensure that it’s indeed $d_1$. But the $H_1$ which the system uses to check the transmission integrity includes the EPC in it. Because the attacker does not have the EPC he cannot generate $H(\text{EPC}+d_1+r_1)$ to compare to the received $H_1$ and therefore he has no way of knowing if the $d_1$ he retrieved is the real value or a false value.

5.5.3.6 Protection against Data Leakage Attacks

When we refer to data leakage we refer to specific type of attack on confidentiality that can be mounted on RFID systems. Unlike in other attacks such as eavesdropping, man-in-the-middle where the attack uses only one communication session a data leakage attack is mounted over a number of different communication sessions. In these attacks the attacker eavesdrops or intercepts the communications between the same tag and different readers over a number of different communications sessions and uses that data to fully or partially compromise the security of the system. To ensure full protection against these types of attacks the protocol must ensure that the messages used over multiple different sessions cannot be used to compromise the confidentiality of the data being communicated. In our protocol we ensure that when most data is communicated it is either as a hash or encrypted using a random number. This ensures that these communications cannot be used to mount data leakage attacks as it’s impossible to retrieve any usable information (partial or whole) from them. Additionally the only message which is not encrypted with a random number or hashed: the current IDS transmitted in the initialization phase, is updated using particularly complex update methods dating the key updating phase as follows:- $\text{IDS}_{\text{new}} = \text{IDS} + [(R_{2\text{left}} \parallel K_{1\text{right}}) + (K_{2\text{left}} \parallel R_{2\text{right}})]$. Therefore it is not possible for an attacker to intercept two consecutive broadcasts of the current IDS and use that to gain any valuable information.

5.5.4 Security Comparison

As the above analysis shows the proposed protocol not only supports a large number of the identified RFID security requirements but it is also secure against a large number of common attacks. Table 5.3 shows the comparison of the security properties of some common RFID security protocols with the proposed protocol. The table clearly shows
that the proposed protocol has much higher security than the other protocols previously proposed in literature.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Mutual Authentication</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Key Confidentiality</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Key Integrity</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Anonymity</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Data Confidentiality</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Data Integrity</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Replay attacks</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Eavesdropping attacks</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Data leakage attacks</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>De-sync attacks</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Man-in-the-middle attacks</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Crypto Attacks</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Forward tracking</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Backward tracking</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Additionally our protocol is the only one that looks at securely transmitting tag data other then the identifier and related keys securely. With more and more systems moving to store additional data on the tag this functionality is very soon going to become a very important security requirement for any RFID system [17].

5.5.5 Potential Weaknesses

As the security analysis carried out above illustrates the protocol proposed by us offer a large number of security functionality and is resistant to a large amount of common attacks on RFID systems. But there are still two weaknesses that may be of consequence in it. In this section we will discuss these weaknesses, analyse them in context of the attack environment and architecture of networked RFID systems and identify ways with which to minimize their impact on the overall system.
5.5.5.1 Limited Tag Tracking

One of the main issues with networked RFID systems is the ability for attackers to track the tags and thereby invade the privacy of the tag holder [51]. While our proposal is immune to most forms of forward and backward tracking as well as tag tracing there is still a limited form of tracking that can be mounted on tags using our protocol. For one attackers can track tags between two authorised readings as the tag IDS and associated keys are only updated after each authorised communication. Fortunately the impact of this type of tracking is limited and the amount of information that can be derived from it is small. But an attacker can use blocking attacks to extend this attack and increase its impact and the amount of information gained. By blocking the last update message ($M_4$ and $M_5$) the attacker can keep the tag from updating its IDS and thereby indefinitely track its movements. To avoid this we suggest a simple mechanic as follows: the system can be made to track the number of authorised conversations after which a given tag does not update its IDS and $K_1$ and $K_2$ (based on the number of consecutive times the back-end database uses the old authorization data instead of the current authentication data). Then a tag can be tagged as potentially at risk after a certain threshold has been reached. Once the tag is tagged as "at risk" either the system can warn the operators and request that the tags at be manually updated or the tag could be automatically updated by running the update phase of the algorithm repeatedly till the tag does update. Either way the attack environment of RFID systems makes this type of attack extremely hard to mount: as explained in Section 5.2 most tag communications take place simultaneously with hundreds if not thousands of other tag communications over an open channel. Therefore selectively blocking the update signal for just 1 tag repeatedly for a large number of transmissions is next to impossible. In addition because the range of passive RFID tags are so limited it’s unavoidable that the tag will be quite frequently able to communicate with authorised readers in places where the attacker cannot gain access to the communication due to distance requirements. These two features of the attack environment make it so the possibility of this type of attack is minimal and not too high a threat.

5.5.5.2 Jamming Attacks

Another potential threat to our protocol is active and passive jamming which leads to denial of service [51]. Passive jamming carried out by flooding the channel with so
much interference that the communications cannot propagate through, cannot be prevented at protocol level. Fortunately this type of attack can be easily detected because all transmissions will be blocked. To handle this type of attack the system can use RFID tags and readers that use multiple different radio frequencies making it harder to attack the system. Increasing the amount of power used to generate the reader signals when interference is detected can also provide a certain amount of protection against this type of attack. Attackers can also try and mount active jamming attacks. These are normally mounted by flooding a specific tag with data requests overloading its capacity to respond to legitimate readers. This kind of attack is harder to detect. To prevent this type of attack we propose a lockout system whereby if any given reader does more than a set number of requests for any given time the tags will temporarily stop responding or accepting data from that reader. While this will not completely stop the issue it will alleviate the impact such attacks have on the system. As future work we plan to further develop this idea and work on preventing active DoS jamming attacks on RFID systems.

5.6 Performance Analysis and Comparison

We will now analyse the proposed protocol performance based on a number of different performance metrics including storage cost, message cost and computational costs.

5.6.1 Computational Cost

All computationally expensive PRNG generation is carried out by the reader. Therefore, the protocol only requires the use of one-way hashing, which is a lot less computationally expensive than cryptographic hashing or keyed hashing, and simple bitwise operations on the tag side. Now while hashing is a bit more computationally expensive than the simple bitwise operations used in ultra lightweight protocols it can still be implemented in under 2K gates. As far back as 2004 one-way hash functions which require only 1.7K gates were developed [60, 77]. Because the current EPCGlobal standard RFID tags can have up to 3K gates dedicated for security implementation the use of simple one-way hash functions is possible on them.
5.6.2 Storage Requirements

Each tag stores its EPC and one tuple consisting of (IDS, K₁, K₂). All of these values will be on 96 bit length making it consistent with the EPCglobal numbering scheme. The EPC is static and therefore can be stored in the ROM area of the tag. IDS, K₁ and K₂ will be stored in the rewritable area as they need to be updated. Also the tag needs to remember at any given time a single random number of 96 bits. Therefore, the total storage capacity required for the protocol overheads (IDS, K₁, K₂ and temporary random number) is just 384 bits. Because most low cost tags nowadays have significantly more than that in memory this is not an issue.

<table>
<thead>
<tr>
<th>Table 5.4: Performance comparison of some common RFID Protocols</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Storage</strong></td>
</tr>
<tr>
<td>-------------</td>
</tr>
<tr>
<td>Tag</td>
</tr>
<tr>
<td>Reader/ database</td>
</tr>
<tr>
<td>Messages</td>
</tr>
<tr>
<td><strong>Hash or CRC Operations</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>PRNG Operations</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>Simple Bitwise Operations</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>Operations</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
5.6.3 Communication Cost

The proposed protocol performs mutual authentication and updating using only 3.6L messages (where L is the key length) which is lower than what’s used by most other protocols. When carrying out the data transmission, the protocol at full security requires the overhead of 1.3L (for every data packet of 1L length transmitted a PRNG of 1L length and a hash of 0.3L are sent in addition to the data packet itself) for every L data sent. At lower security the protocol requires a flat 1L at the beginning and then an additional 0.3L (hash) for every L data sent.

As is illustrated in Table 5.4, our protocol compares very well with other RFID security protocols. In areas of storage space and number of messages our protocol requires equal or less space and messages for the authentication of the tags and readers. Our scheme does not require PRNG or keyed or cryptographic hashing on tag side, and therefore is less resource intensive than the mutual authentication protocol and the hash based scheme. Our scheme uses one-way hashing on the tag side compared to the ultra-light-weight protocols but requires significantly less bitwise operations (2 hashes and 6 XOR operations on tag for our scheme compared to the 40+ bitwise operations, 14 Rotation operations and the proprietary iterative mix bits algorithm for the Gossamer protocol). Also unlike some other protocols which use cryptographic or keyed hash algorithms we employ a simple one-way hash which requires significantly less power and resources.

5.7 Summary

Development of security protocols for RFID systems poses two main challenges (1) Developing protocols that can support the large number of different security functionality required by networked RFID systems and are resistant to the large number of different attacks possible on them [119] and (2) Ensuring that their computing requirements are low enough that they can be implemented on low cost RFID tags with highly constrained computing and power resources. Most security protocols employing traditional cryptographic methods such as encryption algorithms, CRC, tag side PRNG and keyed and encryption hashes fail on the second count. Solutions employing only bitwise operations have so far failed to remain secure for any significant time span with most of them being compromised within a few months of being published.
Therefore, we propose and develop a hybrid security protocol which uses a mix of traditional cryptographic primitives (one-way hashing and reader side PRNG) and ultra-light-weight techniques (XOR operations) to develop a highly secure RFID security protocol which still remains light-weight enough to be implemented on EPCGlobal Class-1 Generation-2 tags. The detailed security analysis carried out and presented prove that the proposed protocol implements a large number of networked RFID security requirements including the confidentiality and integrity of all data being communicated as well as providing reliable mutual authentication and anonymity. The analysis also shows that the protocol is resistant to most of the identified attacks possible on networked RFID systems including but not limited to DoS attacks, replay attacks, man-in-the-middle attacks and de-sync attacks and forward and backward tracking. Also to the best of our knowledge it is the first and only RFID security protocol to look at the secure transmission of tag data other than the EPC and associated pseudonyms and keys. In addition the performance analysis and comparison show that the computing and memory requirements of the proposed protocol are well within what’s available on the current EPCGlobal Class-1 Generation-2 tags. It also shows that the proposed protocol’s performance compares very favourably with the performance of a number of other recently proposed security protocols for RFID.
Chapter 6

RFID Malware Detection

In this chapter we propose a policy based, dual pronged detection and prevention method for tag based malware attacks on RFID systems. Our system is optimized for the architecture of RFID systems and consists of a query structure matching method that uses simple string comparisons. It provides strong security against a majority of the SQLIA types possible on RFID systems. Because all currently identified RFID tag malware is based on SQLIA this system provides security against all known RFID malware. Our research also looks at detecting and preventing tag based second order SQLIA which is a major gap currently in the literature in this area. To provide security against this type of SQLIA, the only attack type that the query pattern matching cannot detect, we include a fairly straightforward input data validation and sanitization technique. The preliminary evaluation of our query matching technique is very promising, showing 100% detection rates and 0% false positives while employing a comparisons technique which is much simpler than the ones used by most existing approaches. We have also justified how input validation and sanitization is very possible in the specific architecture of RFID systems even though they are not a practical solution for web based applications. Therefore the research contributions of this chapter are (1) Review of tag based RFID malware and analysis of their differences to traditional SQLIA attacks, (2) Creation of a policy based RFID malware detection and prevention system (3) Analysis of the proposed technique to illustrate that the security it provides is equal or greater to existing methods while using a simpler approach.
6.1 Introduction

One major security concern for RFID systems is their recently identified vulnerability to tag based malware. In [111, 112] the authors first identified the possibility of RFID based malware attacks. In [26] the vulnerability of RFID systems to SQLIA were proven and the authors demonstrated how a fully functional RFID virus can be used to infect and spread via SQLIA. These papers not only highlighted the possibility of SQLIA attacks compromising RFID systems but also illustrated how the specific architecture of RFID systems makes RFID malware a possibility. These RFID malware spread by infecting new tags and databases just like normal computer viruses, but unlike normal viruses they use SQLIA as their attack vector. The nature of RFID systems means that if a RFID network was infected with such a virus it would spread rapidly. In a global supply chain application the virus would spread to hundreds of thousands of tags and hundreds of different systems very quickly. Currently all of the more complex identified types of RFID malware are based on SQLIA [120]. And therefore if we can successfully defend against tag based SQLIA then we can defend against a majority of RFID malware. SQLIA refers to a specific type of malicious attack in which the data provided by the user (or stored on the RFID tag) is integrated into a SQL query so as to make that input be treated as part of the code rather than part of the input. Successful SQLIA can have a range of detrimental impacts on the overall system including but not limited to: allowing attackers to corrupt the information stored in the backend database, compromise confidential information or in RFID systems act like a virus and propagate from one infected tag to backend databases and from there to new tags [121]. Even though SQLIA has been a major issue for web based systems for a number of years the possibility of them impacting RFID systems was not considered till recently.

While the vulnerability that lead to SQLIA are well understood, they still persist because there are no effective techniques for detecting and preventing them [92]. A number of different techniques have been proposed for SQLIA detection and prevention in web applications, but none of them have been completely effective. In addition the differences in the architecture of web applications and RFID systems mean that most of the approaches proposed for web systems do not work with RFID systems [78]. The lack of any research in detecting RFID tag based SQLIA and the drop in updatable RFID tag prices have compounded the issue of tag based RFID malware. The increase
in storage capacity of said tags have motivated users to store more and more data on the
tag itself for ease of access [32]. This in turn has increased the ability that attackers have
of leveraging those tags to try and mount SQLIA based malware attacks on RFID
systems thereby increasing the potential threat they pose. Hence removing the
vulnerability of RFID systems to SQLIA and therefore malware is currently a very high
priority.

The method we proposed is based on existing SQLIA detection techniques but
modified and optimized for RFID systems based on three key features they exhibit. (1)
RFID tag data is highly structured and the dynamically generated SQL queries are built
by a single point in the middleware [122]. Therefore intercepting, validating and
sanitizing that data is much easier compared to doing it in web based systems. (2) All
dynamic queries have a structure that is defined by the programmer, and SQLIA, by
injecting additional SQL code would change the structure of the generated query from
the legal structure defined for that query by the programmer. (3) Networked RFID
systems require very high tag read through-put which require that the SQLIA defence
mechanism be as simple and efficient as possible.

The proposed technique is a dual pronged defence mechanism for protecting RFID
systems from tag-based SQLIA and is based around policies set by the system
developer. Our methods consists of (1) Validation and sanitization of RFID based data
to ensure that no “bad” data is used in generating dynamic queries and (2) Matching the
structure of those dynamic queries with the legal structure as defined by the
programmers using simple string comparison. Each mechanism consists of two phases.
First, a static analysis phase to identify the formatting and content policies for the data
stored on the RFID tags and to identify the legal query structures for dynamic queries.
Second, a dynamic runtime monitoring phase during which the data is validated and
sanitized according to the policies developed and the structure of the dynamic queries is
matched with the legal structure defined from those queries. In this paper we also
present the results of the evaluation which were promising with the system giving a
100% detection rate from the tested attacks and a 0% false positive rate over around 170
attacks and 130 non malicious queries.

Therefore the main contributions of the work presented in this chapter are: (1)
analysis of tag based malware in RFID systems and the identification of key
requirements of any defence techniques, (2) Review of current SQLIA prevention techniques and identification of their weaknesses in relation to RFID networks, (3) Creation of a SQLIA defence mechanism for networked RFID systems that meets the requirements of RFID systems and (4) Evaluation of the proposed system to quantify its success rate.

6.2 SQL Injection Attacks

Before we describe our proposed defence system will first introduce SQLIA’s. We will then look at how SQLIA are mounted on RFID systems and then analyse the mechanism behind tag based RFID malware. We will also analyse the main differences in web based SQLIA and RFID based SQLIA and discuss how those differences affect any proposed mechanisms for RFID malware prevention.

6.2.1 Definition of SQLIA

An SQLIA occurs when an attacker successfully changes the logic, semantics or syntax of a legitimate SQL query by inserting additional SQL keywords and operators into it in such a manner as to make the changed query compromise the security of the database in some manner when executed [123]. This definition includes all types of SQLIA included in including but not limited to tautologies, stored procedures, piggy backed queries, union attacks and attacks by errors.

Most current IT systems employ databases which use SQL as the main query language. To allow the use of these databases’ to the maximum most systems allow the users to input various parameters that are then used as part of an automatically generated SQL query. These queries are then forwarded to the database and the system carries out various processes based on the outcome of that query. SQL injection attacks (SQLIA) are a unique form of malware that depends on injecting malicious SQL code into normal SQL queries and therefore the database. Imagine a web page which takes in a user name and password and displays the users profile information. Once the user inputs a user name and password and clicks the submit button the page will run a script that dynamically generates a SQL query which contains both the user name and the password input by the user. The auto generated SQL command will be something like

```
SELECT * FROM users WHERE login = 'usrnme' AND password = 'pswd';
```
Where usrnme is the username input by the user and pswd is the password input by the user. Now imagine the attacker inputs “hsf” or 1=1 –“ as username and “gshg” into the web fields. Then the resulting query is:

```
SELECT * FROM users WHERE login = 'hsf' or 1=1 --' AND password = 'gshg';
```

Once this query has been generated it will be sent to the database for evaluation. The database will interpret everything after the WHERE keyword as a conditional clause and everything after – would be ignored as a comment. Because the “or 1=1” part of the query is always true its inclusion into conditional clause causes the statement to always evaluate to true. Therefore on receiving the above query the database would return all details of all users to the attacker after executing this query. This is just one simple example of the wide range of possible attacks possible via SQLIA. For full details refer to [80].

### 6.2.2 Malware in RFID Systems

RFID malware are malicious code that stored on RFID tags that can propagate to the back end database and other tags. When executed the malware has a detrimental effect on the overall system. Figure 6.1 shows a typical RFID system and illustrates how RFID malware can be used to infect the system via SQLIA using tags for the malicious data input.
RFID tags store data that is read by readers. This data is then forwarded by the readers to the middleware. The middleware uses the received data to build dynamic RFID queries (queries which have the tag data embedded into them). These queries are then forwarded onto the database. These dynamically generated queries can either retrieve data from the database or update the existing data. Any results of the queries are sent back to the middleware but they are not forwarded to the tags. When queried by business applications the middleware retrieves the information as required from the database and forwards it to the business applications.

When an attacker wants to mount an SQLIA on this system he saves the malicious data on the tag itself. This can be done by either physically updating the systems tags or by remotely updating the system tags. In addition the attacker may also try and introduce completely new tags which contain malicious data into the system. Then when a reader polls a tag containing the attacker’s data it will read and forward that malicious data to the middleware. The middleware will in turn then use that data to build dynamic SQL queries (queries which are built at runtime based on the data which is received from the tags) which are malicious and forward them to the database for execution [111]. These queries will command the database to carryout processes which
compromise it or the data stored in it. In addition properly written data will act as an RFID virus and propagate to the database. Later on additional tags may be updated with the corrupted data stored in the database. If the malicious data is written correctly this will cause the recently updated tag to also become infected and it will in turn go on to infect and compromise other system’s middleware and databases. This kind of RFID SQLIA malware can propagate and infect a large number of tags and databases compromising them all [79].

6.2.3 Web Based SQLIA vs. RFID Based SQLIA

Because of the architecture differences in web and RFID systems attacking an RFID based system with a SQLIA is a lot more difficult proposition that attacking a web based system. In web based systems, because the dynamic SQL generation is carried out on the user machine, the data needs to be validated on the user’s machine. Moreover there can be very large number of constantly changing and expanding input sources for parameters in the form of interactive web pages. These web sites and pages can be built and maintained by external companies who are not as concerned about the security of the third party database they are accessing as the owner of said database is. This makes it extremely difficult to ensure the proper validation of all inputs possible in to the system [80]. But in RFID systems data is received only from RFID tags and the dynamic queries are generated at a single point in the RFID middleware. Therefore proper validation can be carried out by a single point in the middleware as well. In addition unlike in web applications where the input can vary considerable the data received from RFID tags have a much more limited scope [122]. Hence setting up data standards and checking for those are also significantly easier in RFID systems. Consequently input validation and sanitization is much easier in RFID systems compared to web based systems. But RFID data validation has its challenges as well. One main difference in web data and RFID data is that web data is normally input as discrete blocks with each data field being input separately. But in RFID tags the data is stored as one contiguous block and it is up to the middleware to actually identify each field and separate the data block into its component field. Therefore decisions on how the data will be stored on the tag and what formatting standards will be used have to be made and enforced if RFID malware is to be successfully defended against [2].
Another key feature of RFID systems is the limited number of data stored on the tag and the limited access given to the tag [55]. In web based systems the web clients may have full administrator access and be able to input a vast number of different parameters for query generation. This allows the setting of very strict data standards in RFID systems as the type, size and amount of data expected from each tag is known in advance. This along with the single generation point for dynamic queries makes it much easier to validate and sanitize input data coming from the RFID tags relative to input data from web based systems. In addition the numbers of different types of SQL queries that are automatically generated by the middleware are also much lower than the number of different queries generated in a web/application based system [78]. Because of the limited number of queries and the fact that all those queries are set by the company itself and not outside companies makes the number of valid structures possible for the dynamically generated queries very low and easy to track.

Table 6.1: Differences in web based SQLIA and RFID SQLIA

<table>
<thead>
<tr>
<th></th>
<th>Web-based Systems</th>
<th>RFID Systems</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Query generation location</strong></td>
<td>External (at clients computer)</td>
<td>Internal (in middleware)</td>
</tr>
<tr>
<td><strong>Number of origin points for generated queries</strong></td>
<td>Very large (large number of different web pages and web sites)</td>
<td>Single (only the middleware)</td>
</tr>
<tr>
<td><strong>Number of different valid query structures based on input</strong></td>
<td>Large and constantly changing</td>
<td>Small and fixed</td>
</tr>
<tr>
<td><strong>Input output capabilities of attack origin</strong></td>
<td>The web browser is both a input and output device letting the user input parameters and then view the results of the generated queries</td>
<td>The tags are treated as simple data containers. They hold data that can only be updated by the readers of the system. They cannot request for data and they do not receive any feedback</td>
</tr>
<tr>
<td><strong>Data formatting and standards</strong></td>
<td>Hard to set due to large number of different input points and input values possible</td>
<td>Can be easily set as tags contents are known well in advance</td>
</tr>
<tr>
<td><strong>Number of possible inputs for query generation</strong></td>
<td>Very large and constantly increasing as more and more web pages and web sites are created which query the database</td>
<td>Small and known in advance</td>
</tr>
<tr>
<td><strong>Access to query structures by attacker</strong></td>
<td>Accessible as the query generation scripts must be sent to the attackers web browser</td>
<td>Not accessible by attacker as all query generation is done by middleware</td>
</tr>
</tbody>
</table>

Finally the RFID tags are treated as simple data containers as opposed to the web pages in web based systems which are treated as input output devices. This means the tags can
only provide raw data not queries. They also cannot perform or request for any other processes or data. Also RFID tags origins do not receive data based on queries sent and cannot retrieve data. The data upgrades are done by the middleware in its own choosing [120]. Additionally, unlike in the web based systems where the queries are generated on an external client machine the dynamic queries in RFID are all generated by internal servers. Therefore it is not possible for potential attackers to gain access to the query structures beforehand making it much more difficult for them to create the data they must inject for successful SQLIA.

Table 6.2: Types of SQLIA

<table>
<thead>
<tr>
<th>Type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tautologies</td>
<td>Applicable While tautologies are most commonly used to extract data they can also be used to bypass filters in UPDATE queries and overwrite a whole column of data rather than just a single cell</td>
</tr>
<tr>
<td>Illegal/logically Incorrect queries</td>
<td>N/A These attacks require the attacker to receive feedback based error message generated by the incorrect query. Because the attacker cannot gain feedback in RFID systems this type of attack is useless</td>
</tr>
<tr>
<td>Union query</td>
<td>Applicable Even though union queries are mainly used to either retrieve data from additional tables or to try and bypass authentication mechanisms they can still be injected into update or insert commands to corrupt those queries or can be injected to make the middleware retrieve wrong information based on tag data.</td>
</tr>
<tr>
<td>Piggy backed queries</td>
<td>Applicable Piggy backing a query (inserting another full query along with the original query) is possible in RFID systems</td>
</tr>
<tr>
<td>Stored procedures</td>
<td>N/A If stored procedures are used attackers can use SQLIA to execute remote commands in RFID systems using them. Stored procedures are used in web applications to stop the client machines from receiving the SQL query details. In RFID systems this is not required as the queries are all generated internally.</td>
</tr>
<tr>
<td>Inference (blind injection, timing attacks)</td>
<td>N/A These attacks require the attacker to receive feedback based on the incorrect query. Because in RFID systems this does not happen these attacks are useless against them</td>
</tr>
<tr>
<td>Alternate encodings</td>
<td>Applicable In this case the attackers mount one of the above attacks but encode the injected strings in such a way as to bypass most defensive coding mechanisms in place such as data validation and sanitization. They use mechanics such as the char() function and UNHEX() function to input the string and it is then converted by the DB into malicious data during execution.</td>
</tr>
<tr>
<td>Second order injection</td>
<td>Applicable In these attacks the attacker stores data in the database that compromises the system when it’s later retrieved and used by other applications. The injected malicious data do not affect the DB and is treated and stored by the database as normal string inputs.</td>
</tr>
<tr>
<td>Commenting queries</td>
<td>Applicable Here the attacker uses a comment symbol in the input to truncate the SQL query and make only part of it execute. Only possible for queries which take two user inputs.</td>
</tr>
</tbody>
</table>
Table 6.1 highlights the major architecture differences of web based systems and RFID systems. These differences in architecture mean that some types of SQLIA’s cannot be mounted on RFID systems. Hence while there are 9 different types of SQLIA that can be mounted on web based systems [80] only 6 can be mounted on RFID systems. The reasons for this are explained in Table 6.2. This is mainly because RFID tags do not receive results or error messages and therefore attacks based on receiving feedback from the system in response to the SQLIA are ineffectual on RFID systems. Overall it is much more difficult to mount SQLIA attacks on RFID systems and therefore protecting against them become much easier for RFID systems as well.

6.3 Policy Based RFID Malware Detection and Prevention

In this section we propose a simple yet effective policy based two pronged system for the detection and prevention of RFID based SQLIA. The method we proposed is based on existing SQLIA detection techniques which have been proposed for use in web based systems. But we have modified and optimized those approaches significantly so that they are better suited for use in RFID systems. We have done these modifications and optimizations based on two key features that differentiate web systems from RFID systems.

1. RFID tag data is highly structured and of lower volume compared web based inputs and the dynamically generated therefore intercepting, validating and sanitizing that data is much easier compared to doing it in web based systems.
2. SQL queries are built by a single point in the middleware compared to web systems where they are generated on external client machines. Additionally the number of different types of dynamic queries is much less in RFID systems compared to web systems.

We describe the proposed approach as “policy based” because it requires that the developers set a number of policies concerning the valid tag inputs and legal query. In the following section we will describe the proposed approach in detail.
6.3.1 Approach Overview

The proposed system (Figure 6.2) compromises of two different techniques: RFID tag data cleaning and query pattern matching. Each technique has two main phases: static analysis phase and runtime monitoring phase.

The first technique creates data validation and sanitization policies during static analysis and enforces those policies during runtime monitoring. This ensures that only “clean” data is used in generating dynamic queries. The second technique is a SQL query pattern matching system based on simple string comparison methods. This technique requires that the programmers define policies concerning the legal query structures of the allowed queries during static analysis. The structure of the dynamically generated queries are then matched and validated, during runtime, against the legal query structures defined in the policies.

Figure 6.2: Policy based RFID malware detection and prevention
6.3.1.1 RFID Tag Data Cleaning
SQLIAs depend on inputting data in unexpected or unusual formations and structures to be successful. Therefore the root cause of SQLIA is insufficient input cleaning [80]. To ensure full RFID tag data cleaning we employ two different processes: validation and sanitization. Validation ensures that the data received from the external source adheres to pre-defined set of standards. Sanitization ensures that the data does not contain any “bad” data such as special characters or key words that have specific meaning to the system.

Data validation and sanitization has been dismissed as being unsuitable for securing web systems. But key differences in the architecture of web systems and RFID systems make it a very good option for securing RFID systems as long as care is taken to ensure that the data validation system has been properly modified to suit RFID system architecture. Additionally, because data validation and sanitization uses simple string comparison techniques their overhead is minimal ensuring high throughput and scalability. There are two distinct phases/steps to RFID tag data cleaning.

1. **Setting validation and sanitization policies** – This is carried out by a person with knowledge of both the contents of the tag and the DBMS used by the system. It is carried out during static analysis. This includes setting policies on tag data details such as length, type and formatting of the data. Policies also need to be set on illegal keywords and characters for each data field.

2. **Tag data validation and sanitization** – This is done automatically by the system during runtime monitoring by identifying inputs that do not match the validation and sanitization policies.

6.3.1.2 SQL Query Pattern Matching
While data validation and sanitization is one of the simplest and most effective countermeasures to SQLIA there are methods with which it can be bypassed [80]. By using alternate encoding mechanisms as well more complex SQLIA, attackers can bypass the data validation and sanitization. To ensure security against these types of attacks we propose a second security mechanism. This approach takes into account the structure of legal SQL queries for the system and compares it to the structure of the queries dynamically generated using RFID data. Our query pattern matching
mechanism takes advantage of the fact that SQL injection changes the structure of the query to identify potential SQLIA and prevent them from being sent to the database.

The proposed approach is a simple query pattern matching technique which employs string comparisons and is sufficient for protecting RFID systems. It is also easy to develop as most systems already include simple string comparison functionality. Evaluation shows that it provides stronger or equivalent protection to what is offered by other query pattern matching systems which use more complex comparison methods such as [86, 89, 90] when implemented in the specific architecture present in RFID systems. The simplicity of the technique is possible because the query generation is done on the middleware which has access to the parse function calls of the database and not on external web pages run on client machines. The query pattern matching approach technique consists of two steps:

1. **Defining legal query structure policies** – done during the static analysis phase, this consists of giving each different query a unique identifier and defining their query structure in a format available and understandable by the middleware. This process is explained in detail in Section 5. In our technique this is manually done by the developer who codes the dynamic query generation code.

2. **Query structure matching** – this consists of extracting the structure of dynamically generated queries by parsing (but not executing it) and seeing if it matches the legal query structure for that type of query as defined in the legal query structure policies

### 6.3.2 Static Analysis

During static analysis the first task is the creation of the validation policies which contain rules about the structure of tag data. For this, first data must be stored as separate values rather than one long contiguous block on the RFID tags. This can be done by first identifying all data fields that will be stored on the tag and by ensuring that each field has a specific use. Then a method with which to identify each field needs to be developed. This can be done by giving each field a unique identifier, whether it is a number or name. E.G:- ID, Product Name, etc. Next key data features that can be used for validation of must be identified. Normally this is features such as data-type, max length, min length etc. Finally the values for each of the data features must be identified for each field and stored in a form which is available to the middleware. This is
information such as data-type, format, max length and min length of each data field. Table 6.3 shows an example of a validation policy table for a simple system.

Table 6.3: Example validation policy table

<table>
<thead>
<tr>
<th>Field ID</th>
<th>Name</th>
<th>Type</th>
<th>Max length</th>
<th>Min length</th>
<th>Structure</th>
<th>Min value</th>
<th>Max value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Name</td>
<td>Alphabetic</td>
<td>30</td>
<td>5</td>
<td>String</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>2</td>
<td>Manufacture date</td>
<td>Date</td>
<td>10</td>
<td>10</td>
<td>--/--/----</td>
<td>01/01/2000</td>
<td>Current date</td>
</tr>
<tr>
<td>3</td>
<td>Batch number</td>
<td>Numeric</td>
<td>10</td>
<td>10</td>
<td><em><strong>-</strong></em>-____</td>
<td>000-000-0000</td>
<td>999-999-9999</td>
</tr>
<tr>
<td>4</td>
<td>Price</td>
<td>Numeric</td>
<td>8</td>
<td>4</td>
<td>Number</td>
<td>0</td>
<td>1000</td>
</tr>
<tr>
<td>5</td>
<td>Delivery Address</td>
<td>Alphanumeric</td>
<td>30</td>
<td>30</td>
<td>String</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Next the data sanitization policies must be set (Figure 6.3). As we have already identified and named/numbered all possible fields that will be stored on the tag, now we must create the sanitization rules for each of those fields.

There are two main requirements to fully sanitize data. Data must be clean of illegal specials characters (=, *, ; “ etc) and data must be clean of any illegal keywords, tokens or function names. (Keywords and tokens are defined here as strings or parts of string that have specific meaning to either the DBMS or other software that use data from the database).
Table 6.4: Example sanitization policy table

<table>
<thead>
<tr>
<th>Field ID</th>
<th>Name</th>
<th>Type</th>
<th>Characters not allowed</th>
<th>Character instances not allowed</th>
<th>Keywords not allowed</th>
<th>Keyword instances not allowed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Name</td>
<td>Alphabetic</td>
<td>YES</td>
<td>/ : * = - . ( ) ! &lt; &gt; ;</td>
<td>YES</td>
<td>IF, OR, SHUT, NULL</td>
</tr>
<tr>
<td>2</td>
<td>Manufacture date</td>
<td>Date</td>
<td>YES</td>
<td>: * = . ( ) ! &lt; &gt; ;</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>3</td>
<td>Batch number</td>
<td>Numeric</td>
<td>YES</td>
<td>/ : * = . ( ) ! &lt; &gt; ;</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>4</td>
<td>Price</td>
<td>Numeric</td>
<td>YES</td>
<td>/ : * = . ( ) ! &lt; &gt; ;</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>5</td>
<td>Delivery Address</td>
<td>Alphanumeric</td>
<td>YES</td>
<td>: * = - ( ) ! &lt; &gt; ;</td>
<td>YES</td>
<td>IF, OR, SHUT, NULL</td>
</tr>
</tbody>
</table>

To do this for each identified field first analyse if any special characters are not allowed to be contained in that field. If so decide which characters are not allowed and store them in form available to the middleware. Next for each identified field identify if any keywords are not allowed in that field. If yes decide which “bad” data (key words, special characters and reserve words etc) are not allowed and store them as the sanitization data which is available to the middleware. Table 6.4 shows an example sanitization policy table. To ensure continual strengthen of the data cleaning policies we recommend that the static analyses phase be an ongoing process with the rules being constantly updated as new functionality and programs that access the database are added to the overall system.

Finally the legal query structure policies must be created for each query that can be generated by the middleware. For this, first, all possible query types that incorporate RFID tag data and are dynamically generated by the middleware need to be identified and given a unique identifier. Then, the final syntax for each identified query must be defined and the legal query structure must be created (further explained in Section 6.3.4). RFID systems have relatively little dynamically generated queries containing tag input compared to web systems. Additionally all the queries are developed internally by the company who develops and runs the middleware and database. Therefore, we recommend that the programmer who develops the query generation software also define the legal query structure for each query manually. This has the twin advantages of minimizing the coding required and ensuring the correctness of the developed query.
models without fear of over compensation inherent in models developed by automated systems.

### 6.3.3 Runtime Monitoring

Once all required policies have been identified during the static phase the system enters the run time monitoring phase. During this phase the system reads data from the tags. When data is retrieved from the RFID tags it arrives at the middleware as a single stream of tag data (TD). Before the middleware can apply the validation and sanitization policies it must first identify and separate each individual field (td\_i where i = 1 to n) in the stream. The field identifier (i) is then used to extract the validation policies (td\_FF) for that field from storage. Then for each individual field of data (Td\_i) the data feature values such as max length, min length and data-type (td\_iv\_j) must be extracted by analysing the separated data fields. Then those extracted feature values must be matched against the values stored in the validation data (td\_ff\_j) to see if adheres to the proper data standard. If the values match the policies, then the data is passed on for sanitization else it’s rejected and the tag is identified as being malicious in the malicious tag details.

**Algorithm 1: RFID tag data cleaning algorithm**

**INPUT:** TD, td\_FF, td\_C, td\_K

**OUTPUT:** Validated and sanitized RFID tag data

**BEGIN RFID tag data cleaning**

1. Receive tag data (TD) from a reader
2. Split TD into the separate fields (td\_1, ..., td\_n)
3. **FOR EACH** (td\_i where i = 1 to n) **DO**
   4. Identify the data field using i
   5. Retrieve the feature values td\_FF = [td\_ff\_1, ..., td\_ff\_m] for td\_i
   6. **FOR EACH** (td\_ff\_j where j = 1 to m) **DO**
      7. Extract the corresponding values td\_iv\_j from td\_i
      8. **IF** (td\_iv\_j is not consistent with td\_ff\_j) **THEN**
         9. Reject data
      10. **ELSE**
          11. Mark that tag as suspicious
      **ENDIF**
   **ENDFOR**
3. Retrieve the illegal keyword data td\_K for td\_i
4. **IF** (any keywords in td\_K exist in td\_i) **THEN**
   5. Reject data
   6. **ELSE**
      7. Mark that tag as suspicious
   **ENDIF**
4. Forward to SQL query engine
4. **ENDFOR**
**END RFID tag data cleaning**
Then the system needs to sanitize that data to ensure that it does not contain known “bad” data as defined by the sanitization policies. The sanitization function checks the data for illegal keywords/characters in the inputs as defined by the sanitization policies. To do this it takes the validated data (tdi, where i = 1 to n) and retrieves the corresponding illegal keyword/character data (tdiK) for that field from the sanitization data. It then analyses tdi to see if any illegal tokens/keywords are included in that data. If any illegal tokens/keywords exist that data is rejected and the tag is marked as malicious. If it passes sanitization tdi is handed over to the query generation system. Algorithm 1 presents the algorithm for data cleaning based on preset policies.

The next step during runtime is comparing the structure of the dynamically generated queries with the structures in the legal query structure policies. To do this the query matching modules must first receive a generated query (GQ) and the associated identifier (ID) from the query generation module. When the query is received the module calls the parse function of the DBMS and inputs GQ as a parameter. The DBMS parses that query (but does not execute it) and returns the resulting parsed query (GQp) back to the query matching module. The module then uses GQp to generate the actual query structure (QSa) of GQ. Then the module uses ID to retrieve the legal query structure policies QS1 corresponding to GQ. Finally it compares QS1 with QSa. If the two does not match the query is identified as a SQLIA and rejected. Otherwise it’s forwarded to the database for execution. The algorithm for this process is presented in Algorithm 2.

GQ – Dynamically generated query
ID - Unique identifier that associates GQ with the legal query pattern
GQp – GQ after is has been parsed by the database
QSa – Actual query structure of GQ as extracted from GQp
QS1 – Legal query structure for GQ as defined by developer
Algorithm 2: Query structure matching algorithm

INPUT: GQ, ID, QS\textsubscript{i}
OUTPUT: validated QS

BEGIN Query structure matching

1. Receive GQ and corresponding ID from middleware
2. Submit GQ to a parse function of the DBMS
3. Receive GQ\textsubscript{p} as output of parse function
4. Generate QS\textsubscript{a} by removing literals from the parsed query GQ\textsubscript{p}
5. Use ID to retrieve QS\textsubscript{i} from storage
6. IF (QS\textsubscript{i} \neq QS\textsubscript{a}) THEN
7. Reject query
8. ELSE
9. Submit query to DBMS for execution
10. ENDIF

END Query structure matching

6.3.4 Query Structure Format and Matching

In developing a query structure format we used the concept of tokens to decompose the query into its different constituent parts. We then use those tokens to develop a string based query structure for any given SQL query which indicates the logical structure of the query but removes any user inputs. Our technique is based on two important features in RFID dynamic queries:-

- The same type of query generated in a dynamic manner using tag data in an RFID system will differ only in the user input values in the query.
- The input from the tag will not change the overall logic and structure of the resulting dynamic queries. In other words the user input from the tag is not meant to have any SQL statements or sub statements.

Also unlike a lot of other similar systems which converts the tag structure into a XML document for analysis [124], or employ complex parse trees and compare them [90, 91], or build finite automata for comparison purposes [89, 125] we build string structures and use a simple string comparison to match the structure of the dynamically generated query with the valid query structure policies that were defined by the developer.

6.3.4.1 Query Tokenization

In our query matching approach tokens are defined as individual string parts and can be one of five main types: Keywords, Symbols/Operators, Identifiers, Literals and Comments.
• Keywords: these are words that have specific meaning to the DMBS (SELECT, FROM, INSERT, WHERE and predefined functions such as AVG(), SUM(), CONCAT())
• Symbols/operators: these are either single or compound symbols that have a specific meaning to the DBMS (+, =, ‘;’, etc)
• Identifiers: these are words that identify specific database components (table names, column names and user defined variables)
• Literal: These are bits of code that indicate the literal value of an item (e.g.:- scott, 23.56, 12/07/1982). In our system variables are also considered literals as the variable itself will be replaced by a literal value when the query is dynamically generated by the middleware.
• Comments: extra code that do not have any meaning to the database and is therefore ignored.

The first three types of tokens are important for the logic and structure of the query the third is only user input which has no effect on either the logic or the structure of the query while the last is used by the programmers to makes notes for future use and have no effect what so ever on the actual query. Legitimate RFID tags are not meant to contain any of the first three types of tokens or comments and only contain actual values, or in other words the literals contained in the query. Keeping this in mind we begin developing SQL query structures as follow: The first step in creating the SQL query structures is to break the query down into its component string compromising of words and symbols and identify the type of each token each substring is.

6.3.4.2 Query Structure Policy Generation

Imagine a system which takes as input the tag ID and the product it’s attached to as input from the tag and saves that data to a database table. The resulting query for this process is as follows:

```
INSERT INTO product (tag_id, product_name) VALUES ('tagid', 'productname');
```

In the above query ‘tagid’ and ‘productname’ are string variables which are read in from the RFID tag. The tokenized version of the above query, with each bit string separated and identified, would be:
Now if we strip the literals (the red background) and replace them with “?” as a marker and remove comments from the query we get the common query structure for all dynamic queries generated for inputting product details into the product table based on tag input which is as follows:

![Tokenized query with literals removed](insert_into_product_tag_id_product_name_values_tagid_productname)

By replacing the literals with “?” we ensure that the structure does not take into account the changing values for each different query, allowing the tag input to change as required. By keeping the first three types of tokens we ensure that the structure contains all the data concerning the query logic and structure, allowing for the logic of the dynamic queries to be validated. We discard the comments as they play no part in the query and are not used by the database. Once the query structure is identified, by stripping all the literals and comments, it is then converted into all lowercase (as SQL is case insensitive) and saved as a string along with the unique identifier for that particular query type. In the same manner the query structure for all dynamically generated queries must be identified and analysed. An example for a legal query structure policy table can be found in Table 6.5.

<table>
<thead>
<tr>
<th>Query Identifier</th>
<th>Query Structure STRING policy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>insert into product (tag_id, product_name) values(?,?);</td>
</tr>
<tr>
<td>2</td>
<td>select * from product where tag_id = ?;</td>
</tr>
</tbody>
</table>

### 6.3.4.3 Runtime Query Matching

During runtime the query structure matching module receives, from the dynamic query building module, the generated query and the query identifier which indicates which query pattern policy it should match. At this point the module first parses the received
query and uses the resulting parse information to strip it of any literal values and replace them with “?” . It then uses the received ID to retrieve the legal query structure string for the received query and compares it with the query structure string it generated. If the two match the query is validated and sent to the database for execution if not it’s identified as containing a SQLIA and discarded.

Our technique uses a much simpler method of comparison compared to the methods used by other techniques such as parse trees, XML documents or finite. Most systems and programming languages have built in string comparison and manipulation controls which make implementing this kind of comparison easier than the more complex custom types. Additionally string comparison is also a much quicker and less resource intensive comparison method compared to the other more complex methods used.

6.4 Security Evaluation

The goal of the evaluation presented in this section is to test the effectiveness of the approach presented in this chapter. Because the strength of the security afforded by the data cleaning is directly tied to the completeness and strength of the data cleaning policies we will discuss the additional security afforded by it. Additionally we will also present the results of the testing of the query structure matching mechanism.

6.4.1 RFID Tag Data Cleaning

Input validation is one of the simplest and most effective ways of preventing the simpler types SQLIA [80]. Unfortunately, because the security it affords depend on the strength of the data cleaning rules most data checking/cleaning techniques fail, not due to a flaw in its concept but due to weaknesses or incompleteness in the rules developed for them or due to incompatibilities between them the architecture of the system they are implemented in [80]. We have already analysed the architecture of RFID systems and determined that it’s suited for input validation techniques as a security measure against tag based SQLIA in Section 6.2.3. To minimize the possibility of weak or incomplete data cleaning policies we have set up a strict policy generation methodology (explained in Section 6.3.2) and used two different types of data cleaning approaches based on two different core concepts: (1) Validation: Which is based on the concept of white listing and (2) Sanitization: Which is based on the concept of black listing. The use of this combination enhances the security afforded by the technique by ensuring that more
variables and factors are taken into account by the people who develop the data cleaning rules. Additionally the validation and sanitization policy creation process we have developed and presented in Section 4.2 has been specially designed to ensure the strength and completeness of the policies created. This is done by ensuring that the policies take each separate data field into account. The policy creation process also ensures that multiple features and attributes of each data field are used in creating the validation and sanitization policies leading to much stronger and thorough policies.

In addition to its effectiveness against simpler SQLIA and its simplicity the other reason we use validation and sanitization in our technique is protection against second order SQLIA. Second order injection attacks do not change the structure of the dynamically generated query [78]. Therefore SQLIA detection techniques that rely on query structure/pattern matching, such as [84, 86, 88, 89], are ineffective against these types of attacks and second order injection still remains a very prominent threat to the security of most web and RFID applications. But, because RFID tag data cleaning does not depend on the query structure, but the rather the format and content of the input data, it can still be used to spot possible instances of second order injection in RFID tags. Therefore in our proposal we have included the data validation and sanitization technique in addition to the query matching technique to ensure that there is some protection against second order SQLIA. But to ensure adequate protection against second order injection it is imperative that the analyses carried out in the static phase are complete and the rules created are comprehensive. It’s also important that the developers take into account the other systems that will be accessing the RFID database and build the tag data cleaning rules with their weaknesses in mind.

6.4.2 Query Structure Matching

We also carried out testing to evaluate the security of the query structure matching technique. To carry out a thorough evaluation all three main types of dynamically generated SQL queries possible in RFID systems (SELECT, UPDATE, INSERT) had to be tested. Therefore we developed a number of queries of each type ranging from simple to complex and developed the legal query structure for each query.
6.4.2.1 Methodology

To evaluate our technique we used two programs. One is the freely available demo version of the General SQL parser (GSP Demo) downloadable at http://www.sqlparser.com/download.php. The other was a simple string comparison program written by us. For parsing of the dynamic queries we used the pretty print facility of the GSP Demo. We set the pretty print format options as shown in Figure 6.6. All tokens except comments, strings and numbers were set to show with green font colour. Comments were blue while strings and numbers were red. Shows a dynamic query containing a tautology after it has been passed.

Figure 6.6: Example of a parsed query

Once the query was parsed we replaced all red text (literals) with “?” and deleted all blue text (comments). The resulting string was then compared with the legal query structure (for the more complex queries and attacks) using the comparison program we had written (Figure 6.7). The program takes the dynamic query and strips any newline characters and any multiple spaces replacing them with single spaces. It then runs a
single string comparison to compare result with the legal query structure input at the bottom of the program.

![String Comparison](image)

**Figure 6.7: Example of a query comparison**

### 6.4.2.2 Results

Not all types of SQLIA can be mounted on RFID systems. Therefore when testing our system we only tested the types of attacks possible on RFID systems and ignored SQL attacks such as timing attacks, inference and illegal/illogical queries. The Table 6.6 shows the breakdown of our testing process and the results obtained when testing malicious queries. In total around 300 queries were manually tested.

**Table 6.6: Evaluation results**

<table>
<thead>
<tr>
<th></th>
<th>Select Queries tested(Detected)</th>
<th>Update Queries tested(Detected)</th>
<th>Insert Queries tested(Detected)</th>
<th>Total (Detected)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tautologies</td>
<td>21(21)</td>
<td>21(21)</td>
<td>N/A</td>
<td>42(42)</td>
</tr>
<tr>
<td>Union query</td>
<td>18(18)</td>
<td>6(6)</td>
<td>12(12)</td>
<td>36(36)</td>
</tr>
<tr>
<td>Piggy backed queries</td>
<td>15(15)</td>
<td>15(15)</td>
<td>15(15)</td>
<td>45(45)</td>
</tr>
<tr>
<td>Alternate encodings</td>
<td>12(12)</td>
<td>12(12)</td>
<td>12(12)</td>
<td>36(36)</td>
</tr>
<tr>
<td>Commenting queries</td>
<td>2(2)</td>
<td>5(5)</td>
<td>1(1)</td>
<td>8(8)</td>
</tr>
<tr>
<td>Total</td>
<td>68(68)</td>
<td>59(59)</td>
<td>40(40)</td>
<td>167(167)</td>
</tr>
</tbody>
</table>
For all types of queries and all types of SQLIA types tested our query structure matching technique was able to identify SQLIA with 100% efficacy. In addition during the testing process we also tested around 120-130 legal queries. All legal queries were allowed by the technique with a 0% false positive rate. Even though the testing was limited to around 300 queries in total and carried out at logical level rather than implementation level the 100% detection rate and 0% rate in false positives are very promising.

Table 6.7: Security comparison table

<table>
<thead>
<tr>
<th></th>
<th>Detection rate</th>
<th>False Positive rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed Approach</td>
<td>100%</td>
<td>0%</td>
</tr>
<tr>
<td>AMNESIA [89]</td>
<td>100%</td>
<td>0%</td>
</tr>
<tr>
<td>SQLCheck [91]</td>
<td>100%</td>
<td>0%</td>
</tr>
<tr>
<td>SQLGuard [90]</td>
<td>Results not presented</td>
<td>Results not presented</td>
</tr>
<tr>
<td>SQLrand [86]</td>
<td>Results not presented</td>
<td>Results not presented</td>
</tr>
<tr>
<td>Tautology-checker [87]</td>
<td>&lt; 100%</td>
<td>Not available</td>
</tr>
<tr>
<td>CANDID [126]</td>
<td>100%</td>
<td>0%</td>
</tr>
<tr>
<td>SQLDOM [127]</td>
<td>Results not presented</td>
<td>Results not presented</td>
</tr>
</tbody>
</table>

In Table 6.7 we compare the detection results of our approach against the security by some other SQLIA detection techniques. Please note the results do not take into account second order injection attacks. As the results show our approach is on par if not better than the best of the other approaches that are available in literature.

Table 6.8: Comparison of SQLIA detection techniques

<table>
<thead>
<tr>
<th></th>
<th>Tautology</th>
<th>Union</th>
<th>Piggy Backed</th>
<th>Alternate encoding</th>
<th>Commenting</th>
<th>Second order injection</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed Approach</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Uses simple string comparison for both approaches</td>
</tr>
<tr>
<td>AMNESIA [89]</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Uses NDFA which may over or under estimate the structures</td>
</tr>
<tr>
<td>SQLCheck [91]</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Uses parse trees for comparison and secret keys which increase system overhead</td>
</tr>
<tr>
<td>SQLGuard [90]</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Uses parse trees for comparison and secret keys which increase system overhead</td>
</tr>
<tr>
<td>SQLrand [86]</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Use high over cryptographic techniques</td>
</tr>
<tr>
<td>Tautology-checker [87]</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Only works for tautologies</td>
</tr>
<tr>
<td>CANDID [126]</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Uses a dynamic method to guess the programmer intended query structure</td>
</tr>
<tr>
<td>SQLDOM [127]</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Needs a custom set of classes be built for each database schema</td>
</tr>
</tbody>
</table>

High overhead
Additionally, as Table 6.8 shows our technique is the only technique that actually has a possibility of detecting and preventing second order injections (given strong enough data cleaning rules). It is also one of the simplest approaches available as its uses only simple string comparison for both techniques compared to the more complex techniques used by other systems.

### 6.5 Summary

In this chapter, we presented a simple but secure approach for detecting and preventing tag based RFID SQLIA. The overall technique consists of two different methods. The first method is a simple validation and sanitization technique for the RFID tag data which is based on data formatting and sanitization policies created after careful analysis of the data stored on the tag. This technique prevents ‘bad’ data from being used while building dynamic queries and is effective against second order injection attacks. The second method is a SQL query structure matching technique which uses simple string comparisons to identify possible SQLIA. This technique has the advantage of protecting against all other SQLIA types possible on RFID systems while being simpler than other proposals that use more complex matching techniques such as parse tree validation [90] or query randomization [86].

The testing of the query structure matching method yielded very positive results. We tested all possible types of dynamic queries that may be generated in RFID systems with all possible types of attacks that can be mounted on those systems. In all more than 300 queries were tested with around 170 attacks and around 130 legal queries. The testing showed a detection rate of 100% and false positive rate of 0%. Our approach (specifically the validation and sanitization technique) was specifically designed to protect against second order injection attacks on RFID systems. This type of SQLIA cannot be detected by any query matching system. The main weakness of the proposed techniques is that the security against second order injection relies heavily on the data cleaning policies. To ensure the strength and completeness of the data cleaning policies we have a developed and presented a data cleaning policy creation methodology that is highly structured. This methodology ensures that policies created by following it takes a large number of different attributes into account and is as complete as possible, ensuring that a high level of protection is afforded by the approach.
Chapter 7

Conclusion and Future Research

Even though RFID research and adoption has seen a significant increase in the last decade there are still some barriers that pose a hurdle to its total acceptance. These barriers come in the form of security concerns and performance issues. The main purpose of this thesis is to enable the wider adoption of RFID technology in large networked environments and systems. To achieve this aim we have first developed and presented a networked RFID architecture framework optimised for this type of large scale systems. We have also carried out research on different means with which the security of these types of systems can be improved. This research includes a comprehensive networked RFID security framework, an RFID security protocol and a tag based malware detection and prevention technique for RFID systems. Throughout this thesis we have presented the main outcomes of our research and also the analyses we carried out that show the improvements and advantage our solutions pose over currently existing solutions.

7.1 Summary of Contributions

The first task that was carried out during the research carried out for this thesis was the completion of a comprehensive literature review on a number of different areas of RFID research. For this the following four main areas were reviewed: (1) Current RFID architecture frameworks in context of the requirements and features of very large multi entity RFID systems, (2) Network and RFID security frameworks, (3) RFID security
protocols and (4) SQLIA detection and prevention techniques. During the review we made the following conclusions concerning networked RFID systems and their performance and security:

- The EPCGlobal Architecture Framework has a number of significant issues when used to develop very large scale multi entity RFID networks. These issues include inefficient data storage and look-up, needlessly complex and lengthy partner lookup and unnecessary duplicate data processing. Some of the other proposed architectures have some advantages over the EPCGAF but they also contain some weaknesses. Therefore a new architecture framework must be developed which can perform more efficiently in large scale networked RFID systems.

- The currently existing security frameworks are not suitable for applying to and securing large scale networked RFID systems. The general network security frameworks do not take into account the many differences in RFID systems and general IT networks and therefore lack a lot of detail and security features required by the RFID networks. The frameworks on RFID security are not complete as they only look at some aspects of the overall security of the system. Therefore the need for a targeted and comprehensive security framework for large scale networked RFID systems is very acute.

- While there are a large number of RFID security protocols in current literature none of them have been successful in securing the tag-reader communications of RFID systems employing low cost tags. Some proposed protocols have proven to be secure but they are too resource intensive to be implemented on low cost RFID tags. The protocols that are simple enough to be implemented on low cost RFID tags have been proven to be unsecure and vulnerable to a wide range of different attacks. Therefore an RFID security protocol that is both simple enough to be implemented on low cost tags and provably secure still need to be proposed.

- RFID malware is a threat which has received very little attention but pose a very real and major threat to RFID systems. Next to no research has been carried out and detecting or preventing these types of attacks on RFID systems. The currently existing SQLIA detection techniques can be modified and used to
detect RFID malware but they need to be significantly modified and altered to ensure they can perform at the required efficiently and in the specific environment of

Once the literature review was completed we then started developing RFID architecture which would perform better in the specific environment found in very large scale RFID systems such as global supply chain management systems. In chapter 3 we present the outcome of this research. The architecture developed in this chapter is of a modular and extensible nature allowing for developers to pick and chose the kind of functionality the they want/need from there system. The proposed architecture uses p2P techniques for both data look-up and data sharing leading to significant advantages. It is also a networked RFID architecture optimised and targeted for large scale RFID systems with multiple independent partners. The comparative analysis carried out and presented at the end of the chapter show that our architecture has a number of advantages over the EPCGAF including less data lookup, reduced duplication of data retrieval and processing and more reliability due to reduced reliance on external data sources.

Another major concern for RFID adopters are all the security and privacy issues associated with RFID systems. Because a methodical method with which an RFID developer can identify all the threats, attacks and security requirements of any given RFID system was needed we developed a networked RFID security framework. The outcome of this research is presented in chapter 4. The proposed framework consists of a threat model, attack model and security model. The framework looks at a large number of different security aspects of RFID systems including identification of threats and attacks possible on RFID systems, identification of the core components and security concepts that must be maintained and the identification of all the security functionality required to protect them. We validated the proposed framework by applying it to a real world RFID system. The validation showed the effectiveness of the framework as it identified a few security issues as well as novel methods with which those issues could be mitigated.

In chapter 5 we develop and present an RFID security protocol that uses a hybrid of traditional and ultra-light-weight cryptographic techniques. By using traditional security techniques such as reader side PRNG and one-way hashing we ensure that the protocol has high security. The use of ultra-light-weight techniques such as bitwise XOR
operators ensures that the resource requirements are low. Therefore the protocol we propose, unlike a number of recently proposed RFID protocols, manages to provide both high security and low resource requirements. The proposed protocol was then evaluated for both its security and its performance. The security evaluation shows that the protocol is immune to a large number of common RFID attacks while still providing a number of core security requirements such as mutual authentication, confidentiality, integrity and privacy. The performance comparison proves that the resource requirements of the proposed protocol are less than or comparable to other recently developed ultra-light-weight RFID security protocols.

Finally we looked at the issue of RFID malware detection and prevention. RFID malware is a relatively new type of security threat and was only identified as being possible in 2006. Therefore there has been very little work done on actually securing an RFID system from this kind of attack. In chapter 6 we first analyse RFID malware and conclude that all currently existing RFID malware are based on SQLIA. Because SQLIA is an old and established threat in web systems we then analyse web systems and normal RFID systems to see the differences in SQLIA in web systems vs. SQLIA in RFID systems. Based on the differences identified we conclude that some of the prevention methods deemed unsuitable for securing web systems from SQLIA can be successfully implemented to secure RFID systems from SQLIA. Next we propose and develop RFID malware detection and prevention system that uses two different techniques to detect tag based RFID malware and prevents it from infecting the backend databases. One of these techniques is tag data validation and sanitization system. To ensure that the strength of the black list and white list rules used in this system we present a highly structured method with which these rules can be developed. The second technique is a dynamic query matching system that employs a new method of string comparison to match the structure of the dynamically generated queries with the structure of the legal queries. The security evaluation of the query matching system shows very high detection rates and no false positives.
7.2 Future Research Opportunities

In the course of the research carried out for this thesis we identified a number of open research areas or current solutions that need to be improved upon. For the benefit of future researchers in this area we will now list these research opportunities.

7.2.1 RFID Identifier Management

One of the biggest challenges in massively networked RFID systems is the issue of managing and distributing the identifier used to uniquely identify the tags and the issue of identifying data sources that have data about any given tag in an efficient manner [128]. Currently this task is carried out by the ONS service provided by the EPCGlobal [13]. One of the biggest issues with this system is the lengthy lookup process associated with identifying any given tag and then locating and retrieving data about that tag from partner EPCIS [2]. If this process could be eliminated or at least streamlined and simplified the performance of massively networked RFID systems would be improved. Therefore networked RFID systems are in dire need of new efficient methods with which the identifiers of RFID tags and the associated data storage locations can be distributed to the partners of the systems. This problem can only become more pertinent with time as the number of RFID tagged objects, RFID systems partners and RFID data locations will only grow over time adding even more complexity and load on the currently used ONS services [128].

7.2.2 ONS Security and Performance

One of the main requirements of the data lookup service provided by the EPCGlobal was scalability and a relatively high degree of availability. The DNS services used to identify and locate web addresses on the World Wide Web has both of these qualities. Therefore the EPCGlobal data lookup service (the ONS) was developed based on the DNS service [13]. While this ensured the scalability and availability of the ONS service it also meant that it inherited a number of the weaknesses inherent in the DNS service. These weaknesses include vulnerabilities to DDoS attacks, vulnerabilities to cache poisoning attacks, the need for duplication and real time update of the servers to ensure scalability and availability, and the slow update rate of DNS servers [31]. Because of reliability and scalability requirements of RFID systems are even greater than those of the World Wide Web the weaknesses of the DNS are only magnified when they are
translated to the RFID environment. Therefore research on how to improve the security
and performance of the ONS service can have great benefits on RFID systems as well as
improving the DNS services used by the World Wide Web.

7.2.3 Creation of Low Resource Cryptographic Primitives
A large number of security solutions are based around the use of common concepts such
as PRNG, one-way hashes, cryptographic hashes, encryption algorithms and CRC
algorithms. Therefore over the past years a large number of such cryptographic
primitives have been developed by researchers. Now, strong and secure security
solutions can easily be created by mixing and matching already existing, proven secure
cryptographic primitives [129]. Unfortunately most of the currently existing
cryptographic primitives were developed focused on typical IT systems and therefore
require much more resources than are available on low cost RFID tags. While some
research has been done on creating low cost, efficient, cryptographic primitives for use
in resource constrained environments [77, 130], there still remains a number of
significant gaps in this area. Additionally the progress in creation of low cost
cryptographic primitives would pave the path for more efficient security solutions for
typical IT systems as well as increase the security available in other low cost IT systems
such as sensor networks.

7.2.3 Protection against Active Jamming
A major vulnerability of RFID systems is there weaknesses to active jamming. RFID
active jamming is when attackers create a denial of service between tags and readers.
The two most common ways of active jamming RFID systems is: (1) Emitting a very
strong radio wave that drowns out the tag transmissions and (2) flooding a reader or tag
with so many requests that it get overloaded and cannot respond to legitimate requests.
The first type of attack is easily detected is much harder to recover from. The second
type of request is hard to detect if it’s aimed at tags. Currently next to no work has been
done on securing these kinds of attacks on RFID systems. There are a number of
approaches that are used to detect and recover from active jamming attacks in other
types of systems, especially in web systems where they are commonly mounted in the
form of DDoS attacks. Therefore research into the viability of modifying and using
those techniques for RFID systems security can be fruitful as completely new ways in which active jamming attacks can be detected and stopped.

7.2.4 RFID Tag Data Security
With the price on memory dropping more and more RFID systems will start to store tag data on the tag itself for easier and faster access. Unfortunately unlike data storage components in typical IT systems the tags of RFID systems are highly vulnerable to physical access by attackers. Therefore it is imperative that security solutions that keep the data stored on the tag secure in case of physical access by an attacker to the tags. This means that if an attacker does steal a tag of a system they must not be able to understand the data stored on the tag. In addition with physical access they can now bypass the security protocols embedded in the tag and directly modify the tag contents. Therefore there must be means by which the readers can verify that the tag data has not been compromised directly. Hence there will be a need for security solutions that focus on keeping the data stored on the tag secure in case of physical compromise of the tag and also look at how such compromises can be detected.

7.2.5 RFID Tag Access and Update Control
Very large networked RFID systems are used by a number of independent partners. With the dropping price of tags and increases in the storage capacity of the tags these partners may want to store private data on the tag itself for easier access [2]. In this kind of environment there must exist means to ensure that other partners that have access to the tags do not gain access to the private data of other partners. Similarly the system must also ensure that the private data of one partner cannot be modified by other partners. This requires access and update controls that are based on a set of very fine access rules which are set by an overall system administrator [17]. While access controls exists for other types of IT components such as Databases and programs currently no research has been done in this area for RFID tag data. This remains a very open and compelling area of research.

7.2.6 Non-repudiation
Non-repudiation is the assurance that when a person or entity carries out an action they are entitled to that person or entity cannot, in the future, deny that action. In IT this is a requirement for most shared or communication systems to ensure that users can place
trust in the systems data and information. When networked RFID systems start storing data about the objects it’s attached to on the RFID tags themselves the tags will change from simple data storage containers to a communication mechanism between the different partners. For example a partner can change the final destination data field on the tag to let other partners know where that specific object should be sent to. But once this happens the issue of repudiation will become a security concern for RFID tag data [17]. If a partner can change the data stored on the tag and then later deny making those changes then user trust in the system will decline. Therefore a means by which non-repudiation of tag data updates can be assured is of utmost importance and no research has yet been carried out in this area.
REFERENCES


