Solutions for Internet of Things Security Challenges: Trust & Authentication

Jason M. McGinthy

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Computer Engineering

June 18, 2019
Blacksburg, Virginia

Keywords: Security, Internet of Things, Standardization, Authentication, Lightweight

Solutions for Internet of Things Security Challenges: Trust & Authentication

Jason M. McGinthy

(ABSTRACT)

The continuing growth of Internet-connected devices presents exciting opportunities for future technology. These Internet of Things (IoT) products are being manufactured and interleaved with many everyday activities, which is creating a larger security concern. Sensors will collect previously unimaginable amounts of private and public data and transmit all of it through an easily observable wireless medium in order for other devices to perform data analytics. As more and more devices are produced, many are lacking a strong security foundation in order to be the "first to market." Moreover, current security techniques are based on protocols that were designed for more-capable devices such as desktop computers and cellular phones that have ample power, computational ability, and memory storage. Due to IoT's technological infancy, there are many security challenges without proper solutions. As IoT continues to grow, special considerations and protections must be in place to properly secure this data and protect the privacy of its users. This dissertation highlights some of the major challenges related to IoT and prioritizes their impacts to help identify where gaps are that must be filled. Focusing on these high priority concerns, solutions are presented that are tailored to IoT's constraints. A security feature-based framework is developed to help characterize classes of devices to help manage the heterogeneous nature of IoT devices and networks. A novel physical device authentication method is presented to show the feasibility in IoT devices and networks. Additional low-power techniques are designed and evaluated to help identify different security features available to IoT devices as presented in the aforementioned framework.

Solutions for Internet of Things Security Challenges: Trust & Authentication

Jason M. McGinthy

(GENERAL AUDIENCE ABSTRACT)

The Internet has been gaining a foothold in our everyday lives. Smart homes, smart cars, and smart cities are becoming less science fiction and more everyday realities. In order to increase the public's general quality of life, this new Internet of Things (IoT) technological revolution is adding billions of devices around us. These devices aim to collect unforeseen amounts of data to help better understand environments and improve numerous aspects of life. However, IoT technology is still in its infancy, so there are still many challenges still remaining. One major issue in IoT is the questionable security for many devices. Recent cyber attacks have highlighted the shortcomings of many IoT devices. Many of these device manufacturers simply wanted to be the first in a niche market, ignoring the importance of security. Proper security implementation in IoT has only been done by a minority of designers and manufacturers. Therefore, this document proposes a secure design for all IoT devices to be based. Numerous security techniques are presented and shown to properly protect the data that will pass through many of these devices. The overall goal for this proposed work aims to have an overall security solution that overcomes the current shortfalls of IoT devices, lessening the concern for IoT's future use in our everyday lives.

Dedication

To my family.

Acknowledgments

I would not have been able to achieve this goal without the support and encouragement of many people in my life. First, I would like to thank my wonderful wife, Janice. Without your support, I would have never been able to have the peace of mind to complete this journey. Your proof-reading and suggestions helped make this a document instead of technical jargon. You handled a busy household, completed your own degree, and never failed at making a good point when I needed advice. Next, I would like to thank my advisor, Dr. Alan Michaels, for continually encouraging and challenging me to push through roadblocks and put words on paper. Your guidance was monumental in helping me accomplish this dream. Finally, I would also like to thank the Department of Computer and Cyber Sciences at the United States Air Force Academy for giving me this opportunity to follow my dream. I look forward to returning and imparting the wisdom I have gained through this process into future Air Force officers.

Contents

List of Figures

List of Tables

List of Abbreviations

ABP	Activation by Personalization
AES	Advanced Encryption Standard
ALM	Adaptive Logic Module
AP	Access Point
ARPANET	Advanced Research Projects Agency Network
ASIC	Application-Specific Integrated Circuit
CC	Central Controller
CDMA	Code-Division Multiple Access
CERT	Computer Emergency Response Team
CI	Critical Infrastructure
CIA	Confidentiality, Integrity, and Availability
CMOS	Complementary metal–oxide–semiconductor
CNN	Convolutional Neural Network
CPSS	Cyber-Physical Systems Security
CRC	Cyclic Redundancy Check
CRP	Challenge/Response Pair
CSI	Channel State Information
CSMA/CA	Carrier-Sense Multiple Access with Collision Avoidance
CSSS	Chaotic sequence spread spectrum
DBSCAN	Density-based Spatial Clustering of Applications with Noise
DC	Direct Current
DDoS	Distributed Denial of Service
DES	Data Encryption Standard

DoS	Denial of Service
DPA	Differential Power Analysis
DRAM	Dynamic-Random Access Memory
DSSS	Direct Sequence Spread Spectrum
DTLS	Datagram Transport Layer Security
ECC	Elliptic Curve Cryptography
EN	Edge Node
FDMA	Frequency-Division Multiple Access
FPGA	Field-Programmable Gate Array
FRAM	Ferroelectric Random Access Memory
GDPR	General Data Protection Regulation
GEF	Galois Extension Field
GF	Galois field
GPS	Global Positioning System
GPU	Graphics Processing Unit
HIPAA	Health Insurance Portability and Accountability Act
HKDF	HMAC-based Key Derivation Function
HMAC	Hash-based Message Authentication Code
HOPS	High-Order Phase Shift Keying Signaling
ICS	Industrial Control System
IEEE	Institute of Electrical and Electronics Engineers
IGMM	Infinite Gaussian Mixture Model
IIoT	Industrial Internet of Things
IKM	Input Keying Material
InfoSec	Information Security

IoT	Internet of Things
IPsec	Internet Protocol Security
JSP	Joint Security Plan
KDF	Key Derivation Function
LE	Logic Element
LFSR	Linear-Feedback Shift Register
LIL	Law of the Iterated Logarithm
LoRaWAN	Long Range Wide Area Network
LPD	Low Probability of Detection
LPI	Low Probability of Interception
LUT	Look Up Table
MAC	Media Access Control
MANET	Mobile Ad Hoc Network
MCU	Microcontroller Unit
MFA	Multi-Factor Authentication
MITM	Man in the Middle
ML	Machine Learning
MPU	Memory Protection Unit
NIST	National Institute of Standards and Technology
NLFSR	Nonlinear-Feedback Shift Register
NN	Neural Network
NVRAM	Non-volatile Random Access Memory
OTA	Over-the-Air
OTAA	Over-the-Air Activation
pdf	Probability density function

PHY	Physical
PIN	Personal Identification Number
PK	Parent Key
PKDF	Pseudorandom Number Generator-based Key Derivation Function
PKI	Public Key Infrastructure
PRNG	Pseudorandom Number Generator
PSK	Phase Shift Keying
PUF	Physically Unclonable Function
PUF	Physically unclonable function
RAM	Random Access Memory
RF	Radio Frequency
RFID	Radio Frequency Identification
RNG	Random Number Generator
ROM	Read Only Memory
RSA	Rivest Shamir Adleman cryptographic algorithm
RTC	Real-Time Clock
RW	Random Walk
SAE	Society of Automotive Engineers
SCADA	Supervisory Control and Data Acquisition
SEI	Specific Emitter Identification
SK	Session Key
SNR	Signal to Noise Ratio
SoC	System on a Chip
SRAM	Static Random Access Memory
Standard Hash Algorithm	SHA

SWaP	Size, Weight, and Power
TCP/IP	Transmission Control Protocol and the Internet Protocol
TDMA	Time-Division Multiple Access
TEE	Trusted Execution Environment
TLS	Transport Layer Security
TPM	Trusted Platform Module
TPMS	Tire Pressure Monitoring System
TRANSEC	Transmission Security
TRNG	True Random Number Generator
TSN	Time-Sensitive Network
V2I	Vehicle-to-Infrastructure
V2V	Vehicle-to-Vehicle
V2X	Vehicle-to-Everything
WAIC	Wireless Avionics Intra-communication
WMR	Window Modulus Reduction
WPA	WiFi Protected Access
WPAN	Wireless Personal Area Network
WSN	Wireless Sensor Network

Chapter 1

Introduction

The Internet of Things (IoT) is becoming more commonplace in our everyday lives and experts anticipate the number of these devices to will easily grow past 50 billion by the year 2020 [80]. Recent advances in the electronics and communications fields are allowing numerous miniaturized devices to be deployed in remote locations to continually monitor and collect data for an ever-growing number of applications. The growth of these IoT devices will allow massive amounts of data to be collected and analyze to study consumer, industrial, personal, health, transportation, and environmental factors to help increase efficiency and reduce costs. Sectors that previously did not rely on the Internet are now racing to be early adopters of this promising technology and may only tack on security as an afterthought. There is also a major concern in some of these sectors that the developers do not have sufficient security backgrounds, yet they are designing and manufacturing devices that could expose the devices to a litany of attacks that these manufacturers previously never had to defend against. Many of these devices will not have the capabilities to perform current security protocols, exposing them to possible attacks until sufficient solutions are found. As IoT expands, the threat of attacks on these devices and networks is continually looming. Current Internet security implementations on computers and personal devices are still overwhelmed by the number of daily attacks because a single compromised node can cause damage to an entire network. The scale of IoT will grow exponentially, dramatically increasing the number of devices, which translates to more possible vulnerabilities. At these growing rates, securing our industrial infrastructure will become a bigger challenge as these devices will have access to and provide personal and/or private information about people, vehicles, and critical structures and systems. Overall, improvements in security design have failed to keep pace with the technological advances of IoT; this dissertation seeks to help close the growing gap by providing IoT security solutions for resource-constrained devices.

1.1 The Internet

The Internet that we rely on today evolved from a United States' Advanced Research Projects Agency Network (ARPANET) originally created in 1969 [132]. This system continued to grow and become a commercial commodity that began to link university researchers in the 1980s. Through the following decades, the technology became more mainstream and is now the central nervous system of the world. The Internet provides previously unachievable access and communication from small networks to the World Wide Web. These unbridled connections allow great advances in research and technology, but this great connectivity also provides tools to bad actors that want to cause more harm than good. With the proliferation of the Internet, attacks began to surface. The first well-known attack was launched in 1988 by an MIT student, Robert Morris, and was subsequently known as the Morris worm [75]. The attack infected thousands of computers across networks and provided a glimpse to future attacks. This attack prompted the creation of the Computer Emergency Response Team (CERT) at Carnegie Mellon University and the birth of cybersecurity [98].

Cybersecurity focuses on the protection of computers, data, and the exchange of information across networks. As evident from the Morris Worm, cybersecurity can be a cat-and-mouse game, because it is very difficult to anticipate all new attacks. Therefore, systems must be continually updated to defend against evolving attacks. Since cybersecurity, or what has more recently become more appropriately phrased as cyber-physical systems security (CPSS), encompasses everything from the device to the systems and networks, there are a large number of attack vectors that can be used to compromise a device or system. Many current solutions employ physical security such as secure buildings, strong encryption protocols such as the Advanced Encryption Standard (AES) [181], and other robust security protocol suites such as Transport Layer Security (TLS) [163]. While these solutions are standard for many industries today, the evolution of IoT will render these approaches impractical on smaller IoT devices due to their limited resources. The world is becoming IoT-enabled by commoditization of low-cost, yet very capable, hardware, creating a vulnerable reality until IoT security solutions are universally in place.

1.2 Evolution of IoT

Originally coined the 'Internet of Things' by Kevin Ashton in 1999 [35], IoT has grown from radio frequency identification (RFID) systems to the current IoT research boom that is creating more implementations of this 20-year old idea. These devices are approaching credit card sizes, yet they are being used as embedded sensors, actuators, etc. to perform remote, sometimes autonomous actions. Although many of these devices are more capable than the computers that helped land men on the moon, they are extremely resource-constrained compared to current desktop computers. As previously described, much of the current cybersecurity solutions are designed for these desktop-type computers because that is still the majority of the Internet's composition. However, these computers will no longer be the main targets. IoT has already begun to grow at an alarming rate, introducing an immense number of poorly secured attack surfaces. Without effective IoT security designed specifically for these smaller devices, risks will abound throughout many areas of our lives from healthcare, automobiles, and critical infrastructures.

1.2.1 The S in IoT is for Security

The Internet has always faced security challenges due to its global reach, but many cybersecurity solutions have been developed with the abundant processing power, memory, and power availability. Many traditional security protocols are not practical for IoT systems, because they do not scale down efficiently to the resource levels of many embedded IoT platforms. Some commercial IoT products are adding Internet capabilities without the thought of practical security, introducing more attack surfaces in previously offline applications. Cost, developers' abilities, and lack of standards all contribute to the lack of security in IoT in these applications, adding to the overall challenges of safely and securely deploying devices. Although device manufacturers are trying to improve quality of life, there are many products being fielded with insufficient device and network security, in an effort to be "first to market" and provide IoT to new areas [144]. This lack of security as a main design tenet has led to recent IoT-based attacks. For example, in October 2016, a botnet attack named Mirai caused a wide-scale distributed denial of service (DDoS) attack that caused many outages of popular websites such as Twitter, Spotify, Netflix, Amazon, Tumblr, Reddit, and PayPal across parts of the eastern United States [50]. This botnet attack was successful due to non-existent or poorly implemented security schemes, such as default username and password usage [158]. Another security concern appeared in 2017 with IoT-capable security cameras using a popular, open source, third-party software library that contained a vulnerability allowing complete control of the camera [11]. The continued increase in the number of small, wireless IoT devices, combined with the lack of attention to security during IoT device design, invites common network attacks such as eavesdropping, man-in-the-middle (MITM), denial of service (DoS), node impersonation, and battery draining [70], [212], [114].

1.3 IoT Scenarios

The following sections highlight some of the biggest beneficiaries and design requirements of this expanding technology. Many of these areas have distinct reasons for utilizing IoT, but they also share many common traits in relation to security. Since IoT is very broad in its use, researchers and manufacturers must deliver solutions that allow for differing levels of protection based on the industry-specific application. As will be further described below, some of these areas may focus more on business efficiency, whereas others will impact personal safety and convenience. These differences in scope add to the growing complexity of IoT security, yet collectively, they help establish a broad landscape for deriving requirements and designing IoT solutions that satisfy the common characteristics of the different use cases.

1.3.1 Industry 4.0

One major area that is already seeing benefits of IoT is industrial IoT (IIoT) such as manufacturing and production businesses. The inclusion of IoT is being called “Industry 4.0” to indicate the fourth industrial revolution [93]. IIoT is leveraging the vast amount of data collection through sensors and other devices to help improve processes, reduce costs, and decrease accidents. For example, Amazon uses autonomous robots to control inventory and retrieve customers’ orders in order to streamline its shipping process [24]. Future IIoT will also allow geographically separated sites to communicate production line details more efficiently, reducing inventory costs and providing more insights of issues or delays that can be off-loaded to a different location on the fly. This is feasible due to the vast amount of inter-connected devices constantly communicating through the entire manufacturing and production life cycle.

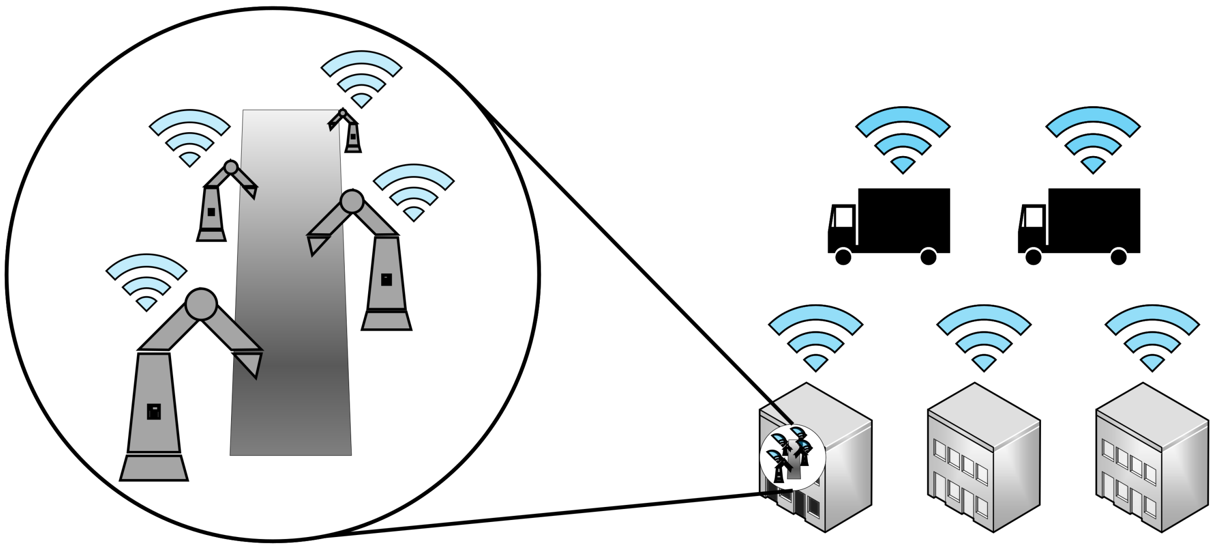
Figure 1.1: Illustration of Industry 4.0. All aspects of smart manufacturing are connected, from the production line, to shipping and receiving, and to the other production sites to maximize efficiency from all areas.

Although IoT is projected to bring great value to many industries, the actual scale is still difficult to truly comprehend. Experts predict that global spending on Industry 4.0 will grow to \$310B by 2023, up from an expected \$119B in 2020 [64]. Industry 4.0 is going to continue to grow in popularity, and the initial cost improvements may outweigh proper security designs. The increasing scalability of IoT devices could potentially overwhelm unprepared system administrators, and combined with the collection of massive amounts of data, could introduce vulnerabilities that could be found and compromised. Therefore, security must be prevalent throughout the transition to these IoT devices and systems.

Authorization and authentication are critical concerns as more systems rely less on human involvement. If an attacker is able to gain unauthorized access to a manufacturing system, monetary and physical damages can occur. For example, suppose a a system is compromised, and then the attacker can gain access to confidential manufacturing designs and intellectual property, as has allegedly been the case with China recently [63]. This will allow the theft of trade secrets. Beyond authorization and authentication, information security is the next layer of defense. If an attacker gains access, but is unable to read or modify data in a realistic time frame, then there is less incentive for an attack to continue. Companies attempt to protect their data from competitors, such as finances, intellectual property, and trade secrets, to maintain market advantages.

Another important concern for IIoT are timing and latency requirements. If latency and timing are negatively affected, the overall availability of a system is degraded. For instance, many of the production lines use robotics and are fine-tuned to precise timing windows. If a denial of service attack knocks a system offline, a factory can incur significant costs for downtime or re-tooling a new line. Moreover, if this factory is connected to a larger production system spanning numerous manufacturing centers, then this single attack could cascade and cause serious impacts to a business.

The rapid increase of IoT devices in Industry 4.0 is coupled with a concern for size, weight, and power (SWaP), and wireless communications. The SWaP challenge will be a compromise between cost and function for many of the businesses, whereas the large number of devices may begin to overwhelm current time-division multiple access (TDMA) and frequency-division multiple access (FDMA) wireless protocols [150].

1.3.2 Critical Infrastructures

As businesses attempt to capitalize on the cost savings of IoT and autonomy, public sector infrastructures will also benefit from this autonomy trend via Critical Infrastructures (CIs). In 2006, the European Union provided guidance for identifying and protecting CIs throughout its member states [79], and per the 2013 Presidential Policy Directive/PPD-21, 16 such CIs were named, highlighting their importance for protection [157]:

One overarching goal in these CI sectors is to integrate predictive analytics and autonomously act on data without human interaction. Also, beyond autonomous data collection, many of these CIs are controlled semi-autonomously through remote systems such as industrial control systems (ICSs) and Supervisory Control and Data Acquisition (SCADA) systems. These systems are in place to help monitor and control many different areas, such as electrical power grids, oil and gas utilities, and water and waste management systems. At the time of these publications, threats were known, but the advancement of the Internet and ever-increasing capabilities of attackers have enabled a greater number of attack vectors. The added nature of IoT only compounds this issue due to the exponential growth of devices, each increasing the attack surface of CIs.

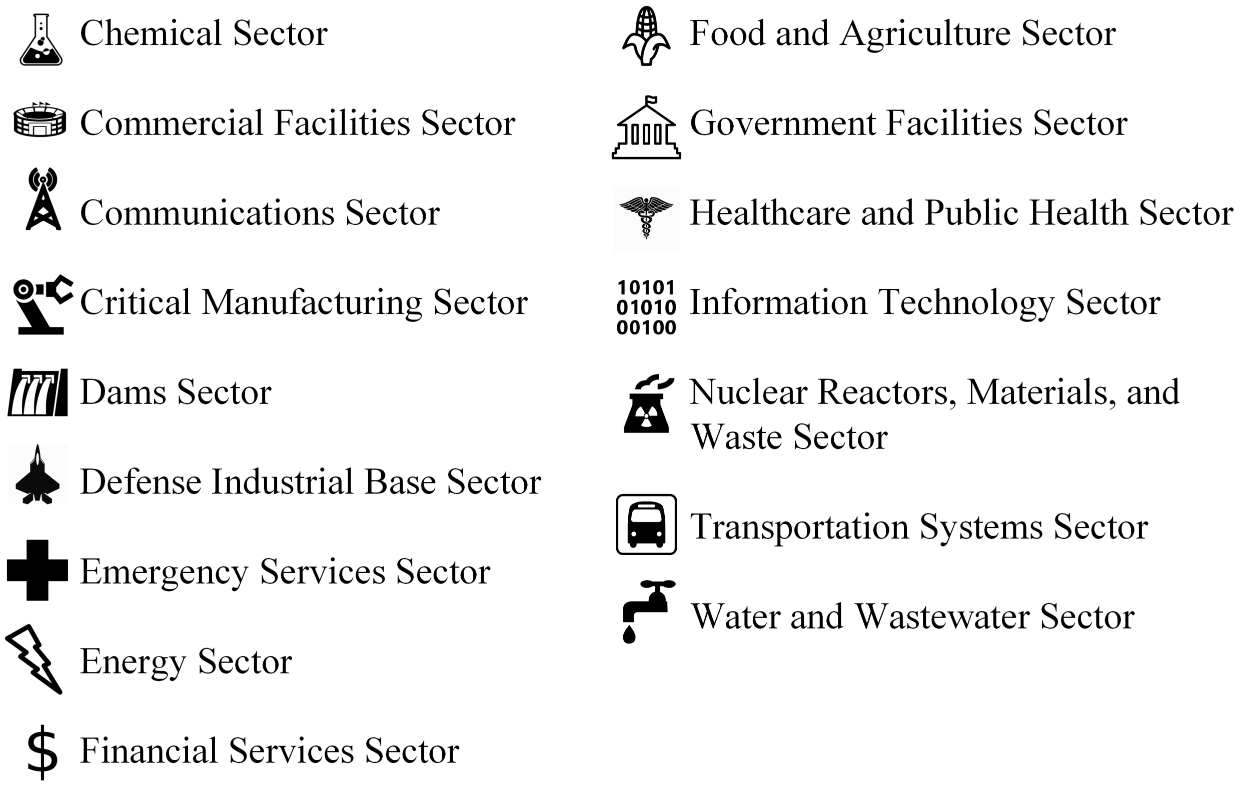
Chemical Sector
Commercial Facilities Sector
Communications Sector
Critical Manufacturing Sector
Dams Sector
Defense Industrial Base Sector
Emergency Services Sector
Energy Sector
Financial Services Sector
Food and Agriculture Sector
Government Facilities Sector
Healthcare and Public Health Sector
10101
01010
00100
Information Technology Sector
Nuclear Reactors, Materials, and Waste Sector
Transportation Systems Sector
Water and Wastewater Sector

Figure 1.2: Sixteen critical infrastructures named in 2013 Presidential Policy Directive/PPD-21.

Many of the same challenges present in Industry 4.0 are found in CI, but the impacts carry more than just monetary losses. For example, an attack on the power grid in Chicago during winter could lead to thousands of deaths. Therefore, the requirements that must be met for CI applications are higher than similar industrial areas. Due to their critical nature, CIs rely heavily on trust and authentication of users and devices. As many of these systems can have impacts on a national scale, ensuring proper trust and authentication protocols is imperative to their security. While IIoT is mainly bound by warehouses and production lines, some CI applications may span thousands of square miles through remote areas, yet still require monitoring to ensure proper safety and security protocols are met. It is very difficult to manually monitor these utilities in remote parts of the country. For example, a utility pipeline carrying gas across the empty expanse of western Texas to El Paso would rely on a secure and robust SCADA system that are remotely accessible across 500 miles of pipeline. This ideal setting for IoT provides better granularity of system performance, such as pressure and flow metrics of these pipelines. If an attacker gained access to the network through device tampering or spoofing, it would be difficult to locate, yet severe damage could be done to the utility system, causing major impacts to customers or the environment downstream from the attack.

1.3.3 Automotive

The automotive industry is also embracing the IoT revolution. Integrating IoT devices into vehicles and roadway infrastructure will allow real-time communication between vehicles and smart infrastructures to greatly improve people's quality of life in the transportation domain. Vehicle-to-vehicle (V2V), vehicle-to-infrastructure (V2I), and vehicle-to-everything (V2X) communications aim to reduce traffic delays, improve fuel efficiency, and improve safety without human input. For example, V2X can assist drivers with intersection movement, weather alerts, road hazards, emergency brake light notification, and forward collision warning. These last two scenarios are depicted in Figure 1.3 as the first car on the right suddenly stops and the truck detects the forward collision, while the driver behind the truck is unable to see the sudden stop ahead, but the car is alerted and deploys proper safety measures to avoid an accident. As smart cities begin to become reality, V2X communication will also aim to improve navigation. The smart infrastructure surrounding the incident is able to propagate the information to traffic management systems in case diversions are needed.

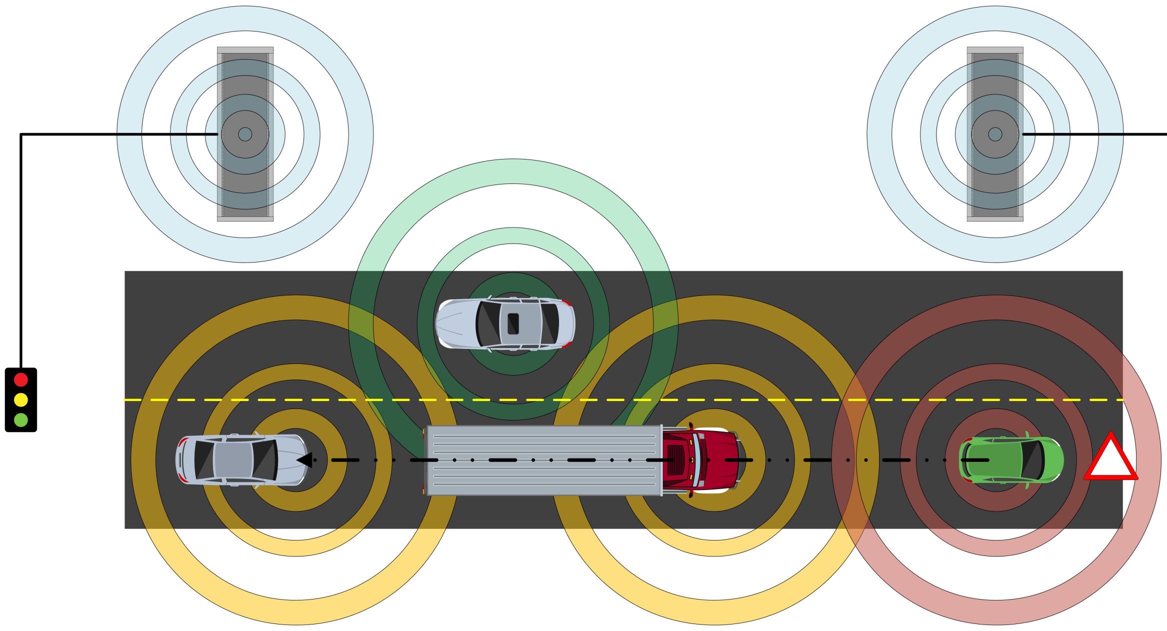

Figure 1.3: IoT can help alert vehicles prior to visual confirmation of dangers. This information can then be passed amongst vehicles and smart city infrastructure to possibly change traffic patterns to alleviate congestion.

Although the safety implications of IoT in vehicles are undeniable, a major issue that still lingers is privacy. Consumers do not want their information such as current location, vehicle heading, vehicle identifier, or speed transmitted in the open. Moreover, recording or transmitting driving behaviors that could be selectively penalized by insurance or police will meet resistance. Information security (InfoSec) is critical to protecting the user data and also providing availability of the data to ensure these safety systems do not fail due to data loss. Moreover, data that is transmitted, not only between vehicles, but also just intended for a single vehicle, should be properly protected from outside observers such as the tire pressure monitoring system (TPMS) [169, 182] and remote key-less entry fobs [83]. Current generations of these devices have well known attacks that permit access to gain vehicle identifiers or subvert locks and security systems. In a similar vein, authentication remains a major concern. If attackers are able to impersonate a sensor or controller, they may be able to track or gain physical access to the vehicle, provide misinformation that could cause serious injury or death, or impact traffic throughout an area. Therefore, more solutions to protect privacy and ensure authentication must be researched.

1.3.4 Wireless Avionics Intra-communication

As the automotive world is embracing IoT, aircraft manufacturers are also considering replacing redundant wired systems, but there are more safety concerns due to failures having greater consequences. Wireless avionics intra-communication (WAIC) is an emerging area for IoT solutions due to the costly weight of the miles of copper wires that run throughout an aircraft [14]. By introducing a wireless solution to the current wired avionics network, there are many beneficial gains including reduced aircraft weight, reduced production and maintenance costs, and future flexibility in terms of new sensors and wireless devices [29]. These benefits show great promise for the future of aviation. Weight savings from the removal of copper wire can improve overall flight efficiency. Easier maintenance may reduce the amount of time an aircraft is grounded as well as assembly times. Wireless technology allows for future expansion of sensors and devices easily integrated into an aircraft. However, all of these anticipated benefits of incorporating portions of a wireless infrastructure into the aircraft architecture require an increased focus on the security aspects of the data handling and transport since opening up a wireless interface offers previously unavailable attack vectors. WAIC networks also have important security considerations. Due to the use of a wireless medium for communications, wireless network security must be fully examined. Current wired technologies have less risk of cyber-attacks due to the assumed lack of physical access to the system. However, wireless communications extend the range of the interface(s) to elements of the system. This introduces new challenges that must be addressed to ensure system security.

Historically the aviation industry has been slower to adapt to emerging technologies due to its safety-critical nature. Also, there are many different components in an aircraft produced by many different manufacturers that want to create and distribute IoT data for the aircraft. Due to this slower transition and numerous bodies at play, standards for creating, transmitting, and securing IoT data in aircraft are also lacking. Typical governing bodies such as the Federal Aviation Administration, RTCA, and Society of Automotive Engineers (SAE) International are well aware of cyber threats [109], but there are few standards in place for emerging IoT technologies in the aviation industry. Without robust standards across the

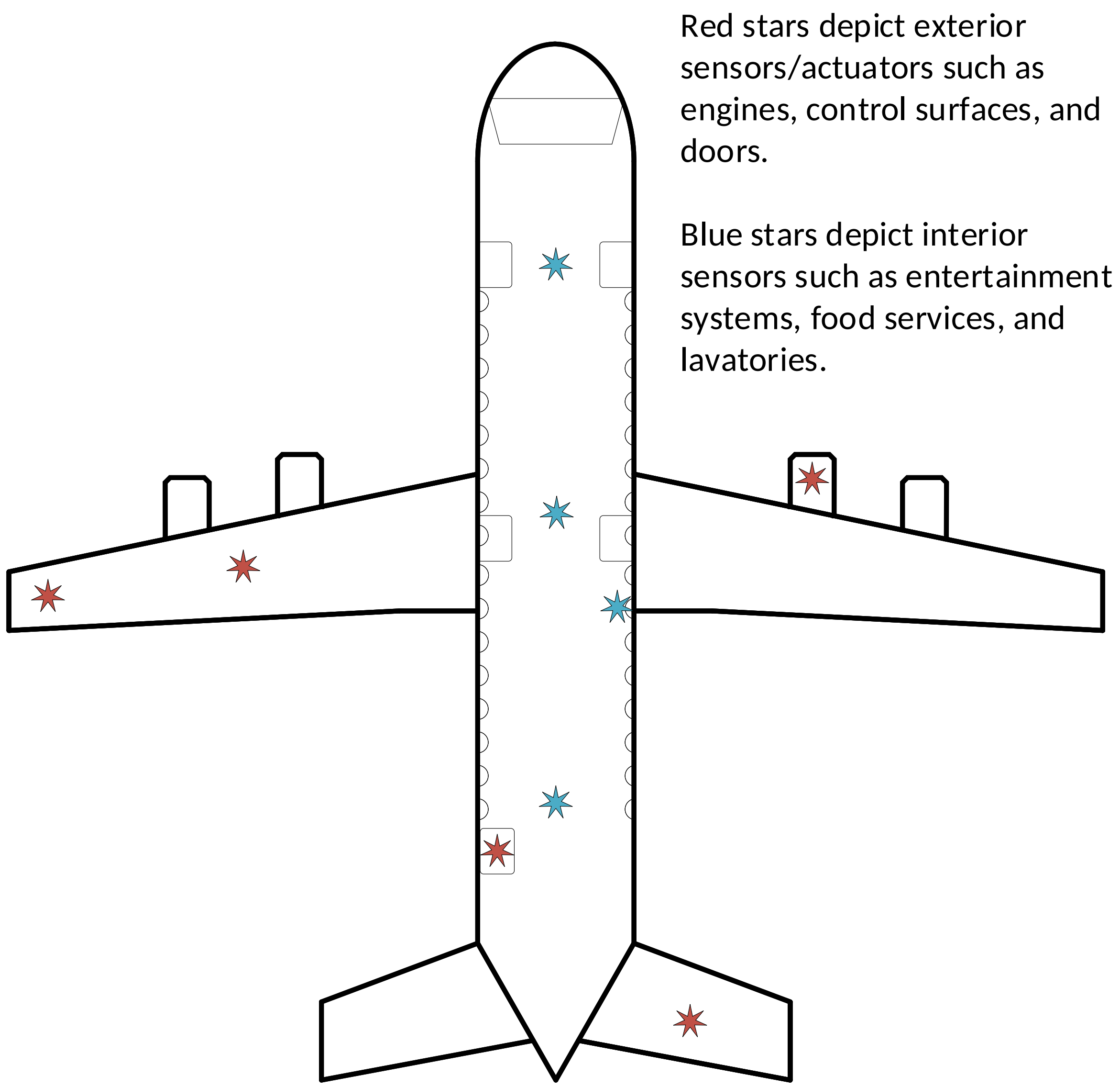
Red stars depict exterior sensors/actuators such as engines, control surfaces, and doors.
Blue stars depict interior sensors such as entertainment systems, food services, and lavatories.

Figure 1.4: Illustration of IoT devices in WAIC.

entire industry, IoT implementation will continue to lag behind other sectors, and aviation companies will continue to forego the potential benefits.

Confidentiality is important because we must ensure that only those personnel that are authorized can access sensitive data. However, due to the open nature of wireless communications, messages will be able to be seen by anyone. Therefore, any data that should not be seen by unapproved sources should be properly encrypted. Integrity of the data on a WAIC system is critical because the pilots must trust that the data provided has not been modified or otherwise tampered. Availability is another crucial aspect of security for WAIC systems.

Since many of the sensors in the system are providing real-time health and status updates of safety systems, any impact to availability could have negative results. A final important trust objective for a WAIC network is non-repudiation. Data sent from a sensor shall not be able to denied that it was sent from that particular node. This will ensure that if there is a problem discovered by a sensor, that the approximate location of that issue is near that sensor, since the sensor is known to have sent it.

However, in a sensor-based system like WAIC, authentication is the most critical security objective for the system. If devices cannot be authenticated, their communications cannot be trusted. If a hacker is able to gain access through a compromised authenticated device or an authenticated rogue sensor, then the entire system could be compromised. Another major concern for WAIC systems is latency due to the safety-critical nature of many of the systems in the aircraft. If the underlying control loops are compromised due to IoT devices, then cascading failures could cause serious problems. One last concern for WAIC is the SWaP requirements of the replacement devices. Since WAIC is trying to reduce overall weight, then the new devices must allow for significant weight savings.

1.3.5 Healthcare

Another important area being considered is the healthcare arena. IoT looks to reduce the overall cost of healthcare by allowing users more rapid digital information flow between a user and medical professionals. Internet-connected wearables allow physicians and caregivers to monitor and observe patients without the need of a trip to the hospital or clinic. This in turn can help reduce the strain on the healthcare system from lack of beds to over-worked doctors and nurses [39]. Unfortunately, this also may cause infringements on individuals' privacy. Considering the fact that these wearables contain very sensitive and private details about a person, there are significant privacy concerns and challenges with the integration of IoT into healthcare. New devices are also being used to help medical conditions, such as diabetes [56] and Parkinson's [147] that are critical for a person's health, but allowing wireless access adds security risks. For instance, if a device transmits an identifier that can be observed as a unique individual, then this could lead to the possibility of tracking that person, similarly to the concern in the automotive industry with the use of TPMS.

With IoT-based healthcare solutions, information security must be properly designed and implemented. Verified users and devices must be authenticated and authorized before having access to any data that has been collected. Data integrity should also be a crucial goal as any tampering could cause risk of injury or worse. Also, if these medical devices are unable to communicate to the larger systems due to availability issues, then the added benefits of these devices is impaired. Beyond good information security practices, the device security life-cycle should be well understood and planned. Once a patient is done with a device, data must be successfully removed from the sensor to stop attackers from gaining information from discarded devices. With the influx of these IoT medical devices, there is still a lack of

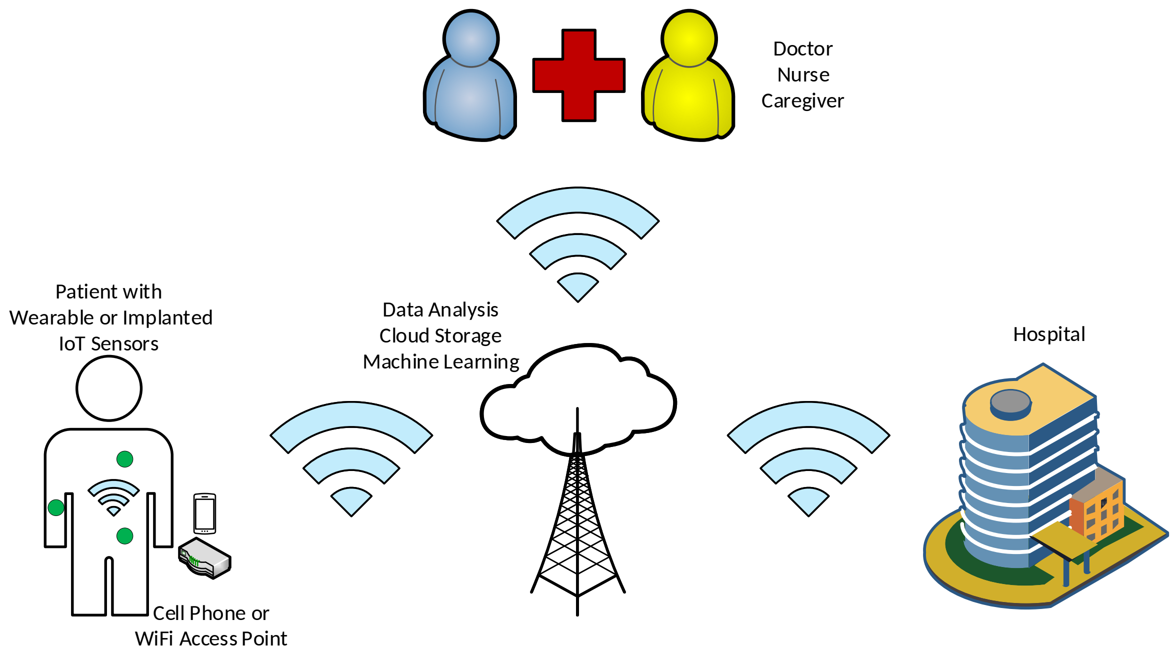

Figure 1.5: Illustration of IoT devices in healthcare connecting patients, medical professionals, and caregivers.

standardization amongst manufacturers that can lead to possible conflicts between devices. This is a major concern since this industry directly impacts people's health and well-being. Recently, there has been an attempt to forge a new framework for medical devices' life-cycles. The Healthcare and Public Health Sector Coordinating Council Cyber Security Working Group recently released its Medical Device and Health IT Joint Security Plan (JSP) [25] that attempts to lay a framework for cybersecurity through a device's entire life cycle. This document illustrates movement in the right direction towards an agreeable framework for these medical devices, but it is not enforceable, which limits resources if vendors choose not to follow.

1.3.6 Smart Consumer & Homes

The final scenario presented is the direct connection to the everyday consumer. IoT is leveraging the massive amount of data collected to tailor business towards people. Early adopters have Internet capable devices throughout their homes. Smart security systems, appliances, and voice-activated servants, such as Alexa, are an easy sell to tech-savvy consumers who want the latest technology in their lives and homes. This craze does not appear to begin slowing down in the near future, so many companies are racing to launch Internet-connected technology for all consumer sectors. Unfortunately, these same developers that are adding

Internet capabilities to refrigerators are not necessarily security proficient, inducing vulnerabilities in our everyday smart devices. Even for security conscious companies, the addition of always-listening, Internet-connected devices causes major concerns. Just recently, Amazon has admitted that their Alexa home systems are still passively collecting and sending recordings to Amazon employees in the name of performance improvements [54].

Figure 1.6: Illustration of IoT devices in smart homes

Inviting vulnerable devices into our homes increases an attacker's ability for surveillance as our habits can be studied and planned around. Internet-capable door locks and security systems could be compromised, allowing attackers to easily enter into an empty home. Many people do not fear attacks because they do not believe they are likely victims, but poorly secured IoT devices can be marketed as the latest technology and consumers are still likely to buy them.

Another major issue with consumer devices is the reliance on the owner to perform suggested or even required updates of these devices. Humans are already the weakest link when information security is involved [68], but increasing the number of IoT devices in a home can become a daunting endeavor of upkeep. Once a device becomes too burdensome to maintain, it will be quickly forgotten, leaving security risks unbeknownst to the patron. Moreover, consumer loyalty is a big factor with companies, so if product reviews call out the inconvenience of security, consumers may avoid products, reducing profits for businesses.

1.4 Challenges in IoT Security

The previous scenarios highlighted numerous use cases possessing security concerns and benefits with the addition of Internet connectivity to previously offline systems. Through comprehensive analysis of these scenarios and related literature, ten significant challenges were chosen and prioritized based on their impact to IoT security:

1. Standardization
2. Trust & Authentication
3. Privacy
4. Information Security
5. Network Attacks
6. Latency
7. Wireless Communications
8. Version Control & Updates
9. Physical Attacks
10. Size, Weight, and Power

Figure 1.7 highlights these challenges in the different scenarios and the corresponding severity in each area. As clearly shown, some challenges are of great importance to all areas, where some challenges impact different applications at varying degrees. Nonetheless, these challenges provide ample research areas that will improve the secure deployment of future IIoT. One final note: although smart home and consumer products are continuing to flood the markets, this dissertation is consciously not prioritizing impacts of compromise. The main focus is on IIoT scenarios and applications due to the large-scale and more severe impacts in these areas. The following sections will provide more details on the selected IoT challenges.

1.4.1 Standardization

Throughout the broad sectors of IoT, device and system manufacturers are currently trying to capitalize on new technologies and be the first to stake a claim in their respective markets. This rush to be the "first-to-market" many times leaves security as a lower concern. Multiple vendors may create similar devices, but they will follow completely different design recommendations; deployment of these heterogeneous devices into a common environment often requires de-activating some security features to enable coexistence/collaborative operation. Many of these IoT devices tack security on as an afterthought [144], causing a greater need for industry standardization. Some industries are encouraging their device manufacturers to heed security warnings and follow security design guidance, but there is still a lack of security standardization across many IoT areas. Although there are a lack of formalized standards, businesses are beginning to understand the impact of security and the need for standardization. A 2017 IoT security survey showed that 96% of the businesses believe that there should be regulation for IoT security [22]. Unfortunately, there is still a great need for this

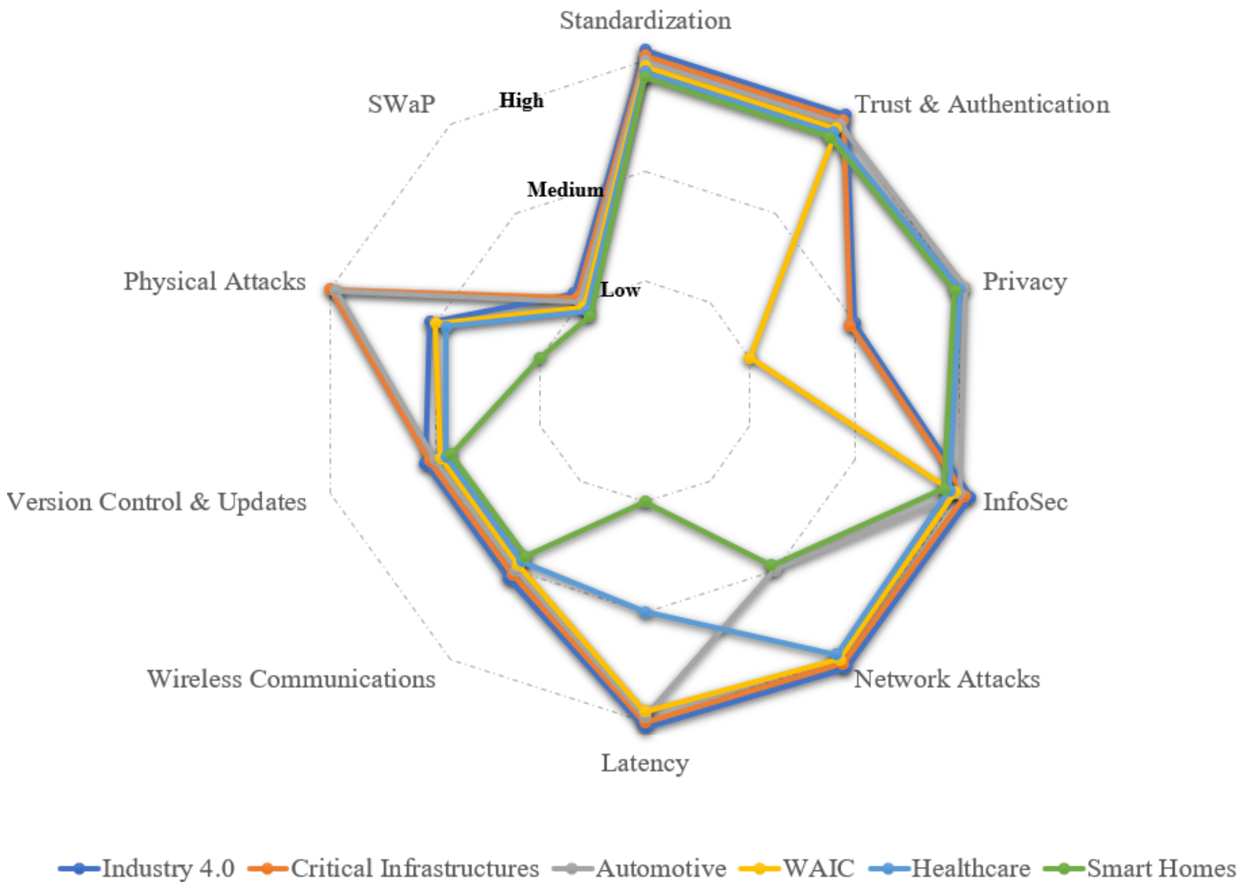

Figure 1.7: Based on the previous scenarios, selected significant IoT challenges were rated based on their overall impact in each presented scenario. Although this is a qualitative representation, it is based on comprehensive research of current technologies and literature.

self-regulation, and more devices are entering the market without proper security in place. As the number of IoT devices continue to grow, even a small number of vulnerable devices can have a major impact of the security of all Internet-connected things. A great concern within IoT is device trustworthiness, and at this current time, the lack of standards does not create a great sense of trust amongst these devices. Some sectors of IoT are currently trying to provide frameworks for their unique cases, such as the previously mentioned healthcare-focused JSP [25], yet as IoT grows, all of these unique areas will be inter-connected and a broader framework will need to be agreed upon at the lowest levels of the security stack. In fact, the National Institute of Standards and Technology (NIST) is trying to determine IoT standards for future operability across many domains [152] in order to protect the future of IoT devices and networks. As IoT continues to mature, NIST and other organizations such as the Institute of Electrical and Electronics Engineers (IEEE) should set standards for all manufacturers and security administrators to follow for a unified IoT. Until a set of defined standards are agreed upon, there is still a fear that companies that do not follow good security practices may put others at risk when their devices are integrated into heterogeneous networks.

1.4.2 Trust & Authentication

IoT brings other security concerns even when current security schemes are properly implemented. For example, the scalability of IoT increases the possibility for remote device or data observation, allowing an attacker to collect vast amounts of information to help in reverse engineering and malicious emulation of security protocols [218]. Therefore, authentication is a critical area of network security to ensure that only trusted devices are able to communicate on a network. Most authentication protocols used perform non-physical data inspection (i.e., data content, not physical signal), similar to Data Encryption Standard (DES) [46], WiFi Protected Access (WPA) [194], WPA2 [195], or elliptic curve cryptography (ECC) [108], harboring latent defects. Other security practices currently used to ensure the integrity of a message perform authentication only on the data, using techniques such as encryption, message authentication, and cyclic redundancy checks (CRCs) at the media access control (MAC) layer, and rely on the use of a unique device identifier, randomly generated values, and/or shared cryptographic keys [78]. While these techniques are very capable of message structure authentication, they fail to authenticate the identity of the actual device transmitting the data, which can lead to possible device compromise or impersonation, as most of the involved parameters can be observed, guessed, or replicated [43].

The current best practice for establishing trust is multi-factor authentication (MFA). This can generally be achieved by performing at least two of the following methods: *what you possess* (*e.g.*, a physical token or ID card), *what you know* (*e.g.*, a password, PIN, or cryptographic key), or *who you are* (*e.g.*, unique physical characteristics) [168]. As a result, methods have been developed and implemented to perform device authentication, such as physically unclonable features (PUFs). However, they achieve mixed results because their deterministic responses can be replicated [57]. Therefore, a method by which to authenticate devices through non-deterministic and truly unclonable means is desired.

1.4.3 Privacy

Privacy pertains to the protection of sensitive data, especially personally identifiable information such as name, account numbers, location, and health. Therefore, the goal of privacy is not to allow access or even knowledge that private data is being stored or transmitted. Privacy is a major concern in the consumer product focused industries, automotive, and especially healthcare applications. In order to provide data privacy, authentication and authorization, data encryption, and transmitted signal security all play important roles. Access to the sensitive information must be strictly controlled and the data must always be encrypted in case unauthorized access to the data is obtained. When the data is transmitted, it should be encrypted and extra layers of transmission security should be in place to obfuscate the signal. Privacy goes beyond just security measures: it is also highly governed by policies such as the recent General Data Protection Regulation (GDPR) [197] and Health Insurance Portability and Accountability Act (HIPAA) [17]. When governing bodies are involved, the importance of standards also becomes a major factor. This combination of standards, authentication, InfoSec, and transmission security (TRANSEC) illustrate the importance of providing proper security measures in IoT to ensure privacy.

1.4.4 Information Security

InfoSec is the focus on properly securing computers, data, and the exchange of information across networks. Many times, this is accomplished through the use of strong cryptographic protocols. These protocols attempt to ensure the confidentiality, integrity, and availability, commonly referred to as the CIA triad, of data and systems. Confidentiality ensures that only authorized users are able to access data. This is a main tenet for applications concerned with privacy. If safeguards are in place to protect the confidentiality of data, then privacy is easier to maintain. Next, data integrity is focused on ensuring that information is not modified by non-authorized users. IoT will generate massive amounts of data, and, in order to properly analyze and make decisions based on it, security experts must properly protect the information gathered and transmitted across the networks. The final part of the triad is availability, which relies on uninterrupted access to the information. If information is unavailable, then it is unusable. This could cause serious impacts in safety realms of IoT such as critical infrastructures, automobiles, aircraft, and healthcare. By ensuring proper InfoSec, data risks can be lessened. The following subsections will go into more detail on the main components of InfoSec.

Cryptographic Functions

InfoSec heavily relies on strong cryptographic protocols to protect sensitive data, and these protocols are used primarily two types of data encryption: asymmetric and symmetric. Public key infrastructure (PKI) is centered around asymmetric cryptographic methods, such as the Rivest Shamir Adleman (RSA) algorithm [181]. Asymmetric cryptography is performed by two devices each having their own pair of keys: a private and public key. These public keys are managed by a certificate authority in order to provide assurance that the public key (certificate) is genuine. This process is depicted in Figure 1.8.

RSA is a very computationally expensive process, so other asymmetric methods perform a key transfer such as Diffie-Hellman to produce a shared secret key for future symmetric cryptographic functions. Through a mathematical process, a common shared secret key is produced for each device, allowing the use of the more efficient symmetric encryption. Many current security protocols utilize these type of public key distribution, such as TLS

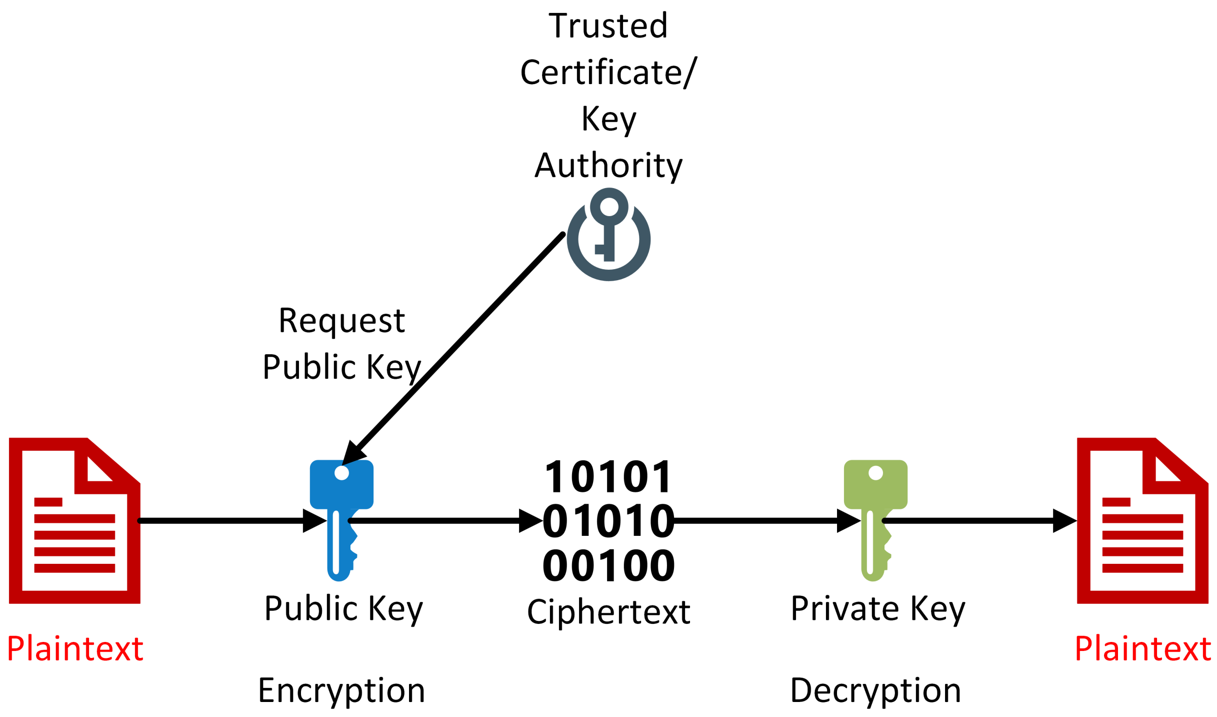

Figure 1.8: Illustration of asymmetric encryption. In order to encrypt data that only the receiver can decrypt, the transmitter must get the receiver's public key from a trusted third party and encrypt the data with that public key. In this process, only the receiver's private key will be able to decrypt the information.

and Internet Protocol Security (IPsec) [181]. Generally, in current IoT designs, two devices will perform a key exchange procedure, such as Diffie-Hellman, in order to create a secure channel to exchange the secret symmetric key. This process requires multiple handshakes between the two parties, making it less than ideal to need to be accomplished frequently on resource-constrained devices.

This asymmetric approach is very scalable as there is no shared cryptographic material initially amongst devices. This also means that in the event of a node-capture, an attacker has no knowledge of another device's secret keys. However, the approach is currently not efficient in resource-constrained IIoT devices due to the expensive computations and large key lengths used in the key exchange process for most algorithms. Moreover, a certificate authority is difficult to build and connect to all nodes in a network, adding to the overall complexity of asymmetric encryption and PKI in IoT.

On-going research in elliptic curve cryptography aims to make public key cryptographic techniques more efficient due to the vast reduction in key lengths (e.g., 2048-bit RSA vs. 224-bit ECC) for comparable security. Although ECC shows promise in terms of efficiency, there are concerns about the overall security due to recent revelations that possible backdoors were designed in certain curve implementations recommended by NIST [108].

Beyond asymmetric approaches used mainly for key exchanges, symmetric encryption is primarily used for actual data encryption due to its higher effective security levels with the same key lengths. Symmetric encryption relies on the use of a shared secret key that each party uses to encrypt and decrypt data, as illustrated in Figure 1.9.

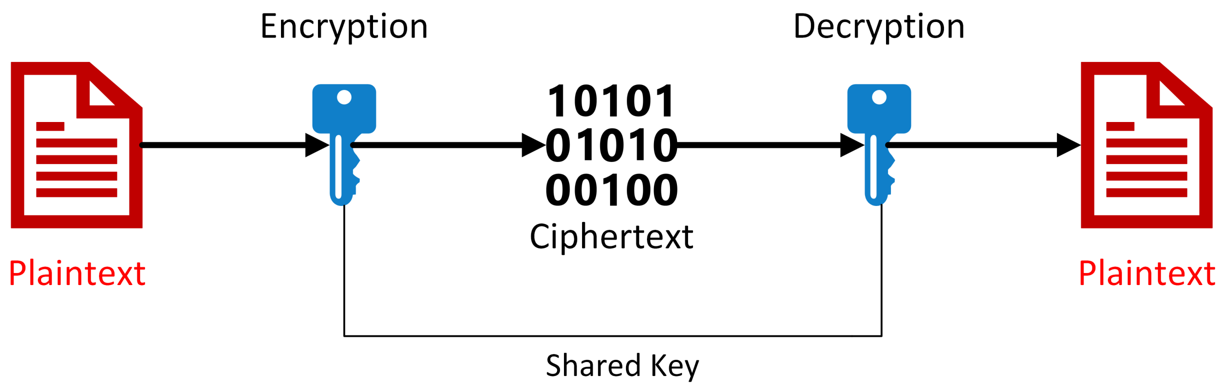

Figure 1.9: Illustration of symmetric encryption. Both the transmitter and receiver use a shared secret key to encrypt and decrypt the data.

AES was developed to provide a high level of assurance that any data encrypted cannot be accessed unless properly decrypted by a shared symmetric key [181]. Currently, AES is the industry standard for data-at-rest for many commercial and private sector security schemes, and hardware accelerators have been designed to increase the speed and efficiency of the protocol for use in many devices. The strength of AES is unquestioned, but in the realm of IoT, it begins to lose some practicality. For instance, much of the data in IoT systems is not at rest, but rather constantly in transit. The data itself may be of little value, but the transmission channel used must still be encrypted to ensure security and robustness of the entire network. This dynamic data necessitates a more flexible solution that can be used when less than the 128-bit AES security is required.

Key Management

Another primary area of focus for IoT InfoSec is key management. Since many cryptographic-based protocols require keys for either asymmetric or symmetric functions, managing those keys in IoT introduces new challenges. For instance, typically in order for two devices to perform the same cryptographic functions, they will share a time-synchronized symmetric key, as in symmetric encryption schemes. Many times an asymmetric key exchange method is used, such as the Diffie-Hellman algorithm. A different approach to asymmetric key transfer and the numerous handshakes required to perform them requires synchronization between devices. These devices may then be able to independently generate the same symmetric key needed to perform encryption and decryption. By removing the need for costly asymmetric protocols, IoT devices can achieve better energy efficiency, prolonging the lifetime of the device. In addition to secure, efficient session key generation, key storage and key revocation are critical factors in ensuring data integrity. Proper precautions must be in place to secure the keying material after it has been generated to block attackers from gaining access to past, current, or future keys. If a key does become compromised, a swift key revocation procedure must be enacted to ensure minimal impact. Moreover, if any aspect of key management is compromised, then the entire cryptographic process is vulnerable to attacks. Compounding this concern is the expected sizes of IoT networks. Possibly thousands of devices will comprise a network, and if proper InfoSec is not embedded, then the overall system is subjected to many possible attack entries.

1.4.5 Network Attacks

The big draw of IoT is the Internet connectivity it offers to previously offline products. As IoT continues to grow, more devices in the wild yields more attack surfaces and devices targeted for common network attacks. Due to the lack of resources, these IoT devices may be more susceptible to attacks. Current robust systems still contend with attacks that can knock them out of service, so new techniques must be researched and implemented so a larger network collecting massive amounts of data by smaller devices can still be reliably secured to an acceptable level. Currently, the use of firewalls and intrusion detection/protection systems provide the upfront network security for many systems. However, even at the current number of devices, these protections are not foolproof. Drastically increasing the scale of these systems with heterogeneous IoT devices will only add more attack vectors, and IoT devices may behave differently than current static computers and servers, requiring additional firewall rules.

Once an attacker successfully gains access to the network, many different attacks can be carried out, such as eavesdropping, MITM attacks, DOS or DDOS attacks, ransomware, or even data exfiltration [21, 70]. These attacks are not isolated to only IoT since they stem from current Internet applications, but the reduction in resources and capabilities of IoT devices limits the ability to impose traditional network security. As previously mentioned, device authentication and authorization is a major challenge area in IoT. Ensuring that devices are properly authenticated and have the correct authorization access can limit the ability of an attacker to infiltrate a network by impersonating these devices. Also, trade-offs in network architectures are made to limit the impact of such attacks. In hierarchical network topologies, such as a star-of-stars design, affected branches of the network can be isolated from the rest of the network, reducing the overall impact. However, some applications in IoT warrant more ad hoc and mesh-type networks where any device can talk to any other device. This can proliferate attacks more easily through a network since there are no single points that an area of a network can be isolated.

1.4.6 Latency

Latency is an important consideration in many IoT systems. Robotics, security systems, and safety-critical processes rely on stringent latency on time-sensitive networks (TSNs) in IoT [201]. If required timing-constraints are not met, then efficient robotic movements will fail causing manufacturing errors and delays. Likewise, in security and safety systems, critical timing windows are imperative to reduce impacts of an alert or failure. In terms of attack detection, latency can be analyzed to detect or reduce MITM attacks due to the fact that the attacker will add latency to the system while performing the attack. Latency can also be used to aide authentication by providing a level of assurance that a device is within a certain distance from the authenticating device. For instance, if a spoofing attack is not located near the target device, then the latency of the spoofed transmitted signal may raise an alert if it is outside acceptable bounds. This method is not foolproof, but it does provide additional information for the protocol to use.

1.4.7 Wireless Communications

Traditional short-range wireless communications favor TDMA, FDMA, or carrier-sense multiple access with collision avoidance (CSMA/CA), such as Bluetooth and Zigbee (IEEE 802.15.4) [30]. However, as the number of devices increases and the necessity for asynchronous timing requirements rises, these wireless techniques begin to become less scalable. Current research is looking at utilizing receiver assigned code division multiple access (CDMA) spread spectrum techniques that will allow an increase in network sizes and meet more robust timing requirements [134], [150]. Spread spectrum techniques also have an added benefit in the form of more secure channels due to a lower probability of interception/exploitation, while also supporting unscheduled multiple access channels. The major concern with spread spectrum protocols is the receiver design, but in many cases the communications will be asynchronous to limit the complexity of IoT edge node design. New receiver designs are also being studied to decrease this complexity as well. If spread spectrum techniques are used, this may increase the SWaP, but it will also add extra channel protections. Also, spread spectrum techniques will be able to be able to handle the increased number of nodes and and associated timing restrictions [150].

1.4.8 Version Control & Updates

Similar to the fact that resource-constrained devices have limited anti-malware capabilities, once these devices are fielded, they may lose the ability to be updated as it is unlikely that a technician can easily access and update fielded devices in many of the expected environments. Other devices may be able to incorporate over-the-air (OTA) updates or re-keying. These updates may include firmware or software updates as required. Re-keying may be necessary when a device's current key has been revoked or synchronization errors occurred due to key mismatch. In order to perform these OTA update/recovery operations, the device should leverage the use of a securely stored pre-loaded key (at time of manufacturing or network assignment) and successful device authentication should be performed. A concern to OTA operations is that they may be observed by a potential attacker, causing the concern of reverse engineering and possible replay attacks. Therefore, the security life cycle of a device should be a major design tenet in the IoT realm. Key management should be secured, and flexibility for performing future updates should be considered.

In addition to the expanding number of attack surfaces susceptible to network attacks, many of the IoT devices lack the resources to have proper anti-malware protection in case these attacks run commonly known malware. Although there are many anti-malware/virus solutions for home computers, such solutions are not generally ready for many IoT devices. Currently, secure bootloading and internal file checking are the primary methods to detect if a device has been compromised through software. These methods typically only run at boot-up, so there is no continuous scanning due to the resource usage costs.

1.4.9 Physical Attacks

An overwhelming appeal of IoT is the benefit of utilizing thousands of sensors and controllers throughout a given environment. Physical security of these devices needs to start with the supply chain itself. A recent discovery of computer servers containing a malicious hardware backdoor inserted during their production stages was perpetrated by China and affected approximately 30 U.S. businesses and government entities [23]. This attack highlights that every aspect of a device must be considered in its security life-cycle. If the supply chain and manufacturing phases are assumed secure, another concern is the remote or nominally inaccessible locations that these devices can be placed. One caveat to this implementation is that it is difficult to provide physical security options such as locked surroundings or video surveillance of these devices' locations. It is not an unlikely scenario that an attacker could locate and attempt to compromise a device in the open.

Physical attacks can be reduced through many different techniques. The aforementioned secure building access or surveillance can provide some oversight, but other devices will not be able to be monitored. For some applications, the use of anti-tamper protections may outweigh the cost of losing a single sensor, but in other cases, the attacker may be able to gain knowledge of the device's network that can cause a larger impact to its operation. If a device may contain sensitive data, anti-tamper techniques should be implemented such as log-in access timeouts, memory erasure, or self-destruct. The log-in access checks may allow a device to indicate tampering and log the failed attempts, but the other methods would render the device useless. Depending on the application, logging may be adequate to allow a technician to examine the problem, but more sensitive data applications may require the destruction of a device to keep the information secure from an attacker.

The abundance of these devices combined with their low-cost will make it easier for attackers and hackers to obtain devices for analysis. Even with safeguards in place, physical attacks are still assumed to occur, and devices will be compromised. These compromises may range from device destruction to attempts at reverse engineering, power analysis, and data extraction. Therefore, the primary concern is that compromises should not cascade into full network compromises. If no further knowledge can be ascertained from a captured device, than the remaining system should still be secure.

1.4.10 SWaP

The final consideration for IoT devices is SWaP. The reduction in SWaP reduces the capabilities of these devices, which allows for more cost-effective mass production. This reduction in capabilities is the largest impact of security of IoT. In order to lengthen the lifetime of a device, emphasis is placed on minimizing the processing power, memory, and battery size. Moving forward with security though, device manufacturers must find a balance in SWaP reduction and effective security of these devices. At this time, due to the infancy of IoT and its lack of security, SWaP concerns are not as critical, because as research and technology continues in these areas, new solutions should help reduce SWaP. Although SWaP is the only challenge that does not directly provide security in IoT, it is the only item in this list that is affected by all the others in a quantifiable metric. Overall, the resource constraints of many devices will drive the need for the preceding security techniques to be efficient.

1.5 Motivation for Efficiently Scalable Security in IoT

As IoT is the convergence of the expansion of the Internet and the reduction of devices' sizes and resources, efficiency becomes a major design choice. With the vast numbers of these devices spread throughout environments, security cannot be an add-on, it must be an integral design parameter. At this time, there are no established IoT security design principles, which has led to the abundance of research in this area, but also hinders the future growth of IoT. Future IoT device security should focus on appropriate embedded security and not on current security solutions designed for a more resource-rich paradigm. Due to the increasing number of IoT devices, security standardization must be accomplished in the near future because once many of these devices are deployed, they may not have the ability to be upgraded or patched, which can lead to future vulnerabilities.

The chart shown in Figure 1.10 again highlights the current need for and impact of not addressing the previously addressed IoT security challenges. Also included in this figure are the results of this dissertation's research that support the higher prioritized challenges. The number one issue with overall IoT security is determining standards that effectively secure individual sectors, but still provides flexibility and coordination across all areas. This lack of

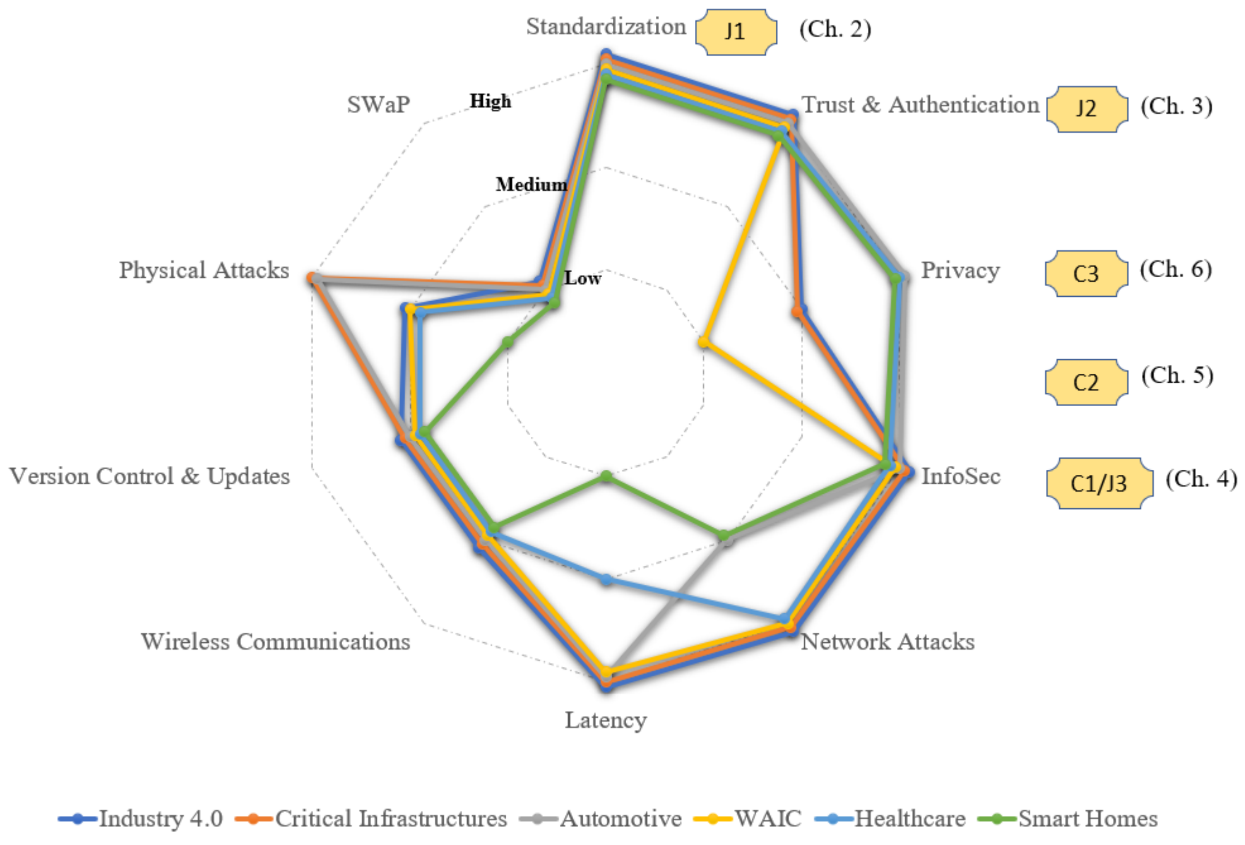

Figure 1.10: Illustration of where the contributions of this dissertation support the highest priority challenges in IoT.

standardization creates many vulnerabilities, because not all realms of IoT may view all the security needs equally, causing great concern for billions of IoT devices with differing security goals and leaving many networks vulnerable to attacks. A security level-based framework is warranted due to the absence of IoT standards. Implementing this framework will allow different systems to have a greater sense of confidence of outside devices that may be able to communicate on a shared network. While investigating the lack of security standards in IoT, this research focused on a secure-by-design approach starting at the lowest layers of the security stack. Two main areas arose to benefit from lower layer solutions: physical device authentication and key management. Therefore, the main research contributions of this document focus on developing a framework for security levels based on a device's resources and capabilities while implementing these new techniques. Beyond these main contributions, other minor contributions were presented in the areas of efficient symmetric encryption and transmission security. Although both network attacks and latency also have higher impacts on IoT, they are not specifically addressed in this dissertation. Network attacks are still a current topic of research in overall Internet and network security. Moreover, network attacks are planned for future work that will also examine size and scalability of future networks.

Latency was previously addressed in another publication that focused on high-order phase shift keying signaling (HOPS) [134] and must still be further researched as well.

1.6 Outline and Research Contributions

As Figure 1.10 illustrated, the contributions of the following publications mainly focused on the four highest priorities of IoT security. Providing solutions to these challenges will further allow security as a design foundation, instead of an afterthought. The following contributions filled in missing gaps in different areas of IoT security. Some of the contributions are major steps forwards, especially in terms of standardizing IoT device security and authentication through specific emitter identification. The other contributions add tools for low-power privacy protection and information security techniques designed specifically for resource-constrained IoT nodes.

Chapter 2 examines the lack of standardization across many IoT devices at the hardware and physical levels. This standardization is critical for certifying the capabilities of IoT devices. This chapter will propose different classes of device architectures based on available security features. As one of the major contributions of this dissertation, characterizing devices will aid in improving security in heterogeneous networks because all devices would have defined security feature-based capabilities. Chapter 3 presents a physical identification/authentication technique based on the unique differences of individual components as they impart small differences in transmitted signals. Using neural networks, these perturbations can be analyzed to differentiate and identify specific emitters. The main contribution of this chapter allows the actual identification of a physical device compared to authentication of data that may be compromised by an attacker. Next, a novel key derivation function is proposed and evaluated in Chapter 4. The main focus on this contribution is to reduce the computational complexity required to allow resource-constrained devices the ability to efficiently generate cryptographic keys for varying applications. The following chapters present minor contributions that aided in the overall main contribution of Chapter 2 by developing low-power techniques to reduce privacy concerns and increase InfoSec capabilities. Chapter 5 developed a low-power stream cipher that accepts variable length keys for greater encryption strength flexibility in resource-constrained devices. As a second minor contribution, Chapter 6 introduces a method to obfuscate a spread spectrum signal by purposefully introducing phase error to each transmitted spreading chip that may or may not be fully removed by the receiver. The effects of the induced error are presented and simulations are run to evaluate the performance impact to the receiver. This obfuscation increases the difficulty for an attacker to correctly observe transmitted signals and reverse engineer the underlying security functions. Finally, conclusions and future work are presented in Chapter 7.

1.6.1 Journal Manuscripts

1. Jason M. McGinthy and Alan. J. Michaels. Secure industrial internet of things critical infrastructure node design. *IEEE Internet of Things Journal*, 2019. ISSN 2327-4662. doi: 10.1109/JIOT.2019.2903242

   This paper presents a candidate security level-based architecture for low-power IoT-CI devices implementing modular, low-power, security primitives that are shown through simulation models and embedded software implementation to create a robustly layered defense-in-depth IoT architecture. This candidate architecture will help provide a foundation for future IoT-CI device security standardization.

2. J. M. McGinthy, L. J. Wong, and A. J. Michaels. Groundwork for neural network-based specific emitter identification authentication for IoT. *IEEE Internet of Things Journal*, 2019. ISSN 2327-4662. doi: 10.1109/JIOT.2019.2908759

   This manuscript proposes neural network-based specific emitter identification (SEI) for use on IoT devices and networks. This work laid the groundwork to determine how well neural network-based SEI will work on IoT devices. The case for physical device authentication in IoT is presented, and the use of devices' transmitted signals' unique physical characteristics are used for device identification. Results showed that the implemented SEI algorithm could be executed quickly to reduce latency impacts of device authentication on an IoT-like device.

3. Jason M. McGinthy and Alan J. Michaels. Further analysis of PRNG-based key derivation function. *IEEE Access* [**SUBMITTED**]

   This article expands on the pseudorandom number generator (PRNG)-based key derivation function (PKDF) that was originally introduced in a prior conference publication [129]. Further analysis is done on the session key outputs of the KDF with different underlying PRNGs to ensure proper randomness tests are satisfied. Performance evaluations are compared on IoT type devices against a current industry standard KDF and indicate exceptional computation and time savings. Finally a hardware prototype is designed for future evaluation.

1.6.2 Conference Papers

1. Jason M. McGinthy and Alan J. Michaels. Session key derivation for low power IoT devices. In *2018 IEEE 4th International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing, (HPSC) and IEEE International Conference on Intelligent Data and Security (IDS)(BIGDATASECURITY/HPSC/IDS)*, pages 194–203, May 2018. doi: 10.1109/BDS/HPSC/IDS18.2018.00050. URL `doi.ieeecomputersociety.org/10.1109/BDS/HPSC/IDS18.2018.00050`. [**BEST PAPER**]

This paper presents a simple, yet very effective session key derivation function (KDF) based on a pre-shared master key and PRNG. The method is PRNG based and allows for variable length keys to be generated. The results passed NIST's tests for randomness, and the algorithms were implemented into an MSP430 microcontroller for energy consumption metrics.

2. Jason. M. McGinthy and Alan. J. Michaels. Lightweight internet of things encryption using Galois extension field arithmetic. In *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 74–80, July 2018. doi: 10.1109/Cybermatics_2018.2018.00046

   This publication presents a lightweight cryptographic scheme for use as a stream cipher based on Galois extension fields (GEFs). This method produced nearly-uniformly distributed ciphertexts that pass NIST's randomness test suite. This method also lends its use as an order-independent multi-party encryption technique in which the order of encryption and decryption do not change the resultant decrypted plaintext. This scheme was implemented and tested on an MSP430 microcontroller to determine performance metrics.

3. Jason M. McGinthy and Alan J. Michaels. Semi-coherent transmission security for low power IoT devices. In *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 170–177, July 2018. doi: 10.1109/Cybermatics_2018.2018.00059

   This paper presents a physical-layer (PHY) semi-coherent TRANSEC technique that adds a random phase error to each chip of a spread spectrum modulation. This effectively obfuscates the true phase of the signal that may be used by an observer to reverse engineer security measures of a system. Since the TRANSEC occurs at the PHY-layer, it is a simple, yet cost effective (in terms of power and computation) security technique. Simulations showed that phase shift keying (PSK) techniques can incorporate uniform or normal distributed phase errors with expected results.

Chapter 2

Feature-based Security Standardization

As previously presented, standardization is the biggest challenge in IoT due to the numerous inexperienced and experienced manufacturers all vying to become the market leader in their respective areas. Currently, there is a widespread deployment of IoT devices and a lack of standards, which all contribute to a security nightmare in IoT's infancy. Multiple vendors may create similar devices, but they will follow completely different design recommendations. Many of these IoT devices tack security on as an afterthought [144], causing a greater need for industry standardization. A 2017 IoT security survey showed that 96% of the businesses believe that there should be regulation for IoT security [22]. A great concern within IoT is device trustworthiness, and at this current time, the lack of standards does not create a great sense of trust amongst these devices. In fact, NIST is trying to determine IoT standards for future operability across many domains [152] in order to protect the future of IoT devices and networks. Without this type of standardization, it only takes a simple, mass-produced IoT device with little to no security to compromise a local or larger network, such as the Mirai botnet attack [50]. Although the need for security of these devices is well documented, there is an absence of a security level framework for CI nodes. Therefore, the goal of this article is to present a secure CI device model with validated low-power security primitives to ensure these systems are secured from the lowest layers based on secure feature-based security levels. Moreover, this proof-of-concept architecture lays the foundation for designing secure IoT-CI devices and integrating them into a robust and scalable CI network. This chapter details the general layered view of IoT systems and related work in solving IoT node security. A proposed security level nomenclature is presented to guide the basis for the proceeding secure design features. It then details the general security features for a secure node architecture as a framework for all future devices to be characterized.

2.1 Acknowledgements

This chapter, in part, is a reprint of the material as it appears in the publication:

- Jason M. McGinthy and Alan. J. Michaels. Secure industrial internet of things critical infrastructure node design. *IEEE Internet of Things Journal*, 2019. ISSN 2327-4662. doi: 10.1109/JIOT.2019.2903242.

The dissertation author was the primary author and Dr. Alan Michaels supervised the research which forms the basis of this chapter.

2.2 Background

IIoT systems will bring about unprecedented complexity in their designs, so one must have a general understanding of how an IIoT system is structured. These systems are comprised of numerous IoT edge nodes connected by access points (for data aggregation and analysis) to form larger networks. These edge nodes may take the form of environmental sensors, actuators, security latch or proximity sensors, and may also perform currently unimaginable functions for these systems. All devices in these systems must be designed with security in place, from the edge nodes to the access points and beyond.

IIoT and wireless sensor network (WSN) systems are commonly composed of three main abstract layers [84, 88, 99], as illustrated in Figure 2.1. The application layer is composed of data analysis, services, and applications relying on the underlying data collection and transportation of the lower layers. It is the forward facing interface of IoT. The network (or transportation) layer consists of data aggregation and possible computations. This layer also controls all communication, not only amongst devices in a network, but also amongst multiple possible heterogeneous networks. Finally, the physical device itself is referred to as the perception layer. It is at this lowest layer that this work focuses, as the physical security of the device and security of data at rest and transit in the device are a major concern since these devices are typically remote and inaccessible for easy problem diagnostics. Although much of the concerns for IoT are currently focused at the network layer or higher due to its heterogeneous nature, recent research [99, 101, 118, 159] for these resource-constrained devices has been conducted for proper security at the lowest level (perception) to help ensure security at the higher levels.

Many current publications have outlined the security challenges and risks in IoT [115], [113], [59], [210], [89], [26], [142], [112]. As previously mentioned, many of the IoT security architectures currently focus at the network level or above due to the heterogeneous nature of IoT, so that more devices can communicate with each other, but the actual device security may still be lacking. For instance, TLS [163] and Datagram TLS (DTLS) [164] are common security schemes that operate above the perception layer. DTLS tends to be favored in some IoT systems due to its better performance (due to no guarantee of data vs. TLS's data guarantee). These protocols work well for device communication over networks, but if a device is exposed prior to network communication, then attacks can still be performed.

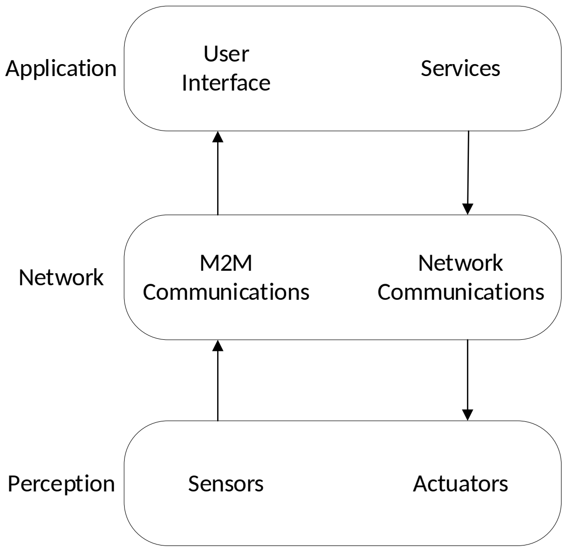

Figure 2.1: Three layers of an IIoT system.

Work in [159] presents the concept of a trusted architecture for farmland wireless sensor networks in order to improve reliability in these human-less systems. At the perception layer, a perception logical layer is combined with a mark recognition logical layer to detect abnormal sensor information. Trust is crucial in WSNs, and the protocol in this work was shown to be feasible in those networks. However, actual security of the perception layer device and data was absent. In order to establish trust, the device and data security must also be designed along with the trust architecture.

Other research from [101] highlights security issues at the perception layer and offers enhancements to system on a chip (SoC) devices. Although their work emphasizes Trusted Execution Environment (TEE), the paper only presented security features needed for secure architectures without a true implementation plan. Babar, et al. [38] present a generic embedded security framework for IoT. Their work highlights the importance of device security while understanding the lack of resources for many of these devices. Their approach takes into account the cost, performance advantages, and disadvantages of hardware, software, and hybrid solutions, Their solution still only presented an abstract framework, which this dissertation aims to expand upon.

2.3 Feature-based Security Levels

IIoT networks will be composed of numerous heterogeneous devices that will have differing levels of security associated with their purpose. As previously discussed, there is a great need for properly securing these different classes of devices, but there has been very little research in forming a generally accepted process for determining security levels based on device classifications. Table 2.1 provides four candidate security levels based on selected security features implemented in IIoT devices. These devices may range from unsecured environmental sensors with no perceived security implications to access points that communicate with hundreds of edge nodes aggregating sensitive data that must be protected and varying degrees of secured devices in between. The security features shown in the table will be expanded upon in the proceeding section.

In order to create a concrete framework for comparing the security of IoT nodes, this section presents a discrete listing of node classes and the baseline capabilities of each corresponding to the aforementioned security levels. As previously shown in Table 2.1, depending on the desired security level of the device, there are different features and levels that may be implemented. Some factors that help classify the actual required security level of these devices are expected use, lifetime, and cost of such a device.

Table 2.1: Security Levels Based on Selected Features

		Level 0	Level 1	Level 2	Level 3
Physical Device Security (Anti-Tamper)		-	-	✓	✓
Trusted Execution Environment		-	-	✓	✓
Data Logging		-	✓	✓	✓
Memory	Unprotected	✓	-	-	-
	Protected	-	✓	✓	✓
Encryption Strength	None	✓	-	-	-
	Low	-	✓	-	-
	Medium	-	-	✓	-
	High	-	-	-	✓
Message Validation Strength	Low	✓	✓	-	-
	Medium	-	-	✓	-
	High	-	-	-	✓
Physical Layer Security	None	✓	-	-	-
	Low	-	✓	-	-
	Medium	-	-	✓	-
	High	-	-	-	✓
Security Parameters	Static	✓	✓	-	-
	Adjustable	-	-	✓	✓

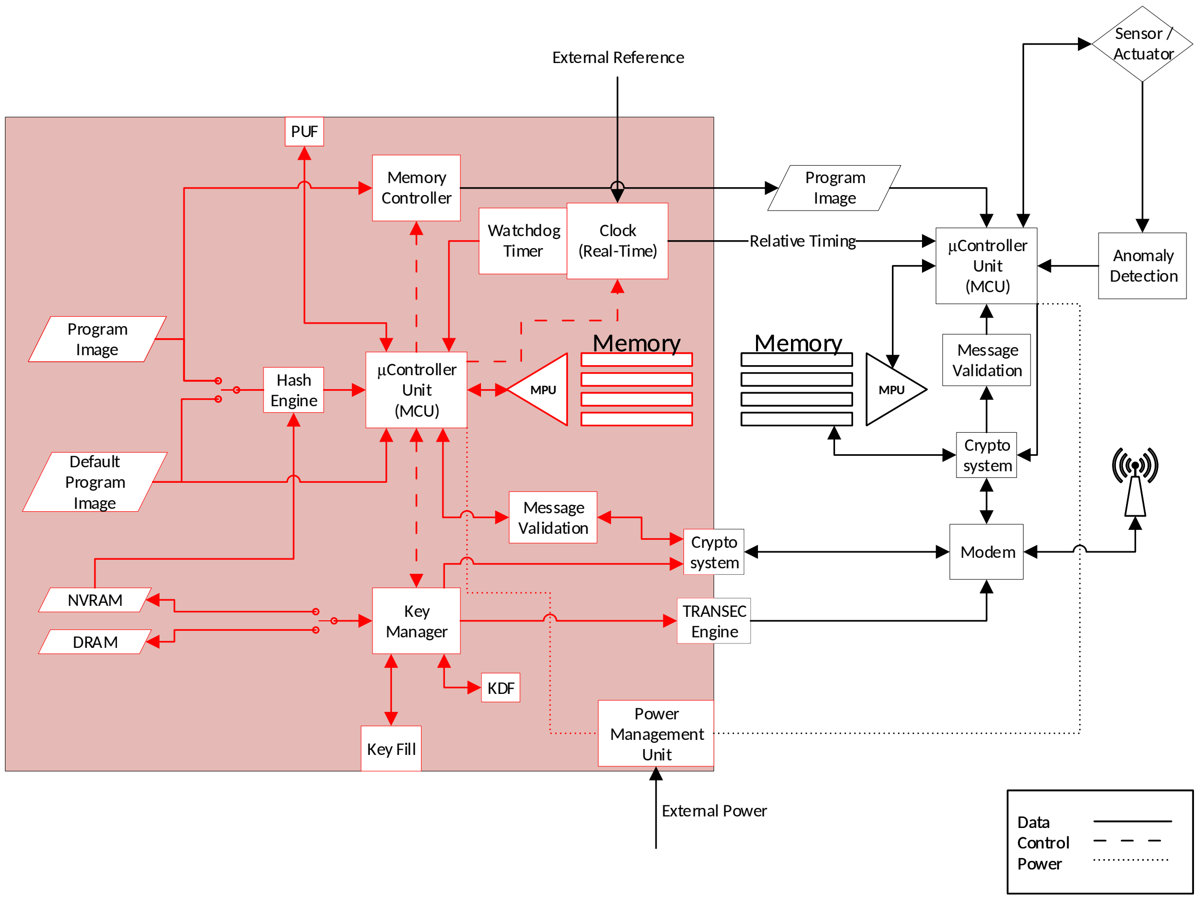

Figure 2.2: General Class 3 Secure Node Architecture. The red area represents the secured partition of a traditional red/black architecture [33].

2.3.1 Class 3

A Class 3 device includes all possible security features making it the most secure device available. It will contain the highest level of encryption, largest memory capacity, fastest MCU to allow for possible data aggregation and analytics from less capable devices. This device may have a constant power supply (direct current (DC) powered) or sufficient batteries (possibly rechargeable) in order to perform the higher computational costs of stronger security operations. Figure 2.2 shows an ideal architecture embodiment of a class 3 device. It is designed with all possible security functions and features and may be realized as a more highly capable device similar to cellular phones or laptops.

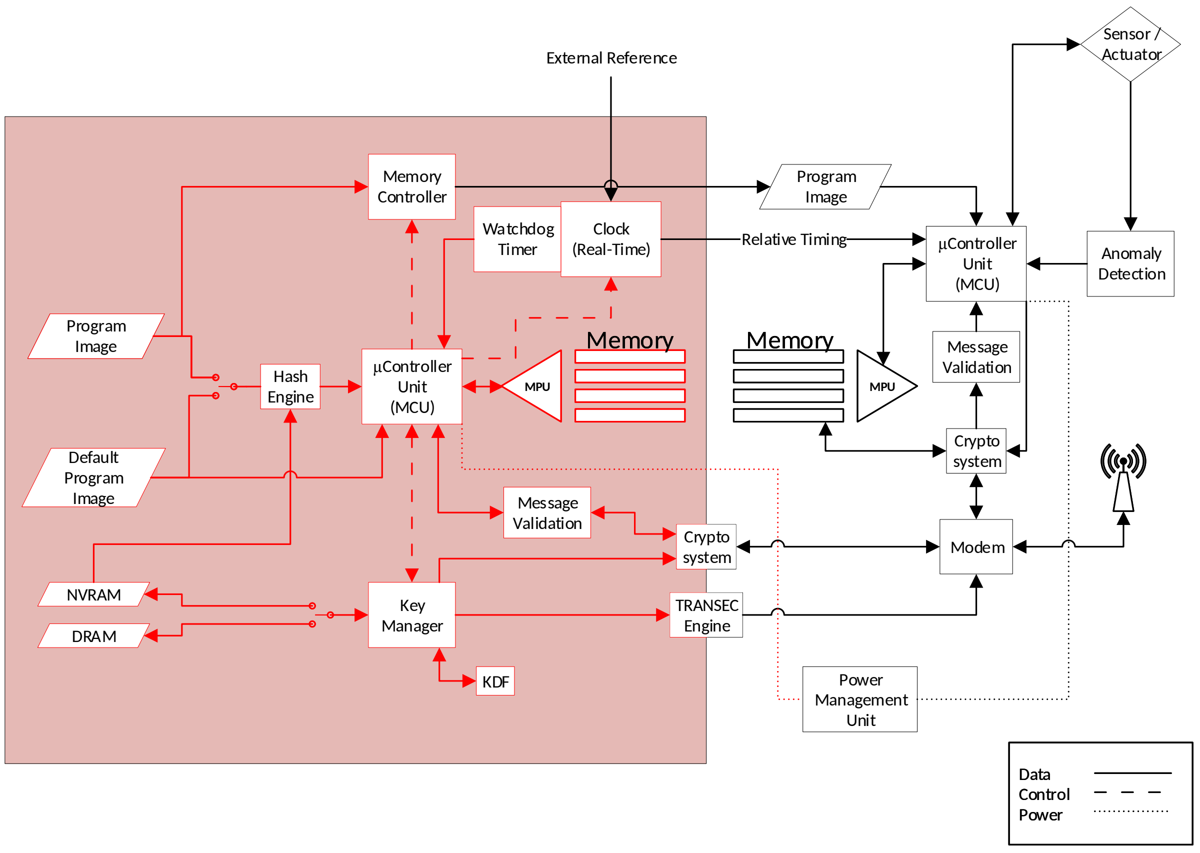

Figure 2.3: Example of Class 2 device architecture. This device is similar to the higher Class 3 device, but omits features such as a PUF and key fill functionality. Requirements placed upon security computational elements (cryptography, message validation, etc.) are also lessened.

2.3.2 Class 2

A Class 2 device will need to be more energy efficient compared to a Class 3 device as it will ideally not have a large power supply. Extra security features such as a physically unclonable function (PUF) or the ability of attaching a key fill device will not be available. The device may be well suited to be deployed in remote unsecured areas, but this will require the need for anti-tamper, a TEE, medium encryption, message validation and PHY layer security. This device also may have SoC components that allow for programmable parameters. An example architecture of this device is shown in Figure 2.3. A class 2 device may not be as capable as a cellular phone, but it is still a higher-functioning embedded device that has the ability to be accessed by authorized personnel in order to change batteries to reduce the impact of energy consumption.

2.3.3 Class 1

Class 1 devices will have a minimal level of security functions implemented and would not have a TEE. This type of device may be implemented as a basic application-specific integrated circuit (ASIC). Therefore, security functions will be initially programmed and remain static during the device's lifetime negating the ability of future updates. An example architecture of this device is shown in Figure 2.4. This may be a very basic sensor or actuator that is located in a secure environment, which removes the requirement of anti-tamper, but may not be easily accessible or cost effective to change out batteries.

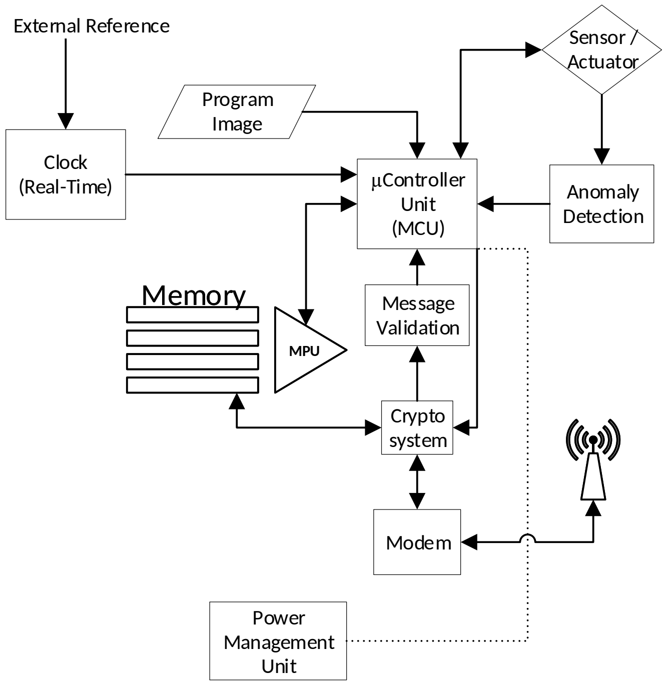

Figure 2.4: Example of Class 1 device architecture.

2.3.4 Class 0

This Class 0 device is an extremely basic device that will have little to no security features deployed for maximum speed or low-power functionality; data processed and transmitted has minimal need for authentication or resiliency. These devices may utilize RFID technology or smart-chip embedded cards. Its power supply will be extremely limited (such as energy harvesting or passively powered by radio waves) and will perform very little computations on the device. This type of device is ideally suited as a low cost environmental sensor (temperature, pressure, light, etc.). If this device fails or is compromised, there are generally

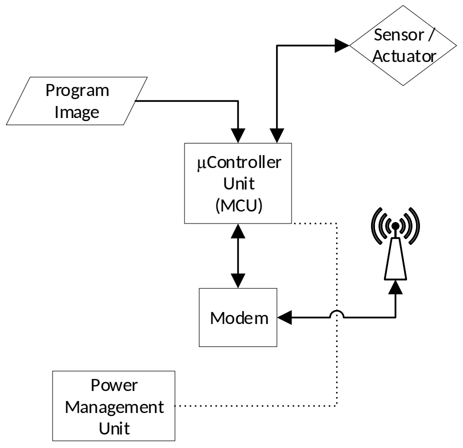

Figure 2.5: Example of Class 0 device architecture.

redundant sensors nearby to support the overall operation. An example architecture of this device is shown in Figure 2.5.

Revisiting the devices infected by the Mirai botnet attack, many of these devices had a primary function, such as video cameras and digital video recorders, that then had Internet access added on later as a feature, but did not have security properly implemented [50]. Many of these devices had weak passwords for access that then allowed the virus to infect the device's code to perform the DDoS attack. These devices did not meet many of the security features shown in Table 2.1 to meet a Level 0 classification, but their intended functions probably warranted at least a Level 1. This highlights the current situation in many first to market IoT devices with security bolted on afterwards. Therefore, future IoT, IIoT, and WSN systems must include security in their initial designs to avoid current security shortcomings of many IoT devices.

Currently, devices are produced with minimal security concerns for their own well-being, which compounds the security of many IoT devices communicating in a system. Moving forward, once manufacturers and vendors are able to produce products based on regulations or standards, then proper security design practices will allow better interoperability amongst heterogeneous devices on IoT networks. Although many businesses and consumers agree that regulation and standards are important for IoT, until these standards are agreed upon, the baseline security of IoT will still be questionable.

2.4 General Security Features

A secure IIoT or WSN sensor node is nominally designed to be low cost, energy efficient, and reliable. In order to achieve all of these goals, certain design security features must be considered. Figure 2.2 highlights a general architecture design of a security level 3 IIoT device based on the aggregation of the following features. This architecture can be reduced to fit the security design goals of other (possibly less secure) devices, but for the purpose of completeness, all features will be discussed in the following sections.

2.4.1 Physical Device Security

In order to secure a system against physical attacks, several precautions must be taken into account in the design of a device. For example, physical security must begin at the manufacturing and production phases of IoT devices. If a device is tampered with and malicious code is added to a device during this period, then further security is likely moot. Beyond the supply chain, manufacturing and production areas, these devices should have some anti-tamper protection in place to protect the sensitive data that may be contained in these devices. Some anti-tamper methods include log-in access time-outs, self-destruction and memory erasure [205] and PUFs [183]. The log-in time-outs are similar to smartphones' failed password attempts in order to combat DoS attacks. Complete memory erasure would completely remove a device from a network without indication, whereas using a PUF may allow detection of tampering due to an incorrect response. Unrecoverable security responses must be balanced with the intended use case since events that trigger these mechanisms have the same effect as a DoS attack. For the purpose of this research, we assume that intelligent network provisioning of secure information is performed so that corruption of one device yields limited added utility in corrupting other devices in the network.

2.4.2 Trusted Execution Environment

Beyond physical security, a designated secure portion of the sensor should be designed to perform all aspects of security, but the design must heed the constraints of such a resource-limited node. Still, an IIoT secure device architecture can be separated into two main partitions: secure and non-secure, reminiscent of traditional red/black separated architectures [33]. The secure region contains all aspects related to the sensitive material and functions. In order to design a secure node, multiple operations must be performed in a TEE. The functions in the red partition are secured from the rest of the normal unsecured functions of the sensor, whereas the components outside the TEE may not be secure and therefore susceptible to malicious attacks.

Currently such environments are found on larger systems with defined instructions such as Trusted Platform Module (TPM) [6], TrustZone [2], and Intel SGX [4]. Such ports to smaller IoT devices will provide a large benefit to remote nodes. For instance, the new ARM Cortex-M23 [1] has been designed to be extremely efficient for IoT needs, while also having TrustZone technology built into the processor. These new processors, while promising for future IIoT and embedded devices, will still pose a trade-off between higher costs (manufacturing and operating) and more secure designs. As more research continues in this area, these trade-offs should be minimized.

2.4.3 Memory

Memory is protected through the use of a memory protection unit (MPU). The MPU is able to designate different memory regions and limits access through memory access policies, such that only certain functions are able to access memory in certain regions, therefore restricting access to other areas. This limits the memory space that can be affected by a single process, mitigating buffer overflow attacks [47].

Different types of memory must be used in a secure node architecture. The code required for booting up the device and performing minimal operations (in case of power loss) shall be stored in read only memory (ROM) as it should not be modified except in a trusted environment. This is shown as the default program image in Figure 2.2. The ability to perform attestation through a hash engine and comparison to a known value shall be able to be performed by the red microcontroller unit (MCU) in order to test the integrity of the default program. Non-volatile random access memory (RAM) (NVRAM) may be used for a re-programmable image. Utilizing this type of memory allows for code updates. For both the default and programmable images, the code size must be well defined so malicious code may not be injected.

The actual key catalog will also be stored in NVRAM to allow key and cryptographic material to be updated. This NVRAM would be separate from the program image and protected by the MPU. The key catalog may also contain volatile RAM (either dynamic-RAM (DRAM) or static-RAM (SRAM)) to store keys that have a short lifetime as compared to other cryptographic keys that may have much longer lifetimes (e.g., seconds vs. days). Recent novel techniques for the pre-calculation of intermediate cryptographic information for intermittent energy harvesting systems using non-volatile ferroelectric RAM (FRAM) has also been proposed [185]. The MPU ensures this memory is protected from unauthorized function calls.

The non-secure portion of the node will have the same type of memory structure with a MPU, but will be physically separated and have no data flow to the secured memory locations. This physical separation is necessary to protect against certain memory attacks such as Rowhammer [106]. The information stored in the non-secure memory should not have any ability to affect the secured portion of the device.

2.4.4 Data Considerations

Many of these devices may be susceptible to physical compromise if not deployed in a secure environment, therefore, special data handling procedures must be in place. Data at rest should be encrypted using as strong of a protocol as possible (i.e., AES) while the cryptographic keys should also be wrapped (encrypted apart from the sensitive data) in order to prevent leakage of material when not in use. These additional layers of encryption decrease the likelihood of an attacker successfully obtaining sensitive data stored in the device's memory.

Data flow in and out of the TEE are minimized to reduce the possible number of attack vectors. For instance, the only unsecured data that can enter the TEE does so through the cryptosystem. The cryptosystem then decrypts the message and sends it to the message validation function. If the message is validated, it is able to be used in the TEE. Other outside functions such as an optional PUF or key fill procedure would also need a method to validate that data entering the TEE may be trusted. The only data leaving the TEE is already validated program memory (sent from the memory controller once it is validated by the MCU), passed through a function that goes through a cryptographic process, such as encryption or a TRANSEC engine, or passed by the clock as a relative time (not real-time) reference for use outside the TEE.

2.4.5 Clocks and Synchronization

Symmetric cryptographic protocols rely on devices having the same key. Therefore, it is imperative for devices to be properly synchronized for the purpose of key derivation, communication, and other functions that may rely on a PRNG. Although in-depth synchronization techniques fall outside the scope of this work, the following techniques are examples that may be implemented on IoT devices.

The easiest method to understand is synchronizing based on Global Positioning System (GPS) timing. Although many personal devices (phones, smart watches, etc.) have built in GPS functions, low-power, intermittently powered devices will not be able to properly synchronize to GPS due to the lack of constant tracking of the satellites. Moreover, most of these devices will not have an open air view of GPS satellites. On the other hand, with the addition of an external reference, special design and implementations considerations must be taken to reduce the threat of a possible attack vector into the system.

Another synchronization technique is based on an IIoT device implementing a real-time clock (RTC) hardware chip. This method removes the need to communicate with GPS for timing and the timing can be kept local relative to a device. However, clock drift may occur amongst devices leading to de-synchronization amongst a large system. Therefore, combining a RTC with an aperiodic beacon signal from an access point would provide a more precise timing feedback to ensure clock drift is minimized through a system. The decision to use an aperiodic beacon adds unpredictability that may make it more difficult for an attacker to affect the system.

2.4.6 Power Management

Many WSNs are designed to be deployed without human interaction, but must have a usable lifespan of years. Additionally, energy consumption of IIoT devices must be minimized for all aspects of operation, including security. Therefore, trade-offs may be considered for levels of security dependent on the power available for a device. Devices that act as network gateways with a DC power supply may be able to utilize all security features with little impact, but current security schemes may not be an end-all solution for other non-DC powered devices. Another important power consideration is limiting the attack vector of the power flow to the TEE, as devices may be vulnerable to side-channel attacks, such as differential power analysis (DPA), through power consumption models [156].

2.4.7 Boot Procedure

In certain circumstances where a device is inaccessible and the device has the ability to power down to conserve energy (such as an energy-harvested based power supply), the device must be able to boot into a known and trusted state. Beginning with a device powering on, the node must go through attestation to ensure that none of the primary code has been modified through natural (device failure) or unnatural (malicious) means. Not only does it verify that code has not been modified, this procedure also provides a root of trust mechanism for the device for use on its network. Previous research such as DINO [116] and Clank [95] into the area of implementing checkpoint schemes on energy-harvested devices shows continuing promise for safe and secure restarts.

2.4.8 Key Management

Device key management acts as a state machine for all key-related blocks such as a PRNG, key derivation function, and other cryptographic processes. Key management contains both the key storage and KDF portions of a system.

Key Storage

Storage of cryptographic key material must be properly implemented due to the extremely sensitive nature of the keys. These keys are commonly used as seeding material for PRNGs and KDFs and may also be used for encryption/decryption keys and/or other cryptographic processes. Ideally, a small amount of key storage would be implemented in NVRAM to allow for keys to remain intact after intentional or unintentional power loss. Further, access to this key store should only be through validated interaction with a key manager, providing the least insight to the actual key material as possible (e.g., validation via hashes rather than access to root keys). For use in IIoT nodes that may rely on intermittent power, this allows keys to be stored for future use [185]. Nevertheless, most of the key storage could be implemented with volatile RAM because new keys would be generated after the device powers up again, but this may depend on the class of device.

Key Derivation Function

The KDF is a critical function of the overall security architecture. The KDF is the sole source of keying material for all primitives in the device. It must create reasonably strong cryptographic keys, while being fast and energy efficient. Typically, a KDF will take an input keying material and a seed to derive the output keys. An assumption of this research is that the key generation process must be deterministic, enabling repeatable generation of symmetric key material at the other side of a wireless link. If this process is not deterministic but based on a truly random process, it becomes a very difficult problem to synchronize symmetric keys amongst devices.

The hash-based message authentication code (HMAC)-based KDF (HKDF) [110] was designed using an extract-then-expand concept in order to take possibly weak initial keying input and create a cryptographically secure output. It begins with an initial input and extracts a key value. Then this key value is expanded to create variable length key derivations for use in a wide range of applications. Optionally a salt value (non-secret) and a specific application information value can be used in the derivation of keys. This HKDF is currently planned to be implemented in TLS 1.3 [163]. HKDF works very well if the input keying material has low entropy, but if the keying material is already significantly strong, then the extract phase requires additional wasted computations.

Previous work proposed in [129] performs key derivation with a secret master key and seed. It is assumed that the secret master key has sufficient entropy and would not require any expansion before the derivation process. The process pseudorandomly extracts bits from the master key to produce a derived key. This method is able to provide a variable-length keys indistinguishable from uniform distributions and passes NIST's pseudorandomness test suite evaluations [44]. Since this process is deterministic, it allows for synchronization amongst devices in order to perform operations such as symmetric encryption and coherent or semi-coherent TRANSEC functions [130].

2.4.9 Pseudorandom Number Generator

Random number generators (RNGs) are an important area for information security. Typically, they are needed to produce random bit streams to be used in key derivation and encryption functions. The better the randomness, the more difficult it is for an attacker to brute force (guess) actual data. Two main areas of RNGs are true random number generators (TRNGs) and PRNGs.

As the names imply, TRNGs are designed to be truly random with no repetition period of the number generation. TRNGs normally rely on an entropy source (source for randomness) such as environmental or atmospheric measurements [12], oscillator rings [184], or integrated circuit metrics from certain device components [186]. Some sources cannot rapidly produce enough entropy for a random value, decreasing the speed and which numbers may be generated [181] [200]. Since TRNGs are truly random, it is impossible to replicate a random value on another device for synchronization, which is important for asymmetric encryption schemes, but this is a limitation for use in synchronized symmetric systems. As such, TRNGs are suitable for functions like randomized backoff times for certain modulation techniques and PHY layer security methods.

PRNGs are designed to simulate truly random results, but they naturally have a cyclic nature. Well designed PRNGs will have a periodicity that approaches lengths that make them seem aperiodic. A PRNG uses a deterministic approach that can be replicated if known states are the same amongst devices. This deterministic property makes PRNGs very practical for synchronization purposes. For example, if two devices are using the same PRNG and seed it with the same value, then the output of the PRNGs should be identical which would allow implementation of symmetric processes, such as encryption or modulation parameters. This synchronization reduces the overall need to transmit seeding material (cryptographic keys) since each device can produce the identical values. The type of RNG used in a system depends on certain criteria such as speed, entropy source (for TRNGs), and need for synchronization (such as a symmetric encryption protocol).

2.4.10 Encryption

Encryption is critical to ensure data confidentiality. If an attacker is unable to decipher a message, then the contents of the data will remain intact. One major concern with IIoT and WSN devices is whether or not they have the resources to employ *strong* encryption schemes. The strength of the encryption at the lowest layers will also impact the cryptographic strength at higher network layers, so they may be trade-offs for devices depending on their capabilities and the capabilities of the networks. Therefore, different types of cryptographic techniques must be examined for use in these devices to provide the appropriate security level required for a device.[1]

Stream Ciphers

Stream ciphers are typically very energy efficient and fast encryption techniques, yet some classic ciphers have been shown to have vulnerabilities and others have had issues with implementations, so they may be valid for use in a low security level device. Linear-feedback

[1]The choice of encryption need not be the same on both the uplink and downlink. Asymmetry can be beneficial when one side one side of the link is more power/resource-constrained than the other.

shift registers (LFSR) are a simple logic circuit that can be used as a PRNG, stream-cipher, and as other cryptographic components [217], [172], [119]. The simplicity in their design and easy implementation have made them a good choice for IoT security [103]. More importantly, a LFSR is linear, deterministic, and reliant on strong seeding. Therefore, vulnerabilities for LFSR-based designs are a serious concern [143],[87].

Other stream ciphers have implemented LFSRs into their designs. Grain-128 [92] and Grain-128a [28] (updated version of the original) are both stream ciphers based on LFSRs and non-linear feedback shift registers (NLFSR) to generate a key stream that is then XOR'd with the data stream that were part of the eSTREAM cipher competition [165]. Grain-128 was found to be vulnerable to multiple attacks [37], [71], and currently Grain-128a attacks have been possibly discovered [117]. These vulnerabilities highlight the challenges in designing a low-cost, fast stream cipher for IoT applications and systems.

RC4 is another well-known stream cipher introduced by Ron Rivest for RSA Security in 1987 [181] that does not implement a LFSR. It was shown to be very simple, perform very fast, and was implemented into the WPA protocol for the IEEE 802.11 wireless LAN standard. However, vulnerabilities [31, 81, 82, 107, 120, 149] were discovered that have questioned the overall security of RC4 and removed it from being used in TLS [153, 155]. This has prompted the research into other stream cipher techniques for low-power, resource-constrained IoT devices. Due to the concerns of LFSR-based designs and RC4, new stream-cipher approaches must be researched.

Block Ciphers

Block ciphers are normally less energy efficient compared to stream ciphers, but many current implementations have proven to be very secure through the test of time. Therefore, certain block ciphers may be implemented for high level security devices. AES [181] is the industry standard in symmetric block encryption. It was selected in 2001 and is used throughout the world. It works on 128-bit blocks of data using encryption key length of 128, 192 or 256 bits. Many software and hardware platforms have been optimized to run AES to make it very efficient. Despite its design, concerning the capabilities of IIoT devices on resource-constrained platforms, AES may not be the most efficient encryption scheme based on power consumption for needed security strength.

Lightweight Symmetric Encryption Schemes

Due to the constrained nature of devices, current cryptographic protocols may not be practical so other lightweight schemes have been presented to help in the IoT realm. These types of schemes aim to be very energy efficient, but may not provide the highest encryption strength offered by AES. Therefore, these techniques would ideally be implemented in devices that required a medium level of encryption strength. As research into more lightweight schemes continues, the encryption strength should increase as well.

Two ciphers have been specified by ISO/IEC 29192-2 for use in lightweight cryptographic operations: PRESENT and CLEFIA. PRESENT [51] is a 64-bit block cipher with a key length of either 80 or 128 bits. It was designed to be very efficient in terms of area and power without compromising security and to be comparable in area size to modern stream ciphers. It is similar to AES in the fact that it is a round-based substitution/permutation structured block cipher.

CLEFIA [177] is a 128-bit Fiestel structure encryption algorithm. It is similar to AES in that it can take a key with a length of 128, 192, or 256 bits. CLEFIA employs a diffusion switching mechanism in order to reduce the number of rounds required and provide added security against differential and linear attacks. CLEFIA's design balances speed, area, and security. Work in [102] shows that CLEFIA is very efficient comparing throughput and area size compared to AES.

Another lightweight symmetric scheme based on novel Galois extension field (GEF) techniques were presented in [136], and the cryptographic application was expanded in [128]. This cryptographic stream cipher technique allows for extremely fast and energy efficient encryption/decryption for variable size messages. This approach also allows for multi-party encryption and order-independent decryption (the sequence of decryption does not need to be the same as the encryption). Chapter 5 provides further details on this technique.

2.4.11 Message Validation

Every message that is received by a device should be validated that it is the correct message that was sent. This may be done through the use of CRCs, message authentication codes, or authenticated encryption schemes such as AES Galois/Counter Mode [181]. CRCs offer a very basic level of message authentication as it is normally accomplished as a mathematical operations (typically summation) of certain bits. Message authentication codes tend to offer a higher level of authentication due to the use of cryptographic hash functions that indicate if a message has been modified during transmission. Extending beyond message authentication codes, authenticated encryption schemes attempt to combine both facets of message integrity and authentication, creating a more robust message authentication scheme, but at a higher computational cost. Validating messages is a very important stage when the information being sent may be overwriting current data, such as programs or cryptographic keys. If these messages are not validated, the data may be corrupted during transmission due to noise or malicious attacks. Once the data is validated, it may then be used to update stored parameters, perform commands for sensor collection, or allow actuator commands to proceed.

2.4.12 Hash Engine

Cryptographic hash functions, such as Secure Hash Algorithm 3 (SHA-3) [73], are ideally used to perform authentication of data. The most important properties of hash functions are their one-way process (non-invertible), such that it is practically infeasible to reconstitute a message from its hash output, and their collision resistance, which makes it infeasible to find two messages that produce the same hash value [181]. In a secure device architecture, a hash engine may be implemented in either hardware or software. A hardware implementation is normally more efficient (time and energy) and more secure against tampering compared to a software version, but the hardware is less suited to updates of hash polynomials or parameters. Moreover, a software implementation inside a MCU is still relatively fast for IIoT use and allows for flexibility in design for future updates.

2.4.13 Modulation

Wireless communication has allowed the expansion of devices into previously unmanageable locations. With wired connections, precautions were needed to ensure that proper routing was done so that wires would not be severed in normal operations. This dependence on a wired connection limited locations of some sensors. However, with enough signal strength, new locations are becoming increasing targets for sensor networks. Unfortunately, this wireless medium also allows unfettered access to transmitted data for anyone near the transmitter. Therefore, it is imperative to properly secure the wireless transmissions of WSNs and IoT devices.

Currently, the favored modulation types are TDMA CSMA/CA. TDMA allows multiple nodes to communicate to an access point by assigning time slots. This scheme is synchronous, so these time slots do not change amongst devices, allowing the receiver to know who is currently transmitting. One drawback is that devices must wait until their time slot to communicate, and if a device does not communicate during its slot, that time is lost for all the other devices. TDMA also does not provide any strong security guarantee as an adversary can easily interfere with it due to is periodic transmission based on timing channels.

CSMA/CA attempts to alleviate the dwell time of synchronous TDMA by allowing devices to transmit at any time, but this can cause collisions to occur if multiple devices try to communicate at the same time. Therefore, devices must sense if the wireless channel is in use when they want to transmit. If the channel is open, the device will attempt to transmit, but if the channel is sensed to be in use, the transmitter will wait a random backoff period and see if the channel is open. This technique allows for asynchronous transmission of data, but it also has its own disadvantages. For instance, the hidden node problem can occur when two transmitting nodes are out of sensing range of each other and continually try to transmit to the access point [90]. This will cause both transmitted messages to be lost as the receiver cannot distinguish between two signals at the same time. Another issues that can occur with CSMA/CA is a cascading effect of backoff times for devices. If the carrier is sensed to be in use, then the devices will continue to backoff and may never efficiently transmit their data causing higher latency in dense networks. Also in this vein, CSMA/CA does not provide any interference protection because the cascading backoff effect can also be used by an attacker to cause network disruption by continually interfering with the channel.

Another well known modulation technique called CDMA has not been typically considered for IoT due to its receiver complexity and energy consumption. Current research has shown that CDMA may be effective in the IoT realm in regards to HOPS communication techniques [134]. HOPS combines the low probability of detection/interception (LPD/I) of direct sequence spread spectrum (DSSS) and a pseudorandom variability of physical (PHY) layer signal controls such as code-, time-, frequency hopping with the ability to operate in dense IoT environments consisting of lower throughput and duty cycles associated with many sensors. HOPS shows promising results adapting military-style PHY layer security with the flexibility and low computational performance required in IoT.

Another important area of IoT communication protocol considerations is the reduction in the total number of bits needed for a single transmission. Reducing the number of bits reduces the amount of energy expended per transmission. Therefore, developing a low-power MAC layer solution for IoT is an important aspect for not only a device, but a network as a whole [151]. Combining the appropriate modulation scheme with a reduced MAC shall create a more efficient and secure IIoT network.

2.4.14 TRANSEC

Transmission security is PHY layer security of transmitted signals. Many different methods have been employed to achieve this security. Such work as presented in [219] relies on TDMA multi-user diversity based on channel state information (CSI). Another method described in [74] uses the random positions of sub-carriers in orthogonal frequency division multiplexing. Other approaches rely on RF channel characteristics such as multipath or complete RF fingerprinting [180]. These methods mainly focus on using known channel characteristics to change the timing of transmissions. Moreover, it may not be feasible to know the current CSI for low-power devices and the extra computation to continually monitor and adjust adds battery drain, and combined with the vast number of these devices, CSI acquisition may be impractical in an IIoT system. Therefore, a pseudorandom process may be employed to change physical characteristics of the signal components, reducing the probability of an attacker to reverse engineer.

One such method was introduced in [130], where an intentional error was introduced during phase rotation of a spreading sequence. The error helps obfuscate the true phase generated through a PRNG process making it difficult for an outside observer to reverse engineer the PRNG sequence. The amount of the error can vary based on the allowed performance loss of the system. This method is a very low-cost, semi-coherent TRANSEC that can be fully coherent if the receiver is synchronized with the transmitter to fully cancel the phase rotation error.

2.4.15 Data Logging

In order to perform some basic device diagnostics, an event log should be stored on the device. This logging will contain failed and successful boot-up attempts, transmissions, and other important events deemed by the user. This logged data and events can be used for basic anomaly detection by the edge node, and if ample storage space or constant power is available, such as a class 2 or 3 device, information can be transmitted through the network, if requested, for larger network anomaly detection. However, in a more constrained device, storage space may be limited, so logged events may be overwritten if no anomalous behavior has been detected. Non-volatile memory will be reserved for data logging to allow access to the data in case of loss of power. The logged data shall also be able to be accessed by authorized and authenticated devices, such as a network central controller, for additional analysis.

2.5 Summary

This chapter described the expanding nature of IoT and current security concerns at the edge device for use in critical infrastructures. More focus on security needs to be included at the lowest layer of IoT (perception) in order to ensure security through an entire system. However, due to the resource constraints of many of these edge nodes, special attention must be taken to allow longevity of remote devices with proper security. A security level-based proof-of-concept architecture was presented with varying degrees of security features. Standardizing security levels of IoT-CI devices will allow a greater sense of trust amongst devices and will ensure that larger networks are more secure because security was designed as a foundation instead of an afterthought. Low-power security primitives were introduced and showed impressive results to provide sufficient security of IoT devices. In order to achieve widespread implementation of these standardized classes of devices, designers and manufacturers must ensure that security is a major focus of all IoT devices, especially those focusing on stringent CI specifications. As IoT devices continue to enter the market at a blistering rate, security research must continue to best utilize the limited resources found on many of these future devices. Many of the current Internet and information security challenges that we face today are from poor implementation (improperly validated) or security as an afterthought (not secure-by-design). As IoT continues to grow, these past examples of improper security cannot be allowed to continue. Therefore, due to the scope and scale of IoT, all products should utilize standardized security features and be properly characterized based on their capabilities. Until there is a consensus for standardizing IoT device security features and sufficient validation and characterization by appropriate organizations such as

NIST, Cicso, etc., products with unknown security will continue to flood the IoT-sphere and continue to introduce great risks to everyone.

Chapter 3

Authentication using Neural Network-Based Specific Emitter Identification

Moving beyond the challenge of standardization, trust and authentication present the next major concern due to the massive scale of unmonitored nodes. The size and flexibility of IoT brings other unique security concerns even when traditional security schemes are properly implemented. For example, the scalability of IoT increases the possibility for remote device or data observation, allowing an attacker to collect vast amounts of information to help in reverse engineering of security protocols [218]. The use of authentication techniques in such a network ensures that only trusted devices are able to communicate on a network. However, most authentication protocols currently used only perform non-physical data inspection (i.e. data content, not physical signal), similar to DES [46], WPA [194], WPA2 [195], or ECC [108], which have been shown to contain latent defects. More specifically, they fail to authenticate the actual device which can lead to possible device compromise or impersonation, as all of the involved parameters can be observed, guessed, or replicated [43]. As the number of small, poorly authenticated, wireless IoT devices continues to grow, common network attacks such as eavesdropping, MITM, DoS, node impersonation, and battery draining [70], [212], [114], will only become more prevalent, if security continues to be neglected. Though, methods have been developed and implemented to perform device authentication, such as physically unclonable features (PUFs), recent work has shown that the deterministic responses of PUFs can be replicated [57]. Therefore, a method by which to authenticate devices through non-deterministic and truly unclonable means is imperative. Additionally, because re-deploying networks of physical IoT devices with improved security systems is likely cost prohibitive in response to latent security vulnerabilities, more robust software-based device authentication techniques, such as radio frequency (RF) fingerprinting, which could be incorporated into the IoT device's layered defenses during production and could be updated remotely, are needed.

In the IoT-realm, low-cost, mass produced devices are often considerably resource-constrained. Consequently, our research aims to pave the way for a low-cost, low-power, and low-latency multi-factor authentication method utilizing neural networks (NNs). More specifically, this work investigates the feasibility of using NNs which cue on the minor, but unique, perturbations of the hardware and PHY layer characteristics on the transmitted signals (RF fingerprints), for multi-factor authentication. When used in addition to other lightweight security schemes, such a solution would provide a robust, layered security approach appropriate for IoT devices [192].

To show feasibility, the use of NN-based SEI algorithms for IoT node authentication is examined using a NN-based SEI algorithm previously developed [203, 204]. NN-based authentication usage protocols for various device and network types are also discussed, and the approach given in [204] is executed on both a resource-rich and more resource-constrained device. The additional memory and computational costs associated with the granularity of such NN-based SEI methods will be examined in the context of the considered approach.

The results shown herein indicate that NN-based SEI algorithms will need further refining before implementation in today's large-scale systems, highlighting the challenges in dealing with NNs in IoT [140]. However, there are numerous benefits to integrating NN-based authentication techniques into the IoT. An approach like that in [204] is well-suited to IoT networks because of the large variation amongst low-cost, mass-produced nodes, caused by relaxed manufacturing tolerances. Such variances result in a highly diverse set of RF fingerprints which are difficult to impersonate based on their analog nature [161]. Using NN-based techniques to help identify and authenticate IoT using their RF fingerprints is particularly promising as such an approach reduces the need for human oversight. More specifically, a NN approach to device authentication can operate on the incoming raw IQ data, eliminating the use of *a priori*-determined, expert-defined features often used. This makes the approach more flexible and robust because feature extraction techniques are often only accurate over a limited parameter range, can be heavily effected by channel conditions, and can be inconsistent. In NN-based approaches, these effects can be far less severe, as they can be trained for. Operating on the raw IQ data stream can also lead to lower latency, as feature extraction can be time consuming,[202, 204] and computation would be reduced for transmitting devices because RF fingerprints are implicit in each transmission. As a result, NN-based SEI shows great promise in the rapidly growing field of IoT security and adds a much needed layer of security to these devices.

3.1 Acknowledgements

This chapter, in part, is a reprint of the material as it appears in the publication:

- J. M. McGinthy, L. J. Wong, and A. J. Michaels. Groundwork for neural network-based specific emitter identification authentication for IoT. *IEEE Internet of Things Journal*, 2019. ISSN 2327-4662. doi: 10.1109/JIOT.2019.2908759.

The dissertation author was the primary author, with Lauren Wong as a secondary author providing the neural network design, and Dr. Alan Michaels supervised the research which forms the basis of this chapter.

3.2 Recent Work

While machine learning (ML) and NN techniques provide the framework to eliminate the need for expert-defined features, recent works in ML and NN-based SEI have continued to depend on these hand-crafted features. Most pertinent to this work, in [133], RF fingerprinting was used with deep learning NNs in cognitive radio networks to identify emitters using *error signals* (the difference between the received signal, which contains emitter specific effects, and the ideal transmitted signal), work by Patel *et al.* [148] created an RF-DNA fingerprint using statistical features (such as the variance and kurtosis, of the instantaneous amplitude, frequency, and phase of the received signal) which were matched to known emitters using ensemble classifiers, and recent work by Chatterjee *et al.* detailed RF-PUFs and their characteristics and used symbol-based NNs for node authentication [57]. However, in both [133] and [148], knowledge of the transmitted signal's modulation scheme or the transmitted signal itself is assumed, and the approaches are unable to identify emitters unseen during training [148]. Additionally, the work in [57] highlighted the impacts of outside factors such as temperature on the approach, indicating RF-PUFs are susceptible to environmental changes. Other work of interest, include [176, 208] and [55]. In these works, pre-defined expert features were used as input to NN or orther ML-based SEI algorithms, such as infinite Gaussian mixture models (IGMMs), to calculate device proximity, perform session key establishment between nodes, distinguish and authenticate physical devices, and detect anomalies at a gateway-level.

3.3 Security Background

Many security practices focus primarily on information security areas of concern: *confidentiality*, *integrity*, and *availability* [181]. However, the IoT networks introduces new issues with additional complexities to security, due to their size and flexibility, as well as the use of low-cost devices. Therefore, device authentication is becoming an increasingly important element of the multi-factor security system, in addition to the current information security practices. Figure 3.1 highlights the Transmission Control Protocol and the Internet Protocol (TCP/IP) network stack and where the following security and communications protocols mainly reside. While current network security protocols such as DTLS [164] and IPsec [174] provide data security, both have been designed with the traditional Internet in mind, so the adaptation to IoT applications is not straightforward. Additionally, neither DTLS nor IPSec are lightweight enough to be executed on IoT devices, and asymmetric encryption's large key sizes, numerous handshakes, and heavy-duty encryption can slow down ideally quick, latency-critical links between nodes [170]. Therefore, more lightweight network protocols and security solutions are becoming prevalent in the literature and implemented in IoT-based networks and devices.

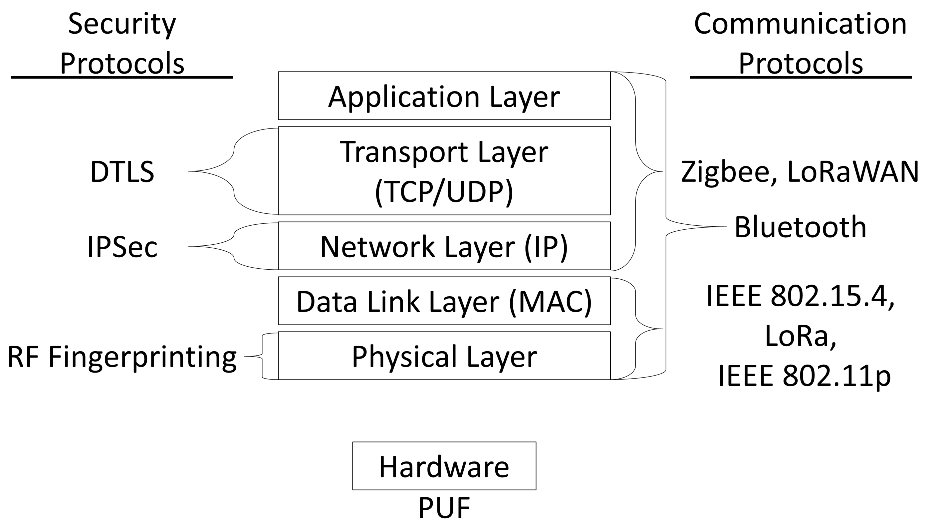

Figure 3.1: A diagram of the TCP/IP network stack highlighting where different current security and communication protocols are mainly performed. Note that PUFs do not fall into the stack, so they are an ad hoc type of security feature.

3.3.1 IoT Security

The communication and security schemes commonly used in IoT systems, described below, are often based on traditional security practices. While many of these security schemes provide data security, they do not provide a reliable method for performing device authentication. Additionally, many of these techniques and security schemes require multiple transmissions between devices and additional computational efforts from both devices to calculate the response to the challenge.

Bluetooth

Bluetooth is a short-range communication protocol that aims to provide low-power links amongst connected devices. Bluetooth relies on an initial pairing of devices with a shared secret value, which may be observed by an adversary, to allow a device to initially join a network. Bluetooth security is based on a combination of device ID, RNG value, personal identification number (PIN), and shared link key to perform authentication and other security features [146]. However, Bluetooth only verifies device authentication based on these values. Therefore, given the ability to watch the pairing process, attackers are able to retrieve and replicate the necessary values, achieving the synchronization necessary to spoof a device. Because the actual user of the device is never authenticated, the network is unable to distinguish one device from another, and the possibility of an attacker gaining access to the network remains. Successful spoofing and man-in-the-middle attacks on Bluetooth devices have been achieved in the past due to these vulnerabilities [91].

Zigbee

Zigbee [13] is a communication protocol standard that is based on the IEEE 802.15.4 wireless personal area network (WPAN) standard [20]. These WPANs allow for the manufacturing and operation of low-cost devices in wireless networks, making this protocol ideal for many IoT systems. Zigbee allows network administrators to broadcast unencrypted network keys to all devices when adding new devices to the network, creating a vulnerability for an attacker to take advantage and attempt to perform device impersonation or MITM attacks. Similar to Bluetooth, the Zigbee protocol performs data encryption and message integrity checks, but the protocol has no ability to perform actual unique device authentication. Moreover, practical attacks against Zigbee have been performed [145, 166], and a Python-based exploitation tool, KillerBee, has been open-sourced [206]. Zigbee also is based on a TDMA CSMA/CA protocol, which can be easily disrupted with timing attacks due to the hidden node problem [191], to inexpensively implement DDoS attacks.

Long Range Wide Area Network (LoRaWAN)

LoRaWAN is a proprietary communication protocol designed for low-cost, low-power devices [5]. LoRaWAN is similar to the previous schemes in attempts to provide security to IoT networks. LoRaWAN utilizes 128-bit cryptographic keys for network formation and data encryption. It provides two methods for devices to join a network: Over-the-Air Activation (OTAA) and Activation by Personalization (ABP) [5]. In [34], a vulnerability in the ABP procedure was exploited through device physical tampering and allowed the security keys to be compromised. Then, the attackers were able to fully impersonate the compromised device and communicate on the network undetected due to the inability of the network to perform actual physical device authentication.

IEEE 802.11p

The IEEE 802.11p standard [15] is the basis for inter-vehicular communication and between cars and infrastructure at the PHY and MAC layers based on the normal WiFi standard. The purpose of this standard is to provide very fast communication capability due to the nature of vehicular movement. The downside of this fast communication is the lack of overall authentication, which must be handled at higher levels. As more smart vehicles and infrastructure become widespread, the use of this standard will grow, requiring fast authentication of devices.

3.3.2 Multi-factor Authentication

Because many of the previously described protocols focus on network formation and data security, true device identity is still a major concern. Therefore, MFA techniques must be utilized to increase trust and security in IoT networks [114, 168]. MFA can generally be achieved by performing at least two of the following methods: *what you possess* (*e.g.* a physical token or ID card), *what you know* (*e.g.* a password, PIN, or cryptographic key), or *who you are* (*e.g.* unique physical characteristics) [168]. In many cases, MFA is layered with the data encrypted through symmetric cryptography [181], which requires both the transmitter and receiver to know the shared encryption key. However, for remote, physically inaccessible nodes, it may not be possible to ensure the device's token will not be tampered with, making *physical possession* the less preferred MFA method. For example, some software packages require a physical key dongle plugged into a computer to authorize use, but this method would not be suitable for a remote device. Therefore, it is assumed that two-factor authentication for resource-constrained IoT nodes will be performed with a combination of *knowledge* (asymmetric or symmetric cryptography) and *physical characteristics*. The following sections highlight some of the current techniques used to perform authentication based on these physical properties.

RF Fingerprinting

Similar to the use of biometrics, such as retina scanners and fingerprints, devices impart their own unique analog modifications to signals, both on the transmitter and receiver ends. Due to manufacturing process variances, even the *same* devices can have minor differences in their local oscillators and I-Q amplitude and phases imbalances. Therefore, identification of devices can be performed by using various signal analysis techniques to observe these differences [57, 133, 173, 192, 202].

One caveat to this approach is the degradation of RF features due to environmental factors such as temperature, pressure, water density, other atmospheric conditions, and component aging [67]. These factors may bound the actual performance of some RF fingerprinting methods. Therefore, special considerations should be made in the design of a network, in order to achieve desired performance. For instance, the number of nodes in the network may be capped due to the additional stored parameters for modeling different environmental cases.

Physically Unclonable Features

PUFs are ideally unique physical features embedded in the silicon components of the chip during the manufacturing process [86]. A PUF is typically a circuit that is designed to give a repeatable response to a stimulus for use in a challenge-response scenario. The actual manufacturing of this circuit introduces or exploits minor imperfections in the manufacturing process producing a unique signature to the actual response used for authentication, much like an intentional RF fingerprint. While PUFs are not a secure communication protocol, they have been implemented to provide physical device authentication. More specifically, a challenge/response scheme is used to determine proper authentication. In this practice, a device receives a challenge command that invokes a response from the PUF based on that specific command. Ideally, the correct response can only be generated by the PUF and not replicated by any other process. As previously discussed in other security applications, if the contents of a device's security scheme are compromised, an attacker is able to impersonate a device. PUFs attempt to solve this problem by providing an additional layer of authentication for devices.

However, though PUFs attempt to be unclonable, research has highlighted vulnerabilities with this technology. For example, it has been shown that some ring oscillator-based PUFs do not exhibit proper randomness for cryptographic functions [214], and ML-based attacks have been successfully been used against 65 nm complementary metal–oxide–semiconductor (CMOS) Arbiter PUFs [96]. Despite their shortcomings, researchers at MIT have done work with PUFs for authentication, and have worked to protect them against ML capabilities to learn the challenge/response pairs (CRPs) [215].

3.4 IoT Model

In order to determine the best use of a NN-based SEI algorithm in IoT networks, assumptions for the network must be defined. The major components of the network are its configuration (size and topology) and its device makeup. These main factors will help bound the parameters needed for proper NN-based SEI in IoT.

3.4.1 Network Configuration

Most IoT networks are heterogeneous in nature, due to the different requirements of the systems involved. Therefore, for certain network configurations, a proper device hierarchy is implemented. This hierarchical design allows for easier control of network traffic because nodes are only able to communicate with other authorized devices. Also, networks with *a priori* knowledge of all devices allows an initial level of trust amongst nodes, given the ability to whitelist allowed connections. This hierarchical design can be used in many different topologies, such as star-of-stars and both sparsely- and densely-connected mesh networks, all of which will be further discussed in the following sections.

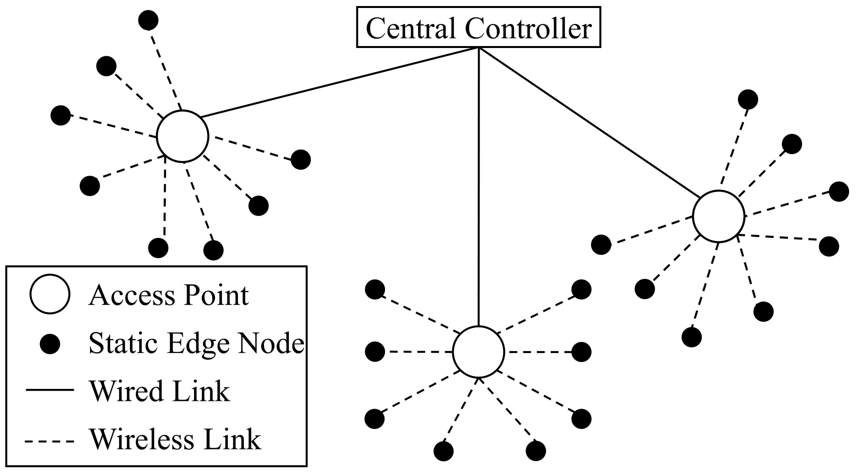

Figure 3.2: Hierarchical star-of-stars network topology.

Star-of-Stars

The star-of-stars (also known as hub-and-spoke) network model, shown in Figure 3.2, is a very simple design. It allows for many edge nodes (ENs) to be connected to a single access point (AP), which is then directly connected to the central controller (CC). The routing is straightforward which eases initial implementation, but also causes inflexibility due to the minimum number of routes. Further, this topology can suffer when APs fail, unless sufficient resiliency considerations are made, such as configuring a backup AP for all nodes. However, this model is still ideal for very resource-constrained ENs that have limited storage capabilities for routing tables as only the main AP and backup AP(s) would need to be stored.

Sparsely-Connected Mesh

The sparsely-connected mesh network is similar to the star-of-stars model in design, but allows for additional routing through specified relay devices. Although this adds more complexity to the routing, there are still hierarchical elements that help with determining known routes, and this added flexibility can help in networks prone to interference. Figure 3.3 shows an example of a sparsely-connected mesh network.

Densely-Connected Mesh

Figure 3.4 depicts an example of a densely-connected mesh network. Densely-connected mesh networks build on the concept of the sparse network, but allow for links between all devices within communication range. This greatly increases the complexity of storing routing

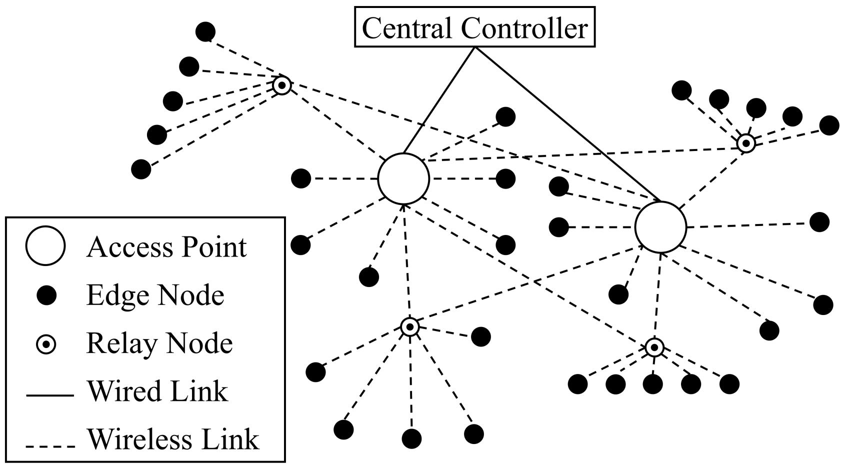

Figure 3.3: Sparse mesh network topology.

information, but also provides the greatest amount of flexibility and resilience. As an added benefit, this type of network structure supports a self-healing capability. More specifically, when a link fails, a new route can be quickly reconfigured, minimizing downtime. Network monitoring also provides the ability to achieve low latency, by periodically calculating the fastest and most reliable routes and via wireless adaptations of time-sensitive networking [209]. However, due to the increase in route management information, this topology may not be well-suited for very resource-constrained nodes depending on the size of the overall network. Densely-connected mesh networks can be compared to mobile ad hoc networks (MANETs) when the nodes are mobile. However, MANETs have even more stringent requirements due to their mobile-based structure and are out of scope for this paper.

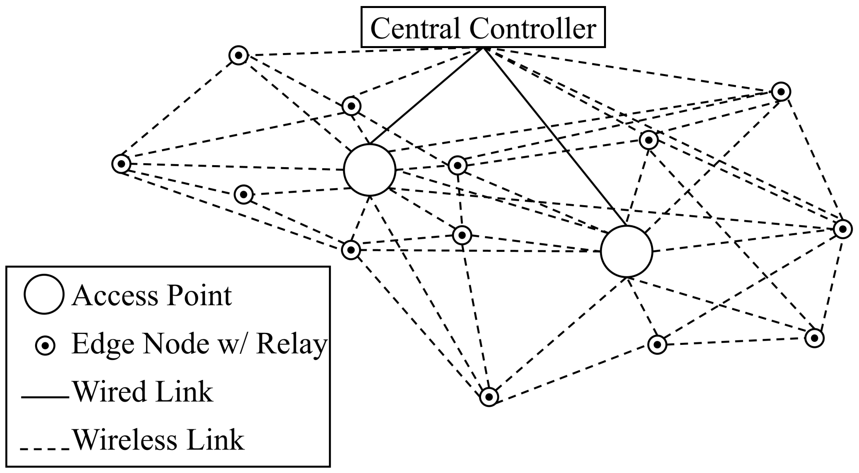

Figure 3.4: Dense mesh network topology.

3.4.2 Devices

Edge Device

The edge device is typically the most variable type of device in an IoT network. They are the major contributor to the heterogeneous nature of these networks. They can range from extremely resource-constrained, battery-powered or possibly energy-harvested sensors, actuators, or controllers to more capable devices with constant power supplies. As an example of a resource-constrained device, the MSP430FR5994 processor has 256 KB non-volatile memory, 8 KB RAM, and a maximum clock rate of 16 MHz [188], which severely limits an SEI implementation on this type of edge node. Therefore, the need to understand the lower bounds of these devices for SEI algorithms must be well understood.

Based on the possible network topologies described above, it may be assumed that a resource-constrained edge node would only need a very limited SEI implementation. For example, in a star-of-stars network, due to the well-defined and small number of links amongst devices, an edge node may only need to perform simple binary authentication; i.e., authentication decisions as to whether or not an incoming transmission belongs to its designated AP. However, as the networks grow more complex, the feasibility of SEI on the outer edge nodes becomes questionable. Because edge nodes only need to identify nearby neighbors that are common links (APs), shrinking the required number of stored parameters should help increase the feasibility of some SEI techniques on less capable devices.

Access Point

An access point in an IoT network will most likely be more capable than its edge nodes, and is typically more resource-rich with clock rates $\geq$ 1 GHz, RAM capacities of $\geq$ 256 MB, memory storage $\geq$ 1 GB, as well as a constant power supply. One example of this type of device is the Cisco wireless gateway for LoRaWAN, which boasts a 1.33 GHz processor, 1 GB of RAM, and 4 GB of flash memory [10]. AP devices may be required to identify upwards of 100 devices on their subnet or combination of subnets (if used as a back-up). However, as more IoT gateways continue to hit the market with specifications similar to the Cisco wireless gateway, the realization of NN-based SEI implementation on these devices becomes very feasible, without significant network degradation or latency.

Central Controller

The CC is usually the master device of the entire network. The CC may also act as a repository for all of the device's SEI parameters (properly safeguarded) that can then be securely transferred to new devices to achieve a type of transfer learning for the network. The CC can be a traditional desktop computer, a server farm (cloud), or may be composed of application-specific graphics processing units (GPUs), depending on application. Therefore, power, processor, and memory constraints of CCs will not need to impact the SEI design.

Other

As the IoT continues to grow, new device types may play bigger roles. For example, relay-type or other hybrid devices may be needed to improve network performance without the use of an additional gateway in order to lower costs. These devices may be more capable than current edge nodes, but will likely not be as resource-rich as APs. Additionally, these devices may have long dwell times before waking up to performing an expensive operation, which will impact how SEI may be performed through these devices. Due to the unknown nature of these devices, NN-based SEI methods will need to be flexible to allow for implementation on future node/device types.

3.5 SEI Background

3.5.1 Traditional SEI Techniques

SEI, the act of matching a received signal to an emitter, has been performed in various domains, including the IoT [57, 104]. Early SEI techniques were used to determine the make and model of automobile engines using audio emissions and harmonics [60]. RF-based SEI techniques continue to be used in both commercial [97] and military [187] contexts to help in determining friend from foe. SEI or RF fingerprinting is possible because of the unique and unintentional characteristics that an emitter imparts onto each transmission, commonly caused by hardware imperfections and manufacturing inconsistency, known as it's RF fingerprint. Traditional SEI techniques utilize expert-defined features to capture these RF fingerprints in tandem with clustering algorithms used for classification and verification [187].

Expert-defined features are typically either extracted from the transient or steady state portion of the received signal. Features extracted from the transient portion of the received signal often examine the time-domain, frequency-domain, or phase-space [77]. For example, a popular frequency domain feature is the power spectral density [162]. The types of features extracted from the steady state signal vary widely including cyclic features [105], preamble periodicity [216], and wavelet based features [45].

The primary limitation of traditional SEI techniques is the use of these expert-defined features to characterize emitters of interest. Expert-defined features are often only accurate over a limited parameter range, can be heavily effected by channel conditions, and are reliant on accurate and consistent feature measurement [179, 187]. Further, expert-defined features only examine certain aspects of the received signal, and therefore the selection of features that characterize the emitters of interest is critical and can be time consuming. The extraction of features itself can also be extremely computationally expensive. In particular, transient processing typically requires high-speed over-sampling. Raw IQ based NN feature extraction techniques offer the potential for fingerprinting when critically sampled, lowering the computational expense.

3.5.2 NN Approaches to SEI

More recent work in SEI makes use of ML and NN-based techniques, but often continues to use expert-defined features as input to these algorithms, such as ensemble classifiers and neural networks, at the classification step [123, 133, 148]. While these approaches often yield good results, the continued reliance on expert-defined features means that the limitations outlined above are still highly relevant. Furthermore, these approaches are often only able to classify a known set of transmitters, failing to identify unknown emitters, and in the worst cases, labeling unknown transmitters as one of the learned classes.

The subset of NN techniques considered in this paper are convolutional NN (CNN)-based, and operate only using the received data, in raw IQ format, as input to the algorithm, relying on the CNN to objectively form a set of machine-learned features that enable accurate decision of the emitter. Two such approaches were developed in prior work [202, 204]. In the first approach, a CNN was trained to estimate the IQ imbalance of the transmitter [203]. By comparing the estimated gain offset to Gaussian distributions fitted to histograms of the CNN output, the estimated gain offset value could be used to identify emitters. Additionally, numerous estimates can be aggregated to produce more accurate results [204]. The second approach used a method called *supervised bootstrapping* to allow the CNN to learn emitter specific features from a known set of emitters. These features were fed to the density-based spatial clustering of applications with noise (DBSCAN) clustering algorithm for classification/identification. It was then shown that these features could be used to describe unknown emitters, allowing for the identification of these emitters using the clustering-based classification approach [202]. Both approaches also consider and handle the case of unknown emitters.

3.5.3 Considered Approach

In the following sections, we examine the use of NN-based SEI algorithms for IoT node authentication using the algorithm presented in [204] and shown in Figure 3.5 as proof of concept. While the protocols described below generalize to any NN-based SEI algorithm that utilize only raw IQ data as input, the approach considered here has several benefits lending itself to implementation on IoT devices. First, the approach is modulation agnostic, and it does not depend on a pre-trained classifier, and as a result, the approach has the potential to identify spoofers as well as the ability to whitelist new emitters in real-time. Second, because

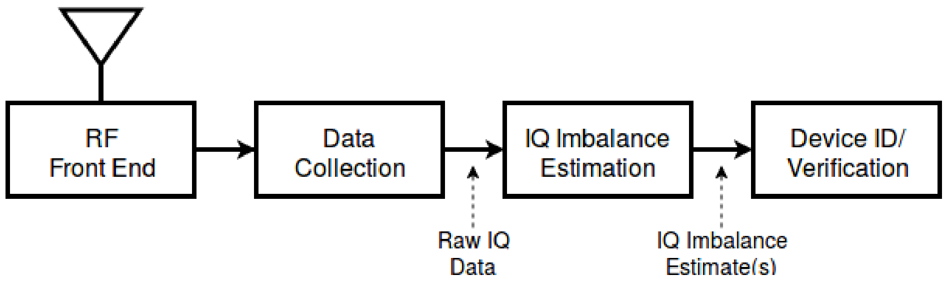

Figure 3.5: The considered CNN-based SEI approach implemented on IoT devices.

the approach uses the inherent feature learning abilities of CNNs rather than hand extracted features, the approach requires only 1024 raw IQ samples, or approximately 2 bytes worth of received data symbols [133], to identify a device. However, given more data, results can be aggregated to produce more accurate identification results. Once the IQ imbalance of the emitter has been estimated using the CNN, the identification and/or verification step is a simple and computationally inexpensive lookup. Finally, because IoT devices are often low-cost, it is likely they will exhibit more severe hardware impairments than high-end emitters, as previously discussed. Therefore, the approach will likely be able to uniquely identify more devices.

As briefly described above, the considered approach is CNN-based, and uses only the received signal, in raw IQ format, as input, which may be obtained using any blind signal detector, such as those found in [154]. Additionally, the algorithm operates without use of test signals, successive iterations, demodulation, or statistical measures, as is often required in traditional IQ imbalance estimation approaches. The considered approach is like the approach proposed in [57] in that both approaches identify emitters using IQ imbalance, a feature of the received signal caused by hardware non-idealities in the transmitter. More specifically, a CNN is used to estimate the gain offset of the transmitter, α, which is then compared to Gaussian distributions fitted to histograms of the CNN output to identify or verify emitters, the details of which can be found in [204].

While the considered approach and that given in [57] both identify emitters using their IQ imbalance, the considered approach uses only the raw IQ samples as input to the CNN, while the approach used in [57] is symbol-based, and therefore requires additional processing (i.e. synchronization) to obtain said symbols and requires significantly more data. Further, in the considered approach, the CNN is used to estimate the expert-defined feature, whereas in [57], expert-defined features are extracted prior to being fed to the neural network for identification. As a result, the approach in [57] is only able to identify prior white-listed nodes, and unknown and potentially harmful devices are falsely identified as known nodes.

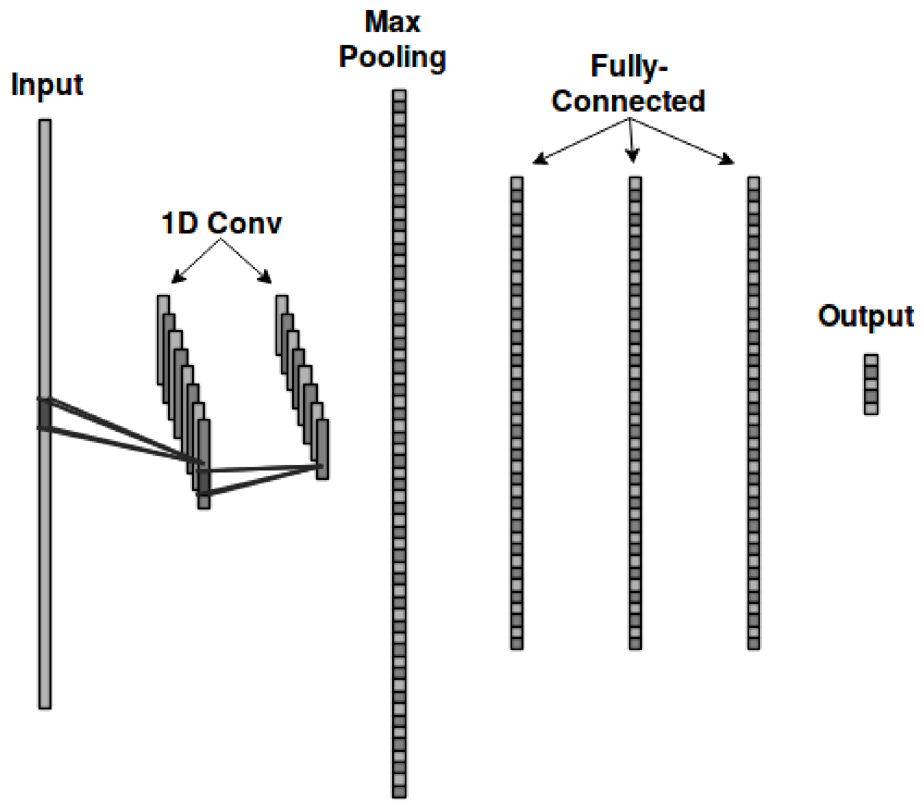

Figure 3.6: The CNN model used for IQ imbalance estimation in the considered CNN-based SEI approach.

Neural Network Design

The CNN used to estimate α, shown in Figure 3.6 and contains two 1D convolutional layers, a single max pooling layer, followed by four fully-connected layers. The final fully-connected 'layer' is just a single node which produces the output, a single value between $(-\infty, \infty)$, representing the estimated gain offset of the transmitter. The hyper-parameters of the CNN gain offset estimator are given in Table 3.1. In total, the model contains 2.3 million trainable parameters. It is important to note that this architecture was developed and selected to achieve the high accuracy, without any attention paid to the size and complexity of the network. Therefore, the architecture could likely be simplified or altered in a variety of ways to reduce the number of trainable parameters and/or overall memory footprint of the trained model, with little to no performance loss.

Table 3.1: CNN IQ Imbalance Estimator Parameters

Layer	Size	Activation Fn.
Input	1024 IQ samples	-
Conv1D 1	12 filters of length 15	ReLu
Conv1D 2	19 filters of length 16	ReLu
MaxPooling	pool size = 2	-
FC 1	282 nodes	ReLu
FC 2	246 nodes	ReLu
FC 3	62 nodes	ReLu
Output	1 nodes	Linear

Performance

The CNN was tested on datasets generated with the open-source *gr-signal_exciter* module in GNURadio [62], which were designed to best model gain and phase imbalances of real systems. The performance of the considered algorithm at various signal to noise ratios (SNRs) is shown in Figure 3.7. More specifically, Figure 3.7 shows the gain offset separation needed between emitters to achieve 80, 90, and 95% probabilities of correct classification. As expected, in lower SNR scenarios, the emitters of interest need to exhibit larger differences in gain offset, in order to maintain a high probability of correct classification. Additionally, as the probability of correct classification required by the system increases, so does the minimum gain offset separation needed between emitters. Further results in [204] also showed potential to improve the performance of this algorithm by narrowing the parameter space of the gain offset estimator network, as well as the ability to aggregate decisions across captures to improve performance. This indicates, for IoT implementation, that each sensor's configurable parameters (such as transmission power and physical placement) may need to be carefully designed and thoroughly tested to ensure that the proper gain offset values are achieved. Also, the trust model can be configured to include multiple tranmissions (independent) to improve the probability of correct classification.

Memory Requirements

In order to execute the proposed algorithm on an IoT device, the device must be capable of holding the weights and biases of the trained CNN in memory. The model used to produce the results shown in Figure 3.7 requires 18.7 MB of memory. This alone makes executing the proposed algorithm on the most resource-constrained edge nodes currently infeasible. However, as previously mentioned, the selection of said model was purely performance-based, and no attention was made to the size of the model during training or testing.

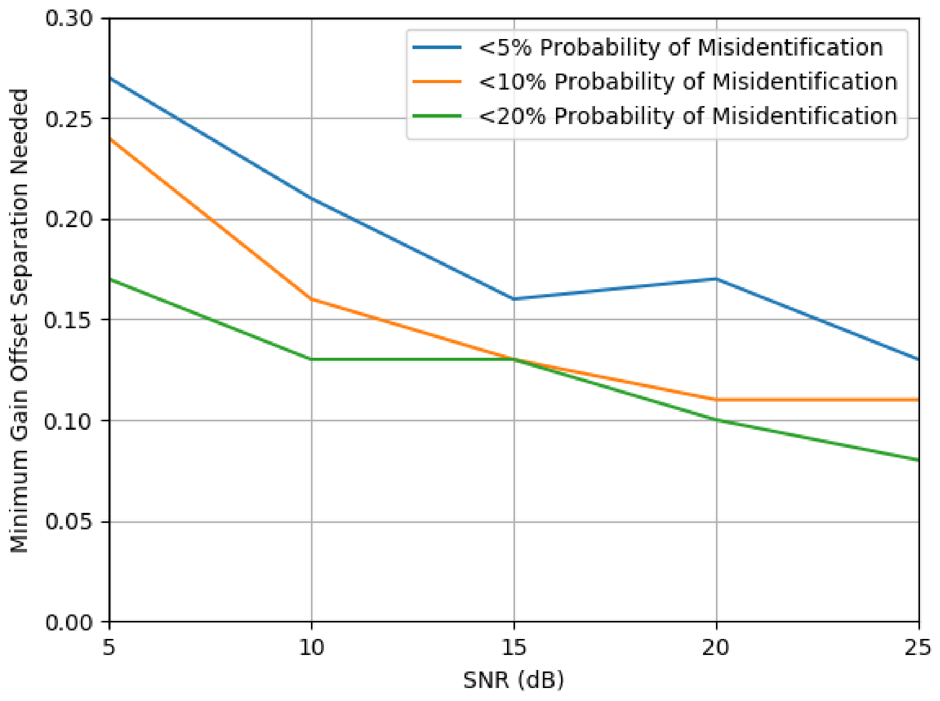

Figure 3.7: Gain imbalance separation needed to achieve different levels of correct classification at different SNRs.

3.6 NN-based SEI for IoT

Although IoT nodes are often resource-constrained devices with very limited functionality, due to their low-cost nature and small footprint, these devices are ideal sensors and actuators that can be placed in remote, not easily accessible locations. Therefore, careful considerations must be made in the overall design of NNs for use on IoT devices. For example, IoT devices may be battery powered with minimal memory, limiting the number of NN trainable parameters that can be stored.

Generally, current NN or other ML applications in IoT are data-driven processes used to help improve overall confidence in and efficiency of existing systems, rather than as a component of the network stack. While there has been work in successfully implementing deep learning network processors on IoT-like devices [40], they are currently limited to tasks such as facial recognition due to restricted memory capacities ($\leq$ 1 MB). As ongoing research continues to improve storage size [52] and overall IoT capabilities evolve, there will be more opportunities to implement NN-based algorithms on these devices.

The setup and usage of the a NN-based SEI algorithm on a network of IoT devices is dependent upon a variety of factors, including network configuration and device/system constraints. However, it may be assumed that all APs, relay nodes, and CCs will be capable of identifying N whitelisted devices and detecting unknown devices. Additionally, it should be noted that N may change dynamically over time as new devices join or leave the network.

As previously mentioned, edge nodes will typically be more resource-constrained than APs, relay devices, or CCs. In star-of-stars or sparsely-connected mesh networks, edge nodes will only need to verify a single AP or relay device. However, in a densely-connected mesh network, edge nodes may also need to be able to identify N devices as well.

As resource constraints are of primary concern when deploying a NN-based SEI algorithm on IoT devices, edge nodes in particular, it should be noted that the algorithm could be utilized in an IoT network in several ways. While running the SEI algorithm continuously would be most secure, it would also require the most power and memory and incur the most latency. Therefore, this is likely to only be feasible on a resource-rich device such as the AP or CC. A more likely use case is to run such an SEI algorithm periodically or on a triggered basis. This will be less secure, yet its frequency of initiation may be considered according to the value of the data being communicated. However, periodic or triggered SEI use is far more power efficient, and will incur far less latency in resource-constrained systems. In the most resource-constrained systems (i.e. edge nodes), it may also be possible to rely on other devices for verification. This is clearly the least secure solution, as the verifying device will be assumed to be a trusted device at all times. However, this risk can be improved by strengthening the timing and distance profile of the verifying node. This solution will also incur the most latency, requiring the node to transmit to another device and await response. Given these trade-offs, three possible protocols for using a NN-based SEI algorithm in an IoT network are described below.

3.6.1 One-Way SEI

Given that it is currently improbable to perform SEI on a resource-constrained edge node, for certain networks, the option exists to only perform one-way authentication of an edge node to an AP. This method assumes accepted risk that an AP is unlikely to be compromised, reducing the need for mutual authentication to be performed by an edge node. This example is illustrated in Figure 3.8. After the AP performs SEI, it may have local policies in place to handle flags raised from unauthenticated devices, or it may relay that information to the central controller to perform further aggregation and decision policies.

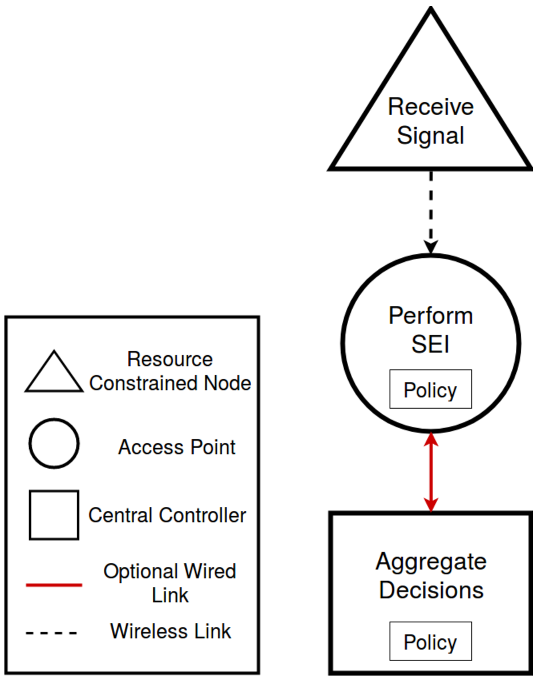

Figure 3.8: One-way SEI approach.

3.6.2 Mutual Authentication Through a Secondary Device

Another approach, allowing mutual authentication for edge nodes, is to perform SEI on a secondary device that is physically located near the edge device, as shown in Figure 3.9. This secondary device would have more capabilities than the typical edge node, such as a Raspberry Pi Zero class device compared to an MSP430 class device, and would perform SEI on-demand (triggered or random) from a nearby edge node. This method allows the edge node to perform its normal operations while off-loading the computationally expensive authentication to a nearby, trusted device. In order to make it infeasible for an attacker to impersonate either of the devices, the distance and/or timing constraint between the edge node and secondary device will need to be well-defined. The benefit of this approach is that the secondary device needs no knowledge of the actual message because only raw IQ samples are needed as input to the proposed NN-based SEI algorithm. Therefore, the secondary device can simply perform the algorithm on the available transmissions from the AP. Additionally, depending on the network structure, a single secondary device could perform SEI for a small number of nearby edge nodes (≤ 5).

3.6.3 Authentication-as-a-Service

A third approach, also permitting mutual authentication is the on-demand (triggered or random) wake-up of a moderately efficient, yet more capable, processor (e.g. ARM11 [8])

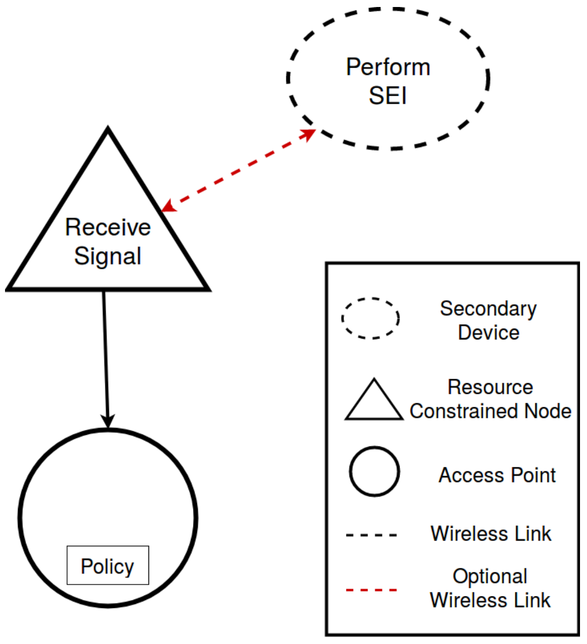

Figure 3.9: Mutual authentication through secondary device approach.

with non-volatile RAM by the primarily used microprocessor (e.g. MSP430). This method results in a relatively small duty cycle for the higher power processor, limiting its overall energy draw to the point of feasibility for an IoT edge node. This model is depicted in Figure 3.10.

3.7 IoT Implementation Results

Once the gain offset of an emitter has been estimated, the identification of said emitter is simply a lookup. Therefore the computational burden is in executing the CNN gain offset estimator. Therefore, this section describes the performance of gain offset estimation CNN portion of the considered approach, implemented in Python using the Keras and Tensorflow libraries [19, 61], on both a resource-rich quad-core Intel CPU [9] and a more resource-constrained Raspberry Pi Zero [18]. The Raspberry Pi Zero is considered due to its low-cost and its ability to still run the CNN in Python. This device still may provide higher computing capabilities than battery-powered IoT devices, but it allows the ability to begin to bound the CNN performance in the IoT realm. Our goal is to move towards smaller SoCs or more resource-constrained battery-powered devices as the CNN is further pruned to work in such devices.

The CNN used in the considered approach and described in Section 3.5.3 requires approximately 7.2 million multiplies and an equivalent number of adds to execute one signal capture containing 1024 samples. On a single thread of an Intel Xeon CPU E5-1620 v4 @3.5GHz

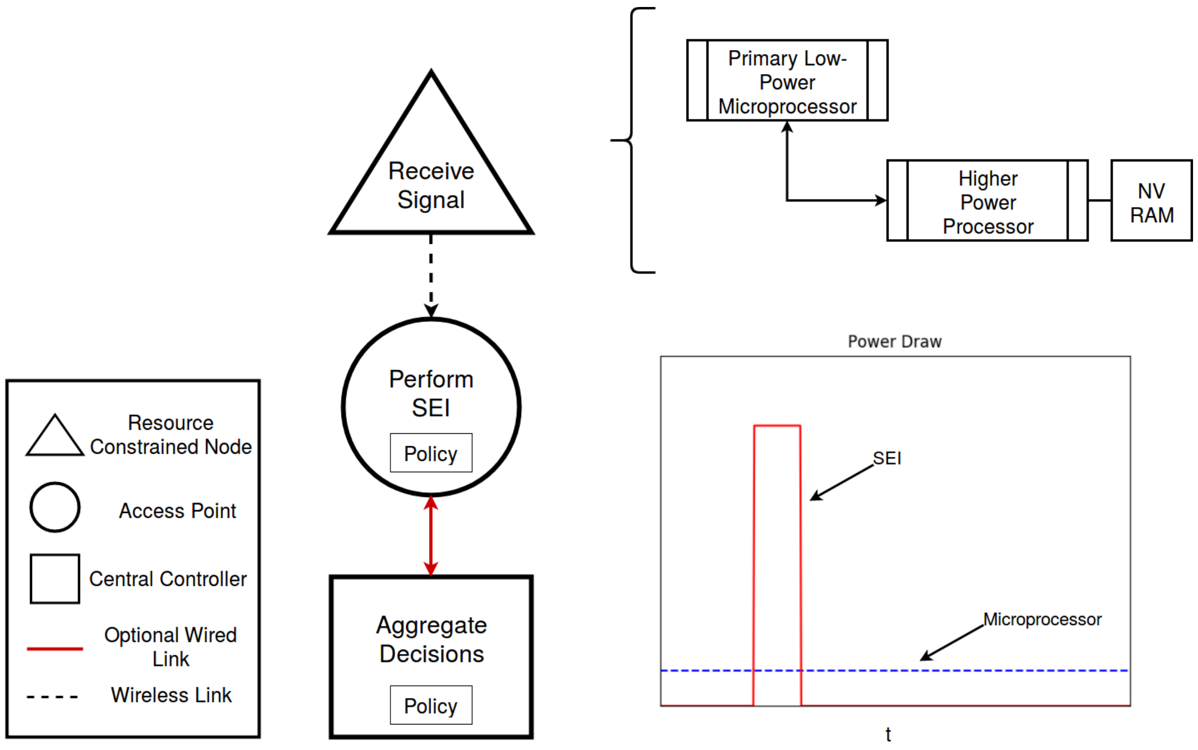

Figure 3.10: The authentication-as-a-service model.

processor [9], this requires 440 microseconds to execute. This highlights the feasibility of executing NN-based SEI algorithms on central controllers or extremely resource-rich APs, with extremely low latency. However, the vast majority of edge nodes will not have this level of capability.

The same CNN model was also executed on a Raspberry Pi Zero [18], representing a more resource-constrained, edge node class of device, housing an ARM11 1 GHz processor with 512 MB of RAM and a removable SD card for non-volatile memory. On this class of device, the current evaluation was focused on the point-to-point link for an edge device's ability to perform SEI as a binary classifier (i.e., if the AP sent the message or not). Figure 3.11 shows a histogram of the execution runtimes of the model on the Pi. The mean execution time was approximately 123 ms with a mode of approximately 108 ms, indicating that this class of device is capable of performing NN-based SEI, with some latency incurred. However, it should be noted that authentication does not need to be performed on every burst, as was previously discussed in Section 3.6. Furthermore, the considered algorithm was not optimized for execution on such a device, and therefore could likely be altered to significantly reduce latency,.

Given these results, it is clear that implementing the considered algorithm on the most resource-constrained devices, such as an MSP430FR5994, is not currently possible, nor is it likely to be possible in the near future. However, the algorithm, as currently implemented,

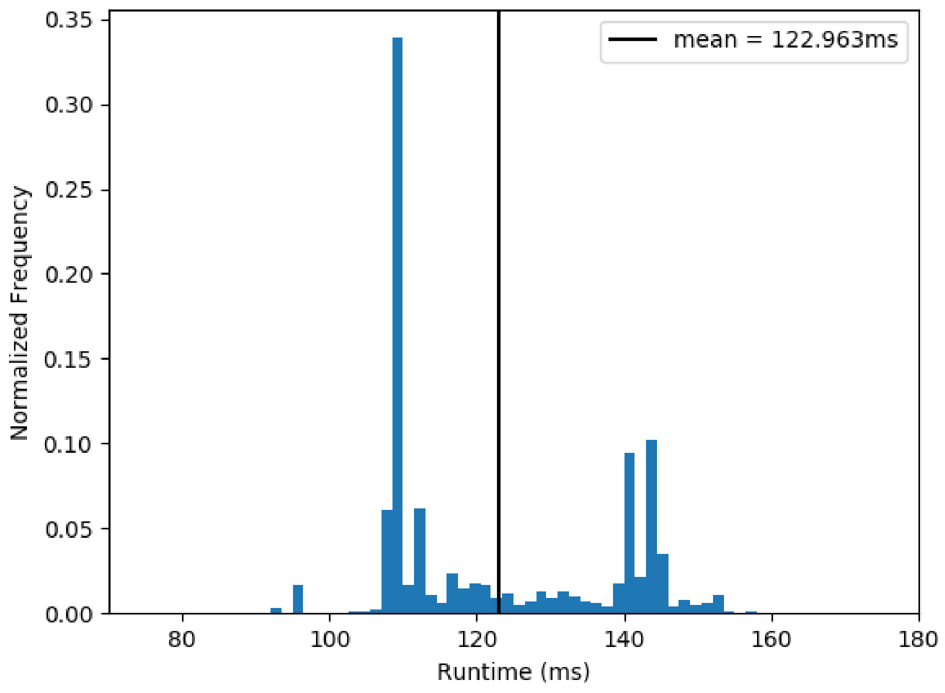

Figure 3.11: Histogram of 10,000 SEI algorithm execution runtimes performed on Raspberry Pi Zero.

could be integrated into an IoT network using one of the proposed protocols given in Section 3.6. Moreover, as IoT device capabilities improve, these results show the promise of using NN-based SEI algorithms, such as the considered approach, in networks containing approximately 100 nodes.

3.8 Summary

Many current applications employ data-centric authentication that can be observed and replicated. Therefore, there is an urgent need to utilize the physical imperfections of low-cost mass-produced IoT devices as a form of physical device identification. This chapter has provided the basis for using a NN-based SEI algorithm as an additional layer of IoT device and network security. More specifically, the challenges of using NN-based SEI algorithms in IoT networks have been discussed, as well as design considerations and existing algorithm improvements to make using such algorithms feasible. This work has shown that an existing NN-based SEI algorithm can be integrated into IoT networks as a SWaP efficient device authentication technique, allowing multi-factor authentication in IoT without human-device interaction, with minimal latency on AP and CC class devices, as well as on edge node class devices, when used on an ad-hoc basis.

Chapter 4

Low-Power PRNG-based Key Derivation Function

The preceding chapters provided solutions for both standardization and authentication challenges in IoT security. The main contribution of this chapter presents an efficient key derivation function for very low-powered devices. Current solutions are more computationally complex which translate to less efficient on these SWaP-constrained nodes. Therefore, this chapter explores the design of a pseudorandom number generator-based key derivation function that is able to produce variable length keys at a reduction in the number of cycles compared to current techniques. The goal of this work is to establish a KDF that will derive time-varying/evolving session keys through a psuedorandom process applied to a large secret key that has already been separately distributed. This approach is well-suited to IoT devices because the pseudorandom process may be extremely low-power, easily implemented, and scalable. This chapter focuses on several methods for mapping the PRNG values from the KDF to the indexes of a secret key to derive a key to be used in symmetric encryption such as the AES [181] or low-power variants using Galois Extension Field techniques [128]. The session key (SK) will be generated from a shared secret parent key (PK) using a low-power KDF. One unique aspect of the proposed stream-based solution is that keys of any (even non-2^k) size may be generated, opening up potential use of efficiently scalable RNS-based approaches. After an overview of KDFs, a comparison to the KDF used in TLS 1.3 is presented. Next, the methodology of the two different PKDF designs is detailed in order to show the simplicity of the design. PKDF simulations are run and results of differing metrics are presented to validate the quality of the key outputs. Following the simulation results, more details and simulations are presented describing the effectiveness of the PKDF based on different PRNGs. One key feature of the PKDF is that it is designed to be PRNG agnostic, allowing flexibility for a node's design parameters. Next, the derivation of a non-binary length key for use in encryption/decryption methods is examined to allow use in the previously mentioned RNS approaches. Finally software and hardware implementations are designed and evaluated for performance metrics.

4.1 Acknowledgements

This chapter, in part, is a reprint of the material as it appears in the publications:

- Jason M. McGinthy and Alan J. Michaels. Session key derivation for low power IoT devices. In *2018 IEEE 4th International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing, (HPSC) and IEEE International Conference on Intelligent Data and Security (IDS)(BIGDATASECURITY/HPSC/IDS)*, pages 194–203, May 2018. doi: 10.1109/BDS/HPSC/IDS18.2018.00050. URL `doi.ieeecomputersociety.org/10.1109/BDS/HPSC/IDS18.2018.00050`. [**BEST PAPER**].
- Jason M. McGinthy and Alan J. Michaels. Further analysis of PRNG-based key derivation function. *IEEE Access* (*submitted June 2019.*)

The dissertation author was the primary author and Dr. Alan Michaels supervised the research which forms the basis of this chapter.

4.2 Background and Related Work

Key derivation is a process in which a key is derived from a secret value or master key [141]. This dissertation defines a *good* KDF as one that produces an output that is indistinguishable from a perfect uniform distribution and does not appreciably lower the entropy of the seed material. The KDF must ensure proper pseudorandomness so that if the attacker gains any knowledge of part of a key, they cannot produce the rest of the key or gain insight into past or future keys. Figure 4.1 illustrates a conceptual process of taking an n-bit PK and performing a permutation to derive a k-bit SK. The blue shows values that were only selected once while the red indicates potential duplication of a value.

One simple method for session key distribution is pre-generating session keys and storing all the keys in memory at the same time as the PK. This approach reduces the computational load of the IoT device because the keys can be generated from a more capable source. However, depending on the needs of the application, the number of keys needed may not fit into the memory on a resource-constrained IoT device. For example, an ATmega328 microcontroller [36] only has 32KB of program flash memory. If all that memory was devoted to key storage, only 2,048 128-bit session keys could be stored. Given that most consumer devices are expected to last for years, this bounded key storage indicates maximum key rollover rates of approximately once a day. Moreover, the publication of all future keys presents a significant risk to network compromise if only a single device is compromised. Not only does this limit the number of keys that can be used, but it would also introduce key distribution concerns when the list needs to be refreshed and yet no mechanism to do so safely.

Outside of traditional KDFs [141], [110], [193], other solutions have been researched for the IoT realm. Simple approaches use LFSR based irreducible polynomials [172, 217]. TPM [7] provides excellent secure storage to help with key management and derivation, but must be

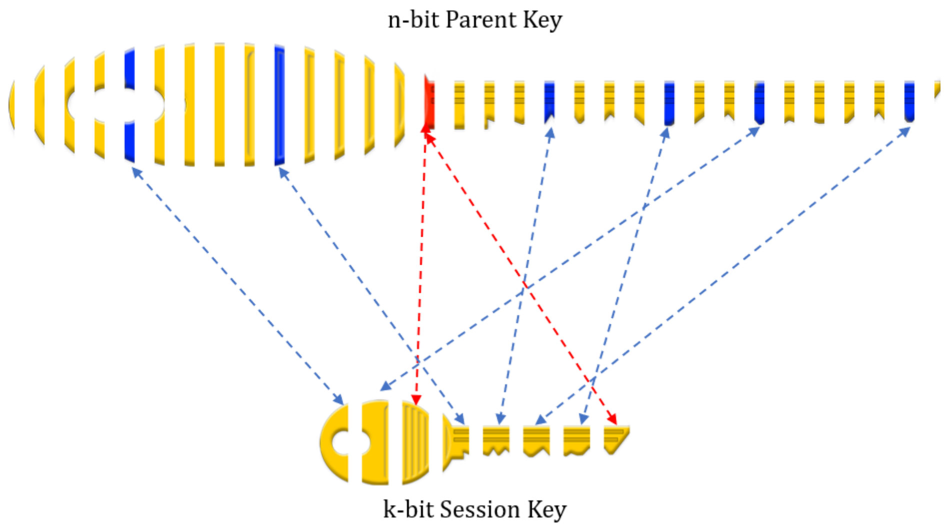

Figure 4.1: Illustration of a random selection process on an n-bit PK to derive a k-bit SK. The blue parts of the PK represent single use values, while the red part of the PK indicates duplication of the value.

present on all devices to be a viable solution. Physically unclonable functions [160, 183], RF fingerprinting [211], and other chip-based [186] or environmental factors may be more complex to implement and use observable information, weakening the source of entropy. Moreover, many of these hardware approaches are limited in their SK generation rates.

Many KDFs used in non-IoT applications are based on good security practices [27, 58]. However, many are password-based [141], and the use of a HMAC [27] [110], although effective in KDFs and more robust, is more computationally complex for low-power devices. Therefore, further research into low-power KDFs must continue. What we desire, and this research seeks to achieve, is a less computationally complex KDF that still yields appreciable security levels.

4.2.1 Comparison to TLS 1.3 HKDF

To support a deeper comparison to the existing state-of-the-art for KDF techniques, we considered comparisons with the hash-based HKDF [110], which is the key derivation function planned for implementation in TLS 1.3 [163]. The HKDF takes three inputs, a secret *input key material* (*IKM*), a *salt*, and an *info* string, where both the *salt* and *info* arguments are optional but can increase security and allow reuse of the *IKM*.

The HKDF is based on the HMAC defined in [111] as

$$HMAC(K, m) = H\Big((K' \oplus opad) \,||\, H((K' \oplus ipad) \,||\, m)\Big),$$

where H is a cryptographic hash function, such as Standard Hash Algorithm (SHA) variants SHA-256 and [69] SHA-3 [72], or BLAKE2 [171], K is a secret key, and m is the message to be authenticated. $K' = K$ if the length of $K <=$ output hash length of H, otherwise $K' = H(K)$. Both *opad* and *ipad* are constant values of 0x36 and 0x5C respectively, and $\oplus$ is the bitwise exclusive or function and $||$ denotes concatenation.

The HKDF is implemented in two phases, `Extract` and `Expand`. The `Extract` phase takes both the *IKM* and *salt* inputs and calculates an HMAC. This is to ensure that any potential weakness in the cryptographic strength of the *IKM* is reduced. It is recommended to pass all *IKM* through the `Expand` phase regardless of strength. The result of this phase is an intermediate key value with a length equal to the hash function output (e.g., 256 bits for SHA-256). The `Extract` only requires one HMAC operation, but it may require an additional hash operation if the *salt* (HMAC key value, K) is greater than the hash digest length.

The output from the `Extract` function is then passed to the `Expand` function with the *info* string. Based on the desired length of the final derived key, this phase performs multiple rounds of the HMAC process until the concatenated outputs produce a properly sized resultant key. Therefore, unless the required key length is equal to the hash function output, extra bits will be produced that are not used (and possibly discarded), causing an efficiency concern. For example, if the HKDF produces keys of length 42 bytes (336 bits) and uses SHA-256, then the HMAC must be calculated twice since $\left\lceil \frac{336}{256} \right\rceil = 2$. The final output from the HKDF is 512 bits long and must be truncated to the correct length.

HKDF Comparison

The HKDF is a well designed function. It offers flexibility to security with the inclusion of multiple input variables, and it is agnostic to the hash function used in the HMAC process. Therefore, its security strength is based solely on the strength of the underlying hash function. Similarly, the proposed KDF is built upon the security of its PRNG, but has an additional layer of protection from the KDF. However, due to the reliance on the HMAC and other design choices, the HKDF is not ideally suited for resource-limited IoT devices. Although hash functions are designed for efficiency in terms of operations, they are still relatively expensive compared to the simple design elements of the proposed KDF. For example, a 128-bit SK requires the HKDF to perform a minimum of 4 hash operations (plus 1 more if length of $K >$ hash digest length). Also, the bit-by-bit key derivation of the proposed KDF allows greater flexibility in terms of speed and energy consumption compared to the larger block sizes of bits generated by the HMAC design and additional truncation operations to generate the correct length for shorter keys.

Table 4.1: Summary of Comparisons of PKDF and HKDF

	PKDF	HKDF
Inputs	Parent Key	Input Key Material
	Window Size	Salt
	PRNG Seed	Info String
Core Function	RNS-Based PRNG	Hash Function
Key Output	Bit-by-Bit	Block

4.3 Key Derivation

For this research effort, we will assume that a new SK of length k bits will be derived from a PK of length n bits ($k \ll n$) that is shared between two devices already stored in each device's memory. The PK will be generated and stored in memory during installation and should never be shared amongst devices except under controlled conditions and with physical access. The newly derived keys may be used for encryption purposes such as AES or any chosen low-power algorithm. The process of deriving shorter keys from a larger key does reduce the number of possible keys, therefore reducing the search space. However, if the PK is *sufficiently* larger than the SK (empirically, $\approx 8\times$ the length), then it will be shown that the search space is bounded either by the entropy of the PK or the seed value controlling the KDF selection of bits for the SK. Moreover, the randomized mixing capability that is capable of selecting elements of the n-bit PK more than once offers the potential of expanding beyond $\binom{n}{k}$ possible SKs, assuming without replacement, or n^k with replacement.

The key derivation process must be deterministic so that two separate devices may derive the same key for encryption and decryption of a transmitted message. Assuming each device uses the same seed for the PRNG, then the same SK will be generated on each device. Further, there is a desire for the key derivation process to support gaps in connectivity (e.g., intermittent access to the network) and targeted key revocation; those key architecture and policy constraints will be addressed in future work.

Figure 4.2 shows a candidate example of the general steps of starting with a PK and deriving a SK. Figure 4.2a represents a 1024-bit PK. Next, the derivation process pseudorandomly selects 128 addresses from the PK to map the bit values to the 128-bit SK shown in Figure 4.2b. Finally after all the addresses are chosen, the final SK is created by mapping the address values from the PK to the SK as shown in Figure 4.2c.

Another consideration for key derivation is whether to allow the algorithm to discard (without-replacement) or reuse previously selected values (with-replacement). The without-replacement method allows the generation of $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ distinct key mappings. In the case of $n = 1024$ and $k = 128$ this yields 2^{552} possible 128-bit permutations. However, since a value is not

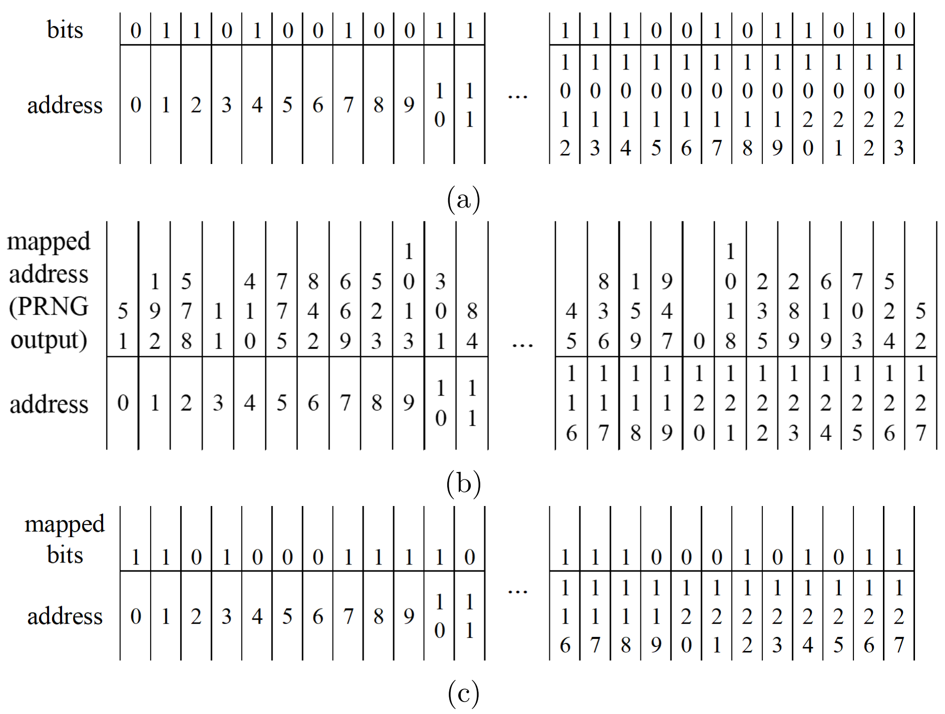

Figure 4.2: (a) shows the bits and address values of the Parent Key. (b) shows the intermediate step of mapping the pseudo-randomly selected address values for the new key. (c) shows the final mapping of the bit values from the selected addresses of the Parent Key to form the derived Session Key.

repeated once selected, the key search space is reduced for each subsequent value. Although there are still a large number of values to choose from, this reduction does cause a practical concern in that the resulting values strongly depend on the underlying PRNG selection process. Implementing this without-replacement method is also more complex in hardware. The saving of previously selected values for comparisons and the actual comparison logic consume precious resources on a constrained IoT device, while eliminating a confounding factor that impairs reverse engineering.

The with-replacement approach allows the chance of value duplication, which in turn enhances protection against standard non-stochastic attacks, such as known- and chosen-plaintext attacks [181]. The possible number of session key mappings that could be created from this method is simply n^k, which for the previous values of n and k yields 2^{1280} possible session key mappings. Based on the birthday paradox [181], which explains the probability of two numbers chosen randomly out of a larger uniform distribution being the same, the with-replacement approach has a 0.9996 probability of the at least one index value being chosen at least twice for a 128-bit key derived from a 1024-bit PK. However, if an algorithm is poorly implemented, there could be a possibility that the with-replacement method may loop through only previously selected values causing a critical security vulnerability. Moving

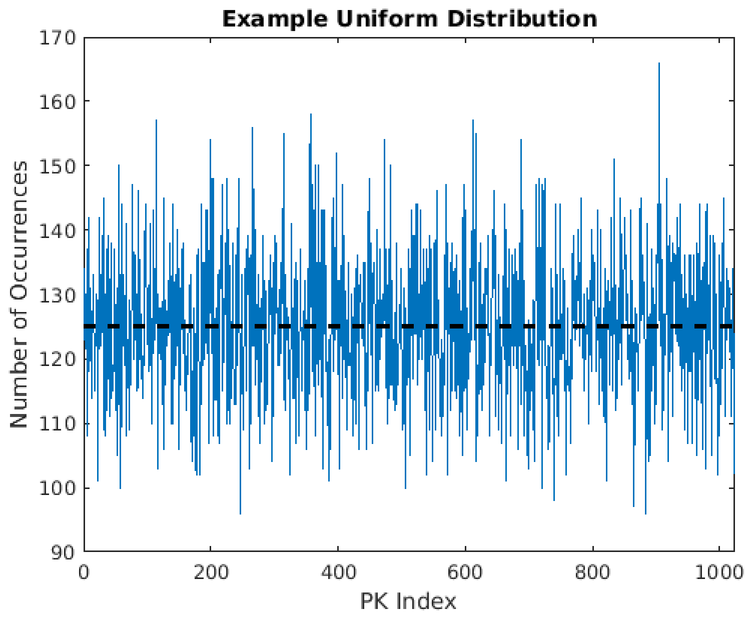

Figure 4.3: An example uniform distribution with replacement of 1000 randomly generated 128-bit long SKs. The blue line represent the number of chosen elements from each PK index for this entire population of SKs. The dashed black line indicates the expected height for each index.

forward, we will assume that all algorithms are properly implemented and will place no restrictions on values selected, welcoming duplications. Despite the exceedingly large number of ways to perform the PK $\Rightarrow \{\text{SK}\}$ mappings, it is worth noting that the maximum entropy of the resulting key stream is bounded by the k-bits of the SK size.

The SK should exhibit properties of a uniform distribution so that it appears random in any observed instances. Figure 4.3 shows an example of a distribution of $N = 1000$ session keys each with $k = 128$ bits derived from a PK with $n = 1024$ bits. The value at each index represents the number of times a PK address was chosen. The expected average number of occurrences of the each column can be calculated as $\frac{k \times N}{n} = \frac{128 \times 1000}{1024} = 125$. In order to achieve this distribution appearance, two techniques were examined.

4.3.1 Selected Techniques

Window Modulus Reduction

The window modulus reduction (WMR) approach begins with a chosen increment or window size, w, to divide into the length of the PK. If w evenly divides into the PK length, there is no sliding of the index values. An illustration of evenly divided windows is shown in Figure

4.4. Each value selected has a probability, $P = (\frac{1}{w})$. However, if w does not evenly divide into the PK length, then the remaining non-integer portion wraps around back to the beginning of the index values. This wrapping is displayed in Figure 4.5. Again, each value has the same probability inside the window, but the values of the windows may wrap around to the beginning of the PK. The motivating factor for this approach is to use a w that is not an integer divisor of the PK length. This spreads the aliasing over more values as the algorithm wraps around the length of the PK multiple times as well as ensuring repeating windows or identical subsets of the PK are minimized.

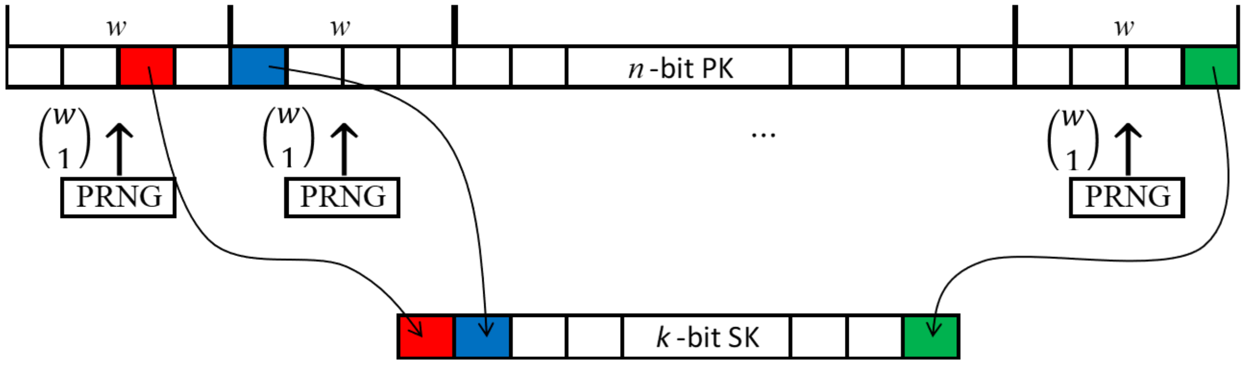

Figure 4.4: Illustration of window modulus reduction approach with w equally dividing into the PK length, n. A PRNG generates a value in the range of w in which a bit is selected with probability, $P(\frac{1}{w})$ in each window.

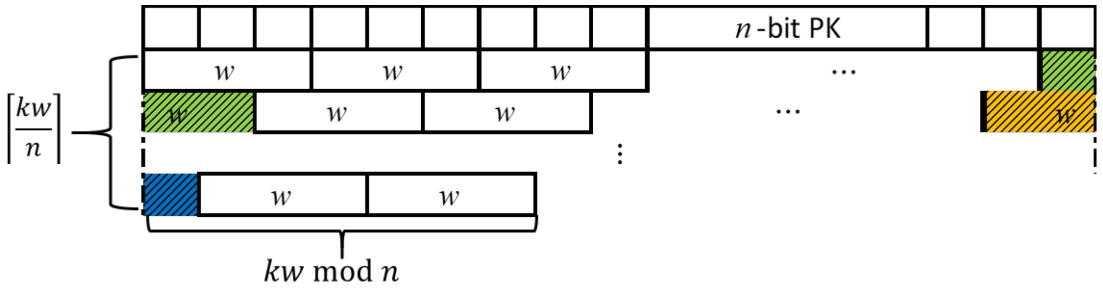

Figure 4.5: Illustration of the window modulus reduction approach when w does not equally divide into n, causing the window values to wrap around the PK and cause aliasing. The shaded windows indicate wrapping around back to the start of the PK.

The window modulus reduction method generates a random value from a large search space, $\approx 2^m$ (where m is based on the size of the PRNG output), and then takes the modulo based on w.[1] After each value is then incremented by w to provide the range of values to continue to the length of the PK. Any value that is larger than the length of the PK is reduced modulo

[1]The choice of PRNG output may be intentionally chosen to force a mixed radix conversion [213],[137] that forces another non-invertible transform within the chain.

the length of the PK, wrapping the values back to the beginning of the PK. The algorithm for this method is described in Algorithm 1.

Algorithm 1 Window Modulus Reduction Algorithm

```
Input: $w, k, n, m, PK$
Output: $SK$
  for $j \leftarrow 1$ to $k$ do
      $RandNum$ = random value in range $[0{:}2^m - 1]$
      $Address = (RandNum \bmod w) + w * (j - 1)$
      if $Address \geq n$ then
          $Address = Address \bmod n$
      end if
      $SK(j) = PK(Address)$
  end for
```

Random Walk

The second method examined was a random walk (RW) algorithm. The RW is similar to the previous method in that it aims to reduce poor distributions, but differs in the fact that the previous index value is used as the next value's initial value plus the random walk bias value instead of using a known step size. The algorithm produces a large ($\approx 2^{16}$) initial value that is then reduced modulo a step size, w. The next value is then produced by taking the previous and adding another value in the range of $[0{:}2^{16} - 1]$ modulo w. As this process continues, once the index values exceed the length of the PK, then they are then reduced modulo the length of the PK. This entire process continues until all the indices have been determined for the SK mapping. The algorithm for this method is shown in Algorithm 2.

Algorithm 2 Random Walk Algorithm

```
Input: $w, k, n, m, PK$
Output: $SK$
  $Address(1)$ = random value in range $[0{:}2^m - 1]$ mod$w$
  for $j \leftarrow 2$ to $k$ do
      $RandNum$ = random value in range $[0{:}2^m - 1]$
      $Address(j) = RandNum \bmod w + Address(j - 1)$
      if $Address(j) > n - 1$ then
          $Address(j) = Address(j) \bmod n$
      end if
      $SK(j) = PK(Address)$
  end for
```

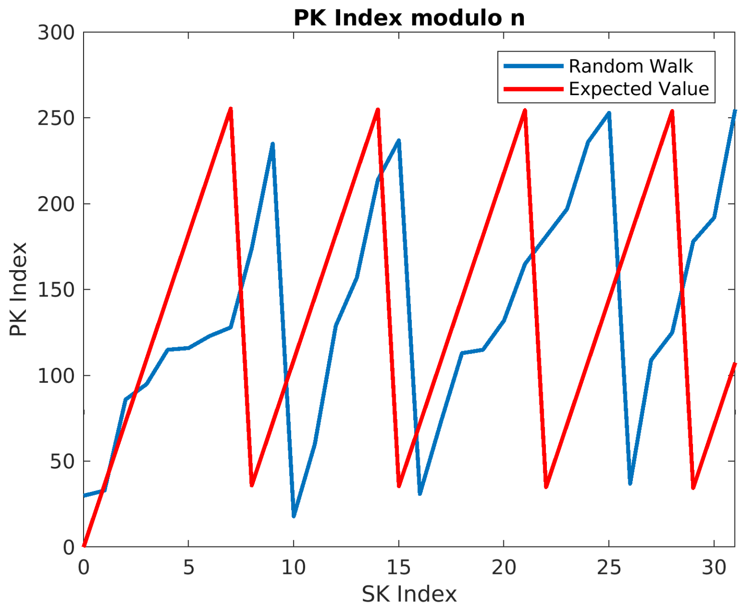

Figure 4.6: PK index values of a 256-bit long PK and 32-bit long SK .The window size was 73.

A comparative evaluation for resulting PK indexes is shown modulo n in Figure 4.6 as well as in an unrolled fashion in Figure 4.7. As shown in both figures, the key stream rapidly desynchronizes with the expected output, increasing the uncertainty of the subsequent index values. As additional SKs are generated, the variance in the unrolled random walk approach begins to exceed that of the PK size n, leading to significant uncertainty on behalf of an attacker attempting to reverse engineer a key stream.

To fully evaluate these proposed methods, special factors must be considered: aliasing, entropy, randomness, and correlation. The following sections will further discuss the impact of each of these factors.

4.3.2 Aliasing

Every possible address value of the PK should have nearly the same probability of selection independent of the number of keys desired for derivation. Aliasing occurs when the range of selectable values is not evenly divisible by the length of the PK. Aliasing, if implemented properly into the derivation function provides additional benefits. By allowing duplication of values, as mentioned earlier, the algorithm will be protected versus standard non-stochastic attacks. However, if this is implemented poorly, aliasing can occur and will cause the distribution to saturate over certain values.

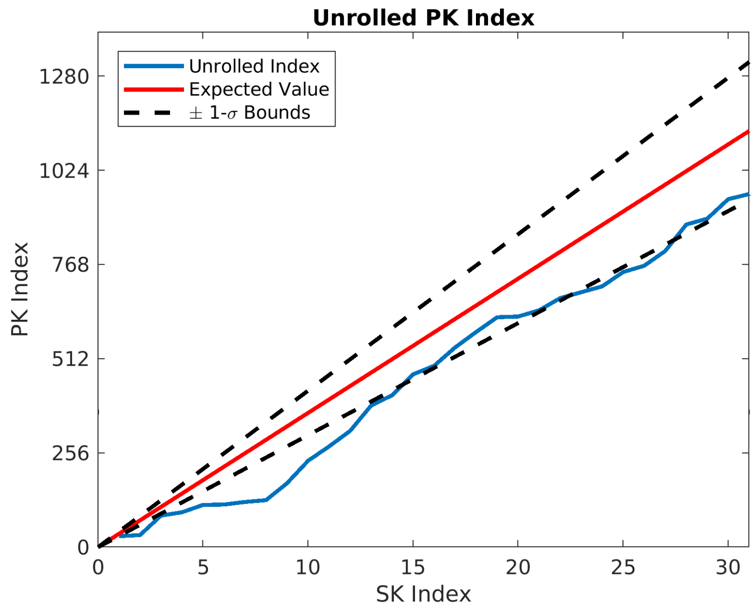

Figure 4.7: Unrolled index values of a 256-bit long PK and 32-bit long SK. The window size was 73.

The distribution effects of aliasing can be calculated using $A = kw \bmod n$, where A is the upper bound of the aliased values. For example, if $k = 128$, $n = 1024$ and $w = 1$, $A = 128$ and since $kw < n$, only the first 128 values of the PK would be used in key derivation. If $w = 2$, then $A = 256$ and again $kw < n$, so only the first 256 values are used. Further, if $kw \geq n$, then all the values of the PK may be chosen, but possibly not at the same probability. This is highlighted in Figure 4.8 for $k = 128$, $w = 10$ and $n = 1024$, because the first 25% of the values have a higher probability of being selected since $A = 256$ and $kw > n$. Poor aliasing may allow an attacker to observe the non-uniform distribution and reverse engineer the KDF more easily compared to a KDF that produces a uniform distribution output. However, if $w = n$, then $A = 0$, indicating that there is uniform distribution amongst all values of the PK for the collection of keys.

The distortion caused by aliasing when multiple keys are generated can also be contributed to the starting index, s, of each key. In Figure 4.8, $s = 0$ for each subsequent key generation, which causes the pronounced stacking of the values. Therefore, not only does setting $s = 0$ for each key cause poor aliasing for a single key, the overall distribution of many keys is affected as well. Different methods for changing the value of s will be discussed further in Section 4.4.

The distortion effects from aliasing can be reduced by allowing the range of selected indices to wrap around the length of the PK many times. The more times the values wrap around, the ratio of the more probable values to the less probable converges towards 1. This is based on the length of the SK, k, and a window size, w (the value that bounds each

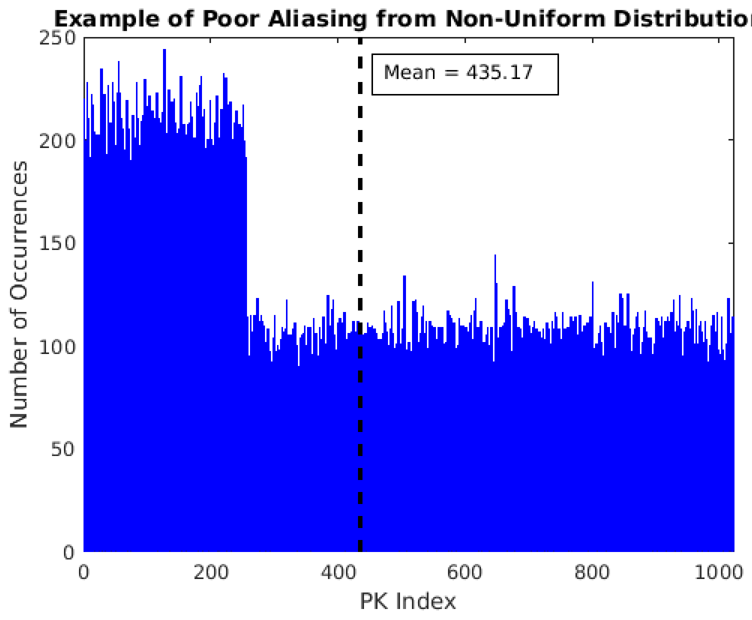

Figure 4.8: Histogram of non-uniform distribution of 1000 randomly generated 128-bit long keys each starting at index $s = 0$. The heights represent the number of times a value was used in a key. The first 25% of the values had a higher probability of being selected.

value of a selected index), as well as the length of the PK, n. For example, if $k = 128$, $w = 11$, and $n = 1024$, then the ratio of most probable to least probable is calculated as $\lceil \frac{kw}{n} \rceil$:$\lfloor \frac{kw}{n} \rfloor$ or $\lceil \frac{128\times11}{1024} \rceil$:$\lfloor \frac{128\times11}{1024} \rfloor$ = 2:1. However, as w increases, the ratio converges towards 1, homogenizing the apparent probability of selecting each bit of the PK. Figure 4.9 shows this convergence as w increases from 1 to n for an example with $k = 128$ and $n = 1024$. It can be noted that the convergence happens quickly, limiting the effects of aliasing based on the window size.

As far as the WMR method, there is a lower bound on w in order for all of the PK values to be in the selection space. This lower bound is defined as $w \geq \lceil \frac{n}{k} \rceil$. Special consideration should be placed on this modulo value in order to have good distribution of values. Figure 4.10 shows the effects of a very small window size chosen for a PK length of 1024 and an SK of length 128-bits. Although the chosen value of $w = 11$ is greater than the lower bound of 8, there is still significant aliasing of the values causing a very non-uniform distribution. This is due to the fact that the range of selectable values only wraps around the PK length once and leaves a remainder of values ($\frac{128\times11}{1024} = 1.375$). However, by choosing a larger value, the effects of aliasing can be reduced. Figure 4.11 shows a better distribution using a $w = 719$ because the values were able to wrap around the length of the PK multiple times ($\frac{128\times719}{1024} = 89.875$) allowing the values to distribute more uniformly.

As with the WMR method, a lower bound on w should be followed to ensure all index values of the PK are selectable for the Random Walk approach. This bound is defined as $w \geq \frac{2\times n}{k}$.

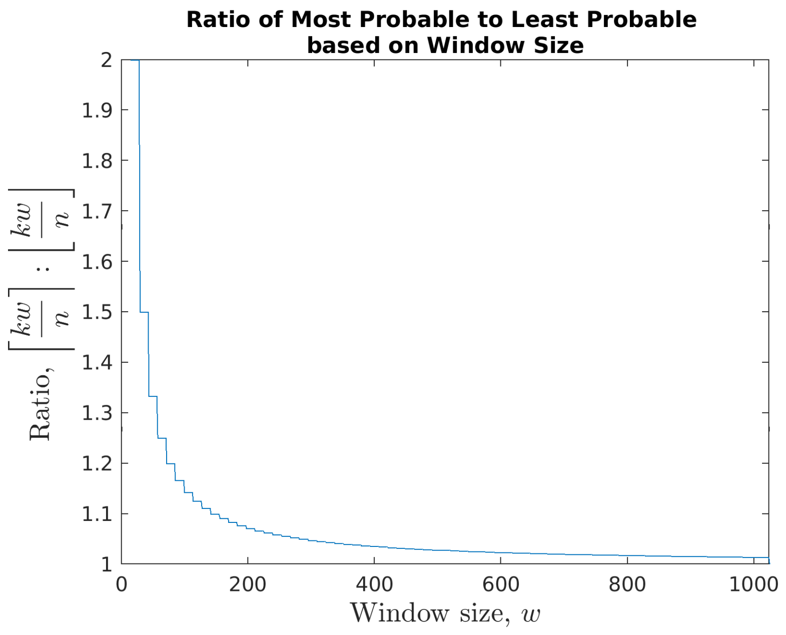

Figure 4.9: Ratios of highest to lowest probability as w increases for $k = 128$ and $n = 1024$.

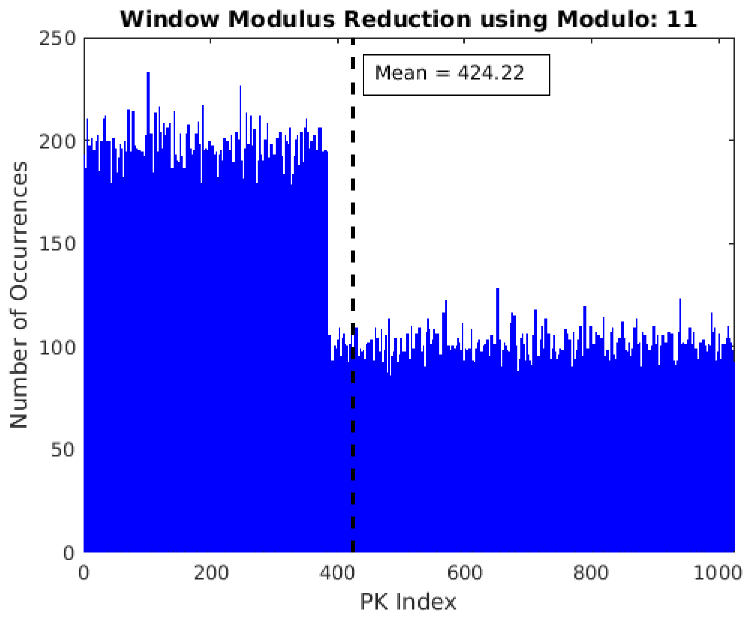

Figure 4.10: Histogram of 1024 Keys generated using the Window Modulus Reduction method with $w = 11$.

Figure 4.12 illustrates the effects of a poorly chosen step size of 11 (lower bound = 16) as not all the index values of the PK have a chance of being selected, reducing the total key

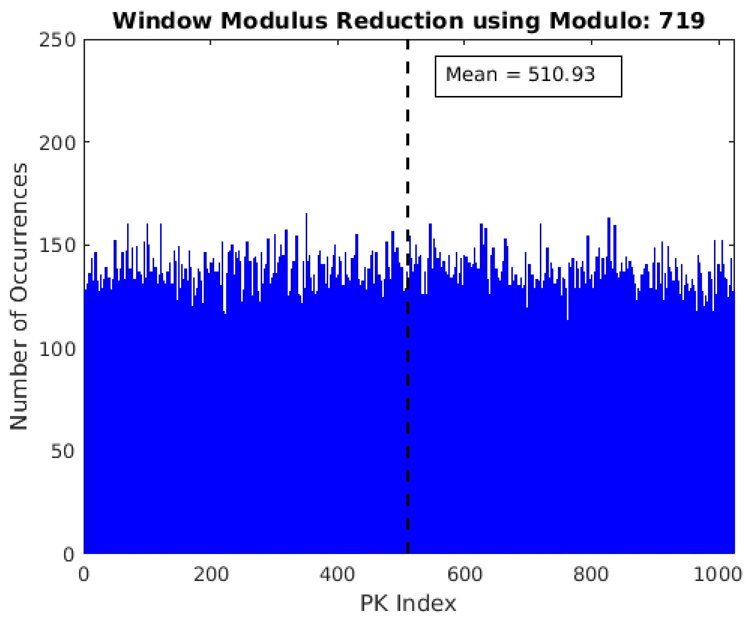

Figure 4.11: Histogram of 1024 Keys generated using the Window Modulus Reduction method with $w = 719$.

space for derivation. A larger step size above the lower bound produces a nearly uniform distribution as shown in Figure 4.13 where the window size is 191.

4.3.3 Entropy

Entropy describes the observed randomness of a system. As Claude Shannon defined entropy in information systems [175], a KDF should have near maximal entropy to ensure the random appearance of its output. Therefore, the entropy of each method will be compared to the maximal entropy as an important metric to determine any adverse impacts of each proposed method.

The following equations show the calculation of the entropy (in bits) for window modulus reduction over a larger key space of n bits, using any window size w. To begin, the total population of n possible ordered bit values in the PK is mapped based on the window size, w. Figure 4.14 illustrates pictorially how 4.1 was derived.

$$n = (n \bmod w)\left\lceil \frac{n}{w} \right\rceil + ((w-n) \bmod w)\left\lfloor \frac{n}{w} \right\rfloor \tag{4.1}$$

Next, by dividing both sides of (4.1) by the total population, it may be treated as a probability distribution.

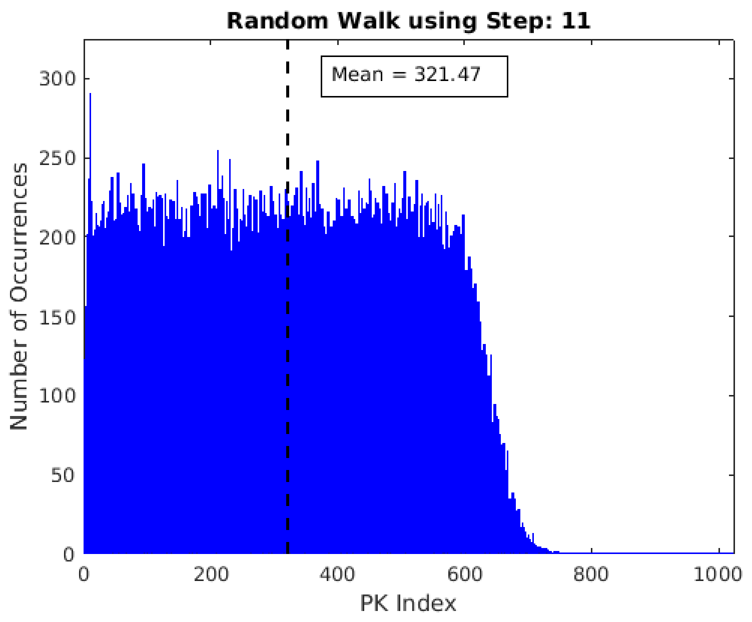

Figure 4.12: Histogram of 1024 Keys generated using the Random Walk method with $w = 11$. Note that due to the small step size, not all of the index values were used in the derivation of keys.

$$1 = (n \bmod w)\frac{\lceil \frac{n}{w} \rceil}{n} + ((-n) \bmod w)\frac{\lfloor \frac{n}{w} \rfloor}{n} \tag{4.2}$$

Finally, the probabilities $\frac{\lceil \frac{n}{w} \rceil}{n}$ and $\frac{\lfloor \frac{n}{w} \rfloor}{n}$ can be rewritten as $\lceil P \rceil$ and $\lfloor P \rfloor$, respectively.

$$1 = (n \bmod w) \lceil P \rceil + (-n \bmod w) \lfloor P \rfloor \tag{4.3}$$

Shannon entropy is defined in (4.4) [66]. This is used to determine the entropy of a single SK based on the values of n and w.

$$H(X) = -\sum_{x \in X} \mathbb{P}(x) \log_2 \mathbb{P}(x) \tag{4.4}$$

where $\mathbb{P}(x)$ is the probability of x being chosen. Applying the probabilities from (4.3) to (4.4), the entropy of a session key can be calculated using (4.5).

$$H = -\Big((n \bmod w) \lceil P \rceil \log_2 \lceil P \rceil + (-n \bmod w) \lfloor P \rfloor \log_2 \lfloor P \rfloor \Big) \tag{4.5}$$

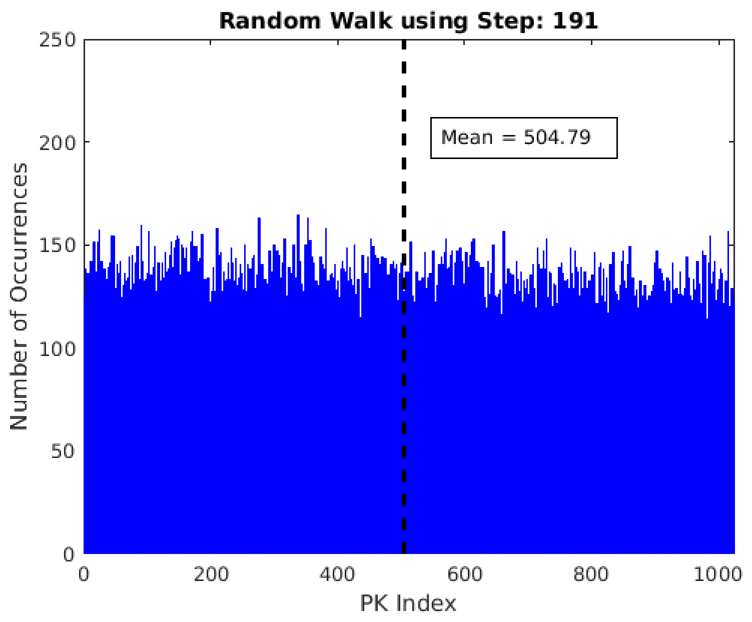

Figure 4.13: Histogram of 1024 Keys generated using the Random Walk method with $w = 191$.

For example, given values of $n = 1024$ and $w = 797$, yield respective probabilities, $\lceil P \rceil$ and $\lfloor P \rfloor$, of 0.00195 and 0.000976. The resulting entropy, H, is then calculated using (4.5) as approximately 9.56 bits.

4.3.4 Randomness

The appearance of randomness in key derivation is critical. The work done in [65] highlights the effects of poor randomness that can stem from low-entropy causing weak key generation. Therefore, it is imperative to design a KDF that produces keys that appear as random as possible. If there are any patterns or other features of a key that make it appear less than random, then an attacker may be able to find a weakness of the KDF or its underlying PRNG. NIST [41] provides guidance on the generation of cryptographic keys that can be extended to key derivation from a master secret key. NIST also provides a test suite for testing the randomness of streams generated by a PRNG [44]. Specific tests from this suite will be used to determine if the SKs derived show random behavior:

- *Frequency*: This tests the key's proportion of 0s and 1s to determine if the sequence has a ratio near $\frac{1}{2}$.
- *Block Frequency*: This is similar to the above test, but tests the frequency of 1s in a M-bit block and compares to the expected value of $\frac{M}{2}$.

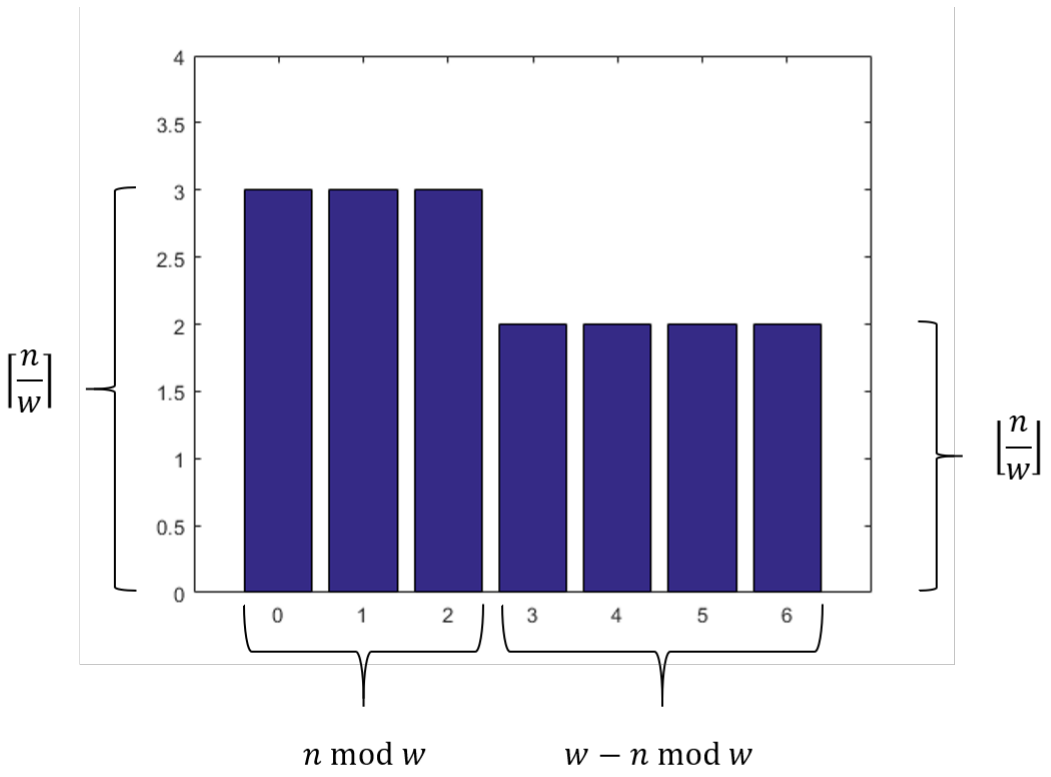

Figure 4.14: Pictorial representation of (4.1). The values of $n = 17$ and $w = 7$ were chosen to show a simple example.

- *Cumulative Sums*: This test determines whether a sequence deviates from the expected outcome of a random walk.
- *Runs*: This tests the number of consecutive sequences of values to determine if the frequency and oscillation mimics a random sequence.
- *Longest Run*: This test compares the longest run of 1s in M blocks to determine consistency with a truly random sequence.
- *Discrete Fourier Transform*: This test detects periodic features using a discrete Fourier transform algorithm.
- *Approximate Entropy*: This test examines the frequency of overlapping, consecutive blocks of data compared to the expected result to a random sequence.
- *Serial*: This test focuses on the frequency of bit patterns across the entire sequence.

For further details on these tests or other tests in the suite, refer to [44].

4.3.5 Correlation

A very important consideration to key derivation is the effect of one key's knowledge leading to enhanced attacks on future keys. If an attacker can gain knowledge of future keys based on any observation, the entire system may be compromised. Therefore, the correlation from one key to another should be minimized. Ideally, a session key should have an equal number of 0s and 1s since it follows a binomial distribution based on (4.6),

$$\mathbb{P}(X = k) = \binom{n}{k} p^k (1-p)^{n-k} \tag{4.6}$$

where n is the length of the PK, k is the number of 1s in the key, and p is the probability of producing k. For a 128-bit SK, the expected mean of 1s produced is $np = 64$. Now comparing one key to another, since both keys should have equal probabilities of producing 1s (0.5), the value of p now becomes the product of both probabilities since for any location in the sequence each key has a 0.5 probability of being a 1. This changes p to $\frac{1}{2} \times \frac{1}{2} = \frac{1}{4}$. Therefore the expected value of 1s common to each key is then $np = 32$.

4.4 Results

4.4.1 Aliasing

To minimize the effects of aliasing for smaller values for a step size illustrated in Figure 4.10, differing the starting index, s, of each key proves very beneficial. Two different methods were examined and both showed excellent results in removing the aliasing and making the distribution appear uniform. The first method incorporated a random starting index for each key. As this method incorporates a new random value for each key and uses up precious clock cycles, a different method using the last index value of the previously generated key was implemented. By only needing to store the previous index, the impact on memory is minimal, but the overall effect produces a uniform distribution. Overall, both methods removed the aliasing and caused uniform distributions for all step sizes.

4.4.2 Entropy

Although the effects of aliasing can be reduced and make the output of any of the approaches appear more uniformly distributed, the actual entropy is not changed based on the step size parameter. Therefore a false sense of good key derivation can be achieved with poor parameters. However, if the key derivation has near maximal entropy, then the changing of the starting index position will be favorable in reducing the effects of aliasing. The window modulus reduction method produced nearly maximal entropy compared to the theoretical maximum with the largest deviation only being 0.086 bits for $w = 740$.

4.4.3 Randomness

The results of randomness from the NIST test suite showed promise for both methods. Each method derived 78,125 128-bit session keys from a PK of 1024-bits with a $w = 719$ to supply the test suite with 10MB of data for each method. For both proposed methods, results are displayed in Table 4.2.

Table 4.2: Results of NIST Test Suite ($k = 128$, $w = 719$)

	Pass Rate	
Test	**WMR**	**RW**
Frequency	99.93%	98.91%
Block Frequency	99.02%	99.01%
Cumulative Sums	98.99%	98.89%
Runs	98.99%	99.07%
Longest Run	98.79%	98.83%
DFT	98.87%	98.87%
Approx. Entropy	98.70%	98.96%
Serial	98.98%	98.95%

NIST recommends a minimum pass rate of 98.7%, indicating that these simulations passed all of the tests. It also appears that there is no apparent difference in the methods as far as the randomness of derived SK streams. Other simulations with differing values of k and w were run and produced similar results. It must be noted that other simulations may vary depending on the strength of PRNG used and strength of PK used. Generally, the length of k should be chosen to meet or exceed the required security strength for the desired key (if not a fixed value), and w should be chosen to have a ratio of 1.1 or less as discussed in Section 4.3.2 and shown in Figure 4.9. Overall, for a low-power IoT device, these results are a very promising for key generation.

4.4.4 Correlation

Another important consideration of key derivation is the relationship of one key to the next. Knowledge of any previous key should not be able to be used to determine a future key. Also, knowledge of a current key should not enable an attacker to determine past keys. Therefore, each session key derived should be independent of any keys past and future. One crude way to measure this independence is calculating the distance between a pair of keys. This is related to the correlation of keys as described in Section 4.3.5. The expected distance between two keys is the number of 0s or 1s common to each key. This distance is calculated as shown in (4.7),

$$d_{i,j} = \left| SK_i \cdot SK_j - \frac{||SK||}{2} \right| \forall i,j \tag{4.7}$$

where $SK_1 \cdot SK_2$ is the dot product of the two keys and $||SK||$ is the length of the key. A histogram of the distance values for 1,024 128-bit keys derived using the WMR method and a $w = 1019$ is shown in Figure 4.15 compared to a binomial distribution. The distance values do tend to follow the binomial distribution centered around the expected mean of 32 as described in Section 4.3.5.

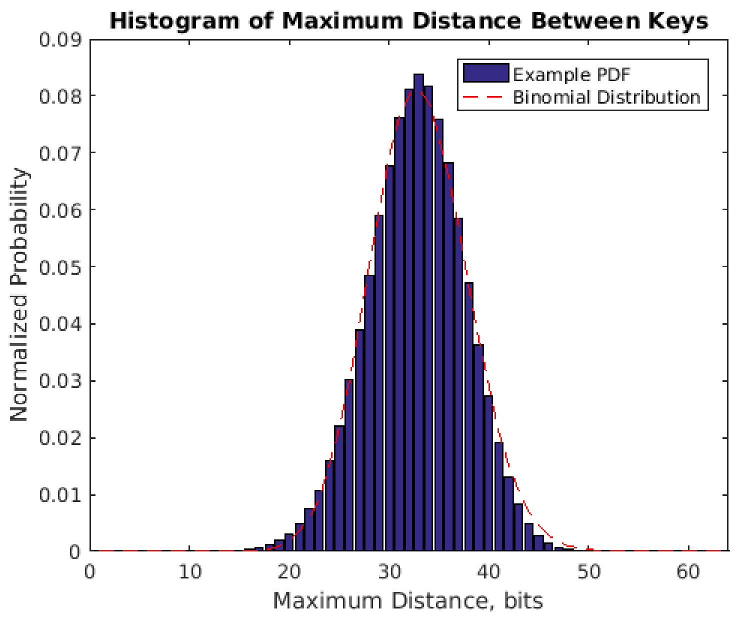

Figure 4.15: Histogram of calculated distance values between 1,024 128-bit keys. Red dashed line shows the expected binomial distribution.

4.5 PRNG-Based Security

Based on the design of the PKDF, the main security concern should be based on the choice of the PRNG. There are many different types of PRNGs that may be implemented in devices. Linear feedback shift registers (LFSRs) are a very basic circuit that are simple, efficient, and relatively fast, which have made them popular as PRNGs, stream-ciphers, and other cryptographic primitives [217], [172], [119]. However, LFSRs are designed to be linear, deterministic, and reliant on strong seeding, creating vulnerabilities that have been compromised in past systems [143],[87]. Another popular PRNG is the Mersenne Twister [122] as it is implemented as a standard PRNG in numerous software packages. It is based on the Mersenne prime $(2^{19937}-1)$, allowing a very long period before it repeats. It is a very fast and efficient PRNG, which contributes to its popularity among numerous software distributions, such as Python and MATLAB. A recent residue number system (RNS)-based PRNG has been developed in [139] to be very efficient and allow for an exceedingly long repetition period. The design allows simple linear scaling to produce exponential increases in period length, making it an ideal candidate for resource-constrained devices.

4.5.1 NIST Statistical Test Suite

These different PRNG stream outputs were then tested for randomness with the NIST test suite. Two different length LFSRs were evaluated to demonstrate that shorter LFSRs do not generally provide good randomness compared to longer ones. These results are shown in Table 4.3. As expected the 15-tap LFSR failed many of the tests (highlighted in red) indicating its lack of security as a PRNG. The Mersenne Twister also did not meet the minimum recommended pass rate for one of the tests (highlighted in yellow), but it was very close (98.65% vs. 98.7%).

Table 4.3: PRNG Stream Output Test Results

Test	15-Tap LFSR	63-Tap LFSR	Mersenne Twister	RNS-Based [139]
Frequency	100%	99.16%	99.08%	99.17%
Block Frequency	96.02%	98.81%	99.01%	99.08%
Cumulative Sums	99.79%	99.20%	99.14%	99.19%
Runs	99.86%	98.98%	98.99%	98.89%
Longest Run	94.59%	99.09%	99.13%	98.93%
DFT	83.70%	98.92%	98.97%	99.16%
Approx. Entropy	69.95%	98.79%	98.65%	98.98%
Serial	85.27%	98.92%	99.15%	99.07%

*Minimum NIST Pass Rate is 98.7% for each test.

In order to test the effectiveness of the PKDF, the same PRNGs were then used to produce session key streams. A random 1,024-bit MK was used to produce a stream of 80,000 128-bit SKs with each PRNG (providing over 10 million bits for statistical testing per PRNG). The generated bits were then tested with the NIST test suite for randomness. These results are provided in Table 4.4. The PKDF successfully masked the poor results provided from the weaker LFSR and all PRNG tests were passed.

After both of these test cases, more extensive NIST statistical tests were run to indicate other patterns and flaws in the output. All PRNGs passed except for the 15-tap LFSR.

Table 4.4: PRNG Test Results with Random Parent Key

Test	**15-Tap LFSR**	**63-Tap LFSR**	**Mersenne Twister**	**RNS-Based** [139]
Frequency	98.92%	98.90%	98.89%	98.91%
Block Frequency	98.80%	99.05%	98.97%	98.92%
Cumulative Sums	98.98%	98.87%	98.92%	98.90%
Runs	99.03%	99.05%	98.92%	99.04%
Longest Run	99.04%	98.92%	98.97%	98.94%
DFT	98.97%	98.87%	98.76%	98.84%
Approx. Entropy	98.78%	98.71%	98.92%	98.83%
Serial	98.86%	98.91%	98.93%	98.96%

*Minimum NIST Pass Rate is 98.7% for each test.

Therefore, based on these results, the PKDF provides sufficient security for all the PRNGs except the 15-tap LFSR.[2] This additional layer of security in addition to the PRNG is based on the window size used and the initial starting index (both internal states to the PKDF) that must be kept secure in the case of a PRNG compromise. However, if a strong PRNG is used, such as a cryptographically secure PRNG (CSPRNG), then the exposure of the PKDF's internal state would still not allow enough information to an attacker to generate the correct derived key.

4.5.2 Law of the Iterated Logarithm

Although the NIST randomness test suite provides a relative level of confidence for the statistics of the output, more testing was performed based on the law of the iterated logarithm (LIL) [198] to show that the random walk variations still appear random in nature. The LIL test evaluates the fluctuations in the variance of the sequences provided from the PKDF. A sequence passes the test if it converges towards a bounds of [-1, 1] and still shows significant fluctuations. All of the PRNGs were tested with tests presented in [199]. Based on these tests, sequences pass if they are below three distance thresholds calculated by the test. As expected, the 15-tap LFSR did fall within the bounds, but it did not pass any of the thresholds because it did not have sufficient variance changes due to its short period. The 63-tap LFSR passed two of the three measurement values, indicating the weakness from an LFSR. The remaining PRNGs (Mersenne Twister and RNS-based) passed all three tests without any apparent concerns. The HKDF was also tested and passed the LIL tests.

[2]A 15-tap LFSR is a relatively weak PRNG due to its short repetition period of 32,767 bits. Therefore, in practical terms of security, this type of PRNG should never be implemented in such a design. The failure of these tests for such a weak PRNG indicates the strength of the overall PKDF.

The LIL testing did show the PKDF is very sensitive to the count of 1s and 0s in a MK. For example, the PKDF passed when a MK was balanced in terms of equal numbers of 1s and 0s, but even with a single bit flip, the LIL test showed that the sequence variance shifted beyond the passable bounds. Figure 4.16 shows the effects of different MK imbalances on the expected variance of the PKDF sequences. The top plot shows that as the imbalances grows larger, the variance is affected more. The bottom plot zooms in and shows that although the 15-tap LFSR stays within the bounds (shown with dashed lines) its variance is not sufficient due to the small period length. The PKDF is shown as the black distribution and falls perfectly inside the bounds with proper variance. Therefore, for the PKDF, a balanced MK must be used, which for a MK of a minimum length of 1024-bits reduces the number of possible MK combinations to $\binom{1024}{512}$ which is approximately 2^{1019} MKs and does not lessen the security strength significantly.

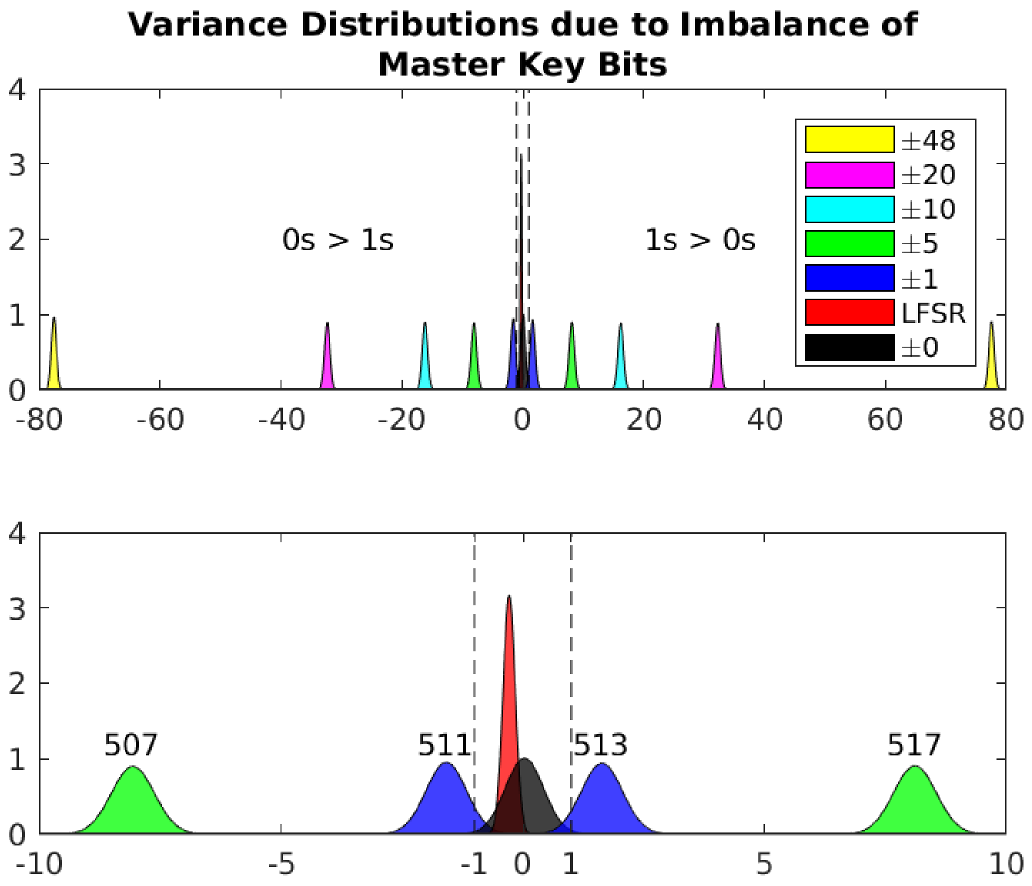

Figure 4.16: Plots of the distributions created by different imbalances in the parent key's 1s and 0s. Also included is a comparison of the 15-tap LFSR that failed both the NIST and LIL testing. The top plot highlights the great effect of an unbalanced MK, and the bottom plot zooms in to show that a properly balanced MK passes the LIL tests.

4.5.3 Joint Entropy

After testing the output of the PKDF, more analysis was needed to determine if any knowledge of bits is passed from one key to the next. This is related to the joint entropy of the two keys. Given two discrete random variables (keys) X and Y, the joint Shannon entropy is defined as:

$$H(X,Y) = -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} P(x,y)\log_2[P(x,y)]$$

where x and y are values (bits) of each key and $P(x,y)$ is the joint probability of the two values [66]. Ideally, the PKDF is designed to create independent keys, therefore joint probability, $P(x,y)$ should be 0. If there is any shared information between consecutive keys, this indicates a possible vulnerability that can leak information about the key derivation process. In order to validate the assumption that keys are independent discrete values, the `XOR` function was used on successive keys to create a new set of keys to be evaluated against the NIST and LIL testing from the previous sections. The new `XOR` keys all passed both the NIST and LIL testing indicating that there is no discernible information shared between keys from the PKDF.

4.6 Non-Standard Length Key Derivation

The derivation of non-standard length keys for use in RNS approaches allows many benefits to the standard binary-length keys used in other encryption techniques such as AES. By allowing variable length keys for encryption, an attacker has a much harder time trying to reverse the process. The following results are from a SK with $k = 73$ using the WMR method of key derivation with $w = 719$ and $n = 1024$. The negative effects of aliasing are minimized with the selection of k to reduce the ratio of most probable selection to least probable as $\lceil \frac{kw}{n} \rceil : \lfloor \frac{kw}{n} \rfloor$ produces a ratio of 1.02. Moreover, the lengths of the session keys generated may be dynamically chosen to support agile PRNG approaches, such as an RNS system with time-varying prime components. The results of the NIST test suite evaluations of tests run using 10MB (136,987 SKs) of data are presented in Table 4.5.

4.7 Software and Hardware Performance Characterization

In order to validate and characterize the performance of the PKDF, software and hardware implementations were evaluated. The software implementation was assessed on an embedded device platform to demonstrate the performance on a typical IoT-like device. It was then compared to the previously discussed HKDF design on the same platform. Next, the PKDF

Table 4.5: Results of NIST Test Suite ($k = 73$, $w = 719$)

Test	**WMR**
Frequency	99.11%
Block Frequency	98.99%
Cumulative Sums	99.17%
Runs	98.93%
Longest Run	99.14%
DFT	98.97%
Approx. Entropy	98.88%
Serial	99.22%

*Minimum NIST Pass Rate is 98.7% for each test.

was designed and implemented on a field-programmable gate array (FPGA). The hardware design's goal was to determine the overall physical footprint size and achievable throughput rates.

4.7.1 Software Implementation

MSP430

In order to accurately compare both the PKDF and HKDF performances, each one was implemented in C on a MSP430FR5994 [188]. This MSP430 device is representative of a candidate IoT edge node with limited 16-bit processing (running at 1 MHz, but maximum of 16 MHz) and memory (256 KB non-volatile, 8 KB RAM) resources. Both implementations were evaluated in total memory size, key derivation time, energy costs, and cycle counts.

The PKDF used the random walk technique with a master key of length $n = 1024$ bits and $w = 809$. A PRNG based on work in [139] was implemented as well. The memory footprint was 2,268 bytes of flash memory (ROM) and 568 bytes of RAM. Energy consumption was measured through the development kit's EnergyTrace software tool [3] by deriving 1,000 128-bit SKs from a 1024-bit MK. This derivation of keys took 52.1 seconds and 103.36 mJ of energy, resulting in an average of 52.1 ms and 103.36 μJ per key.

Next, a 128-bit SK was derived as a baseline for cycle counts. The total number of cycles required to generate a single SK was 652,693. Closer inspection of this result showed that the PRNG setup took 600,394 cycles (primarily for one-time establishment of lookup table values), while the actual derivation of the SK took only 52,211 cycles. The PRNG setup is a required cost to the PKDF, but is only called when RNS primes are changed; re-seeding initial conditions requires less than 0.6% of the total setup cost. Therefore, this setup cost can be amortized over many keys.

The HKDF was implemented using SHA-256 as the hash function due to the ease of finding optimized C libraries for embedded systems. This implementation used only a 22-byte *IKM* with no *salt* or *info* strings.[3] The baseline was performed by deriving a 256-bit SK due to the size of the SHA-256 digest. The memory footprint was 26,202 bytes of flash memory (ROM) and 182 bytes of RAM. Energy consumption was measured by deriving 1,000 256-bit SKs. The derivation of keys took 1,140 seconds and 2,108.8 mJ of energy, resulting in an average of 1.14 s and 2.11 mJ per key.

The overall key derivation required 1,139,389 cycles. Upon further breakdown, it was revealed that the hash function costs an average of 187,790 cycles per call. Based on the design of the HMAC and HKDF, the hash function is overwhelmingly the highest performance cost. Each HMAC call consists of two hash function calls. Therefore, for this baseline, three HMACs were produced, requiring a total of six hash function calls. It is also worth noting that a 256-bit block is the smallest that can be produced for this HKDF, indicating that this is the minimum number of cycles for this implementation. Moreover, due to the fact that a key is produced in multiples of the hash length, there is no amortization of hash function costs over multiple keys.

Five test cases deriving different length SKs were run to compare the performance of each KDF. The first test case derived a key of length 73 bits to highlight the scenario if a shorter (less secure) key may be needed for use in an IoT device. The second and third test case illustrate the creation of standard AES keys. The fourth test case shows the impact of the HKDF's block output design as only 8 more bits are needed for the SK, so the PKDF stays nearly constant to produce the new length, while the HKDF must produce an additional 256 bits and then truncate to achieve the new length. The final test case shows the continued scaling for the PKDF compared to the HKDF step scaling. After deriving 1,000 keys for each scenario, the average number of cycles needed to derive each key (excluding the PRNG initialization) are presented in Figure 4.17. The cost of the PRNG setup is factored in to show the scalability of the PKDF highlighted in Table 4.6, showing the nearly constant cycles per bit of each derived key length. The slight decline also shows how the cost of the PRNG setup is amortized over many keys.

MSP432

After the MSP430 comparisons were completed, the disparity in the performance between the KDFs was quite large. Therefore, a second evaluation was run on a different, slightly more capable device, a MSP432P401R [189], which contains a faster 32-bit Cortex M4 processor (up to 48 MHz) and larger amounts of memory (256 KB flash and 64 KB SRAM). In order to better utilize some of these increased capabilities, some modifications were made

[3]Although these additional strings may be used in real applications, their use adds additional function overhead. Other scenarios were tested with different length strings for each input. As the lengths increased, the total number of cycles increased due to the hash function compressing more data. Therefore, only this scenario was chosen for best comparison performance.

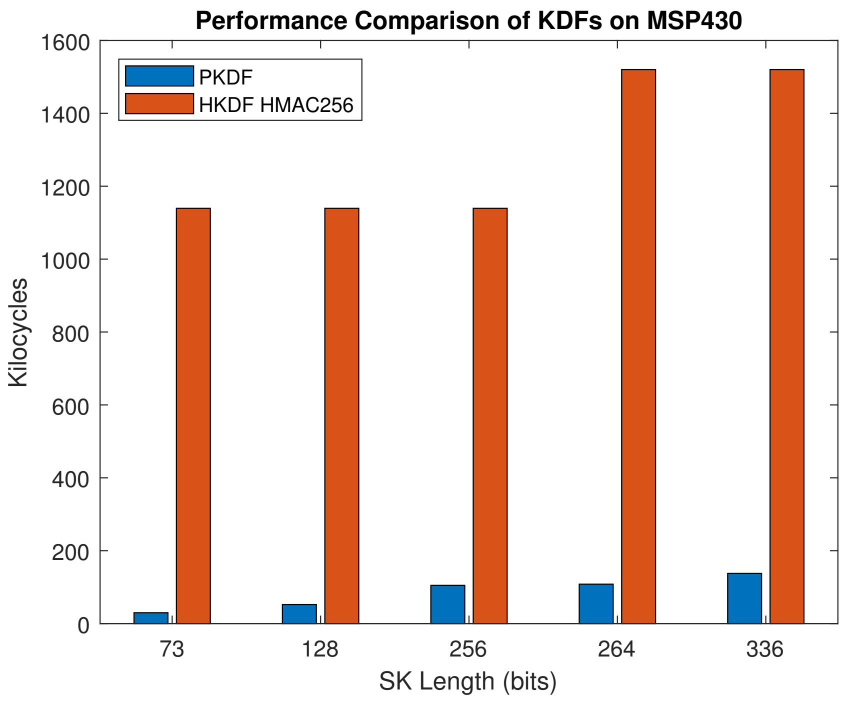

Figure 4.17: Performance comparison of KDFs as implemented on an MSP430FR5994. The SK lengths illustrate the linear scalability of the PKDF.

to the PKDF code to allow it to perform better on the MSP432 device. However, the HKDF implementation did not require any code modifications since it was already based on optimized C libraries.

For the new PKDF implementation, the RNS-based PRNG used more residue numbers due to the increased size of available memory to store the look-up tables. This greatly increased the period to approximately 2.03x10^{23}, yet the actual PRNG cycle cost greatly decreased from that used on the MSP430. Using the new values, the PRNG setup cost is 218,679 cycles, whereas it was previously approximately 600,000 for a smaller period PRNG on the MSP430. Since the PRNG is using more residue values, the total memory cost of the program grew to 16,840 bytes, yet this is still smaller compared to the HKDF implementation which was 23,024 bytes.

After the code was updated, new comparisons were run to determine the performance of both the PKDF and HKDF on the MSP432. First, similar to the MSP430, time and energy evaluations were performed on the MSP432. The PKDF was run using the same parameters ($n = 1,024$, $w = 809$) but the output key length was set to 256 bits to give a better comparison to the HKDF. For 1,000 256-bit SKs, the PKDF took 588 ms and required 17.97 mJ, resulting in an average of 588 μs and 17.97 μJ per key. The HKDF took 694 ms and

Table 4.6: Scalability Comparisons of KDFs on MSP430

	Cycles/bit	
SK Length (bits)	PKDF	HKDF
73	416.0	15,608
128	412.6	8,901.5
256	411.7	4,450.7
264	411.8	5,758.4
336	411.4	4,524.4

required 22.08 mJ of energy in order to produce 1,000 256-bit keys, averaging 694 μs and 22.08 μJ per SK.

Next, in the same manner as the MSP430, to determine a baseline of cycle counts for key derivation, a 128-bit SK was derived using each KDF. The PKDF required 253,956 cycles in which 218,879 of those cycles was for the initialization of the PRNG. Therefore, after this sunk cost, which can be amortized over multiple keys, each 128-bit key only requires 35,077 cycles. The HKDF performance greatly increased on the more capable MSP432 reducing the total number of cycles needed to produce a 128-bit key to 76,764. Again, the hash function is the most computationally complex function of this KDF as it requires 12,213 cycles, and the minimum number of hashes for any size SK $\leq$ 256 bits is six for the HKDF.

The final comparison performed on the MSP432P401R highlights the number of cycles per bit needed to produce different key lengths. These results are presented in Table 4.7. This test shows that the PKDF is nearly constant in the number of cycles per bit with a slight decrease due to the cost of the PRNG initialization being spread over more bits. The HKDF is most efficient at the 256-bit length due to the use of the HMAC256 hash function, but at smaller key lengths, it still is less efficient compared to the PKDF.

The testing on the MSP432P401R overall showed good performance for the PKDF and substantial improvement for the HKDF compared to the evaluations on the less-capable MSP430FR5994. Much of the gains can be attributed to mature, optimized code for the architecture of the MSP432 microcontroller. Future efforts should focus on optimizing the bit-to-bit process of the PKDF to achieve increased performance. The serial nature of software limits the ability to optimize bit-slicing techniques, yet hardware allow for better performance based on the current design.

4.7.2 Hardware Implementations

Software implementations are very flexible due to the ability to update code, but compared to hardware implementations, they are significantly slower. Hardware implementations allow the design to be highly optimized, increasing the performance gain of a function. Therefore,

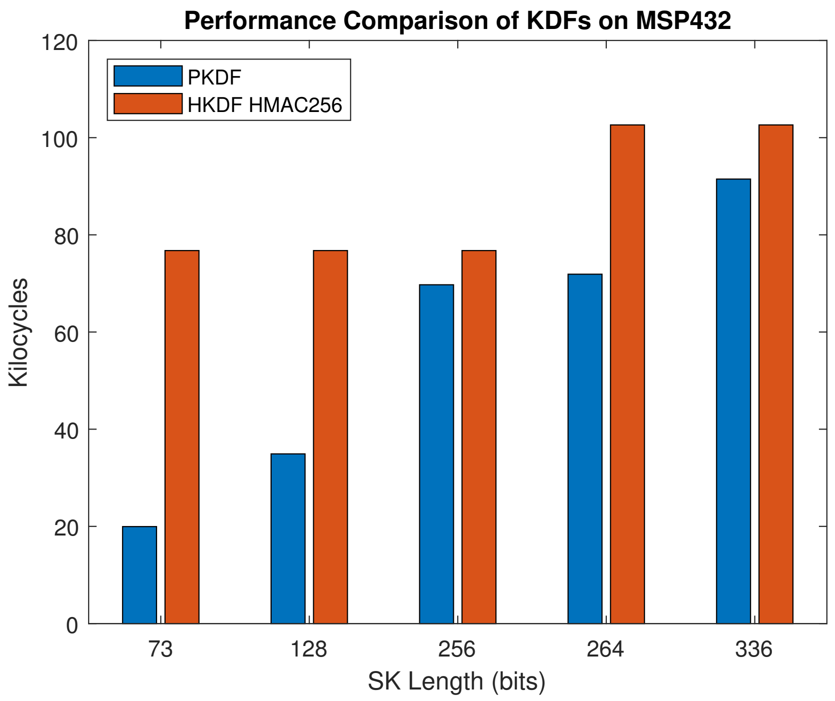

Figure 4.18: Performance comparison of KDFs as implemented on an MSP432P401R. Again, The SK lengths illustrate the linear scalability of the PKDF as was previously shown on the MSP430 comparison figure, Figure 4.17.

the proposed KDF was designed for implementation on a field-programmable gate array (FPGA). The KDF was initially designed in the MATLAB® Simulink® HDL Coder toolchain [121].

First, a 16-bit PRNG value, R, is passed to the function (via a register). In order to perform the modulo operation of this PRNG value, the design leveraged the use of look up tables (LUTs). The 16-bit value is then split into two 8-bit values corresponding to the upper and lower 8 bits of the PRNG value. Each 8-bit number has a 1:1 modulo value based on the modulus value, w. The LUT outputs are then summed together to return the equivalent $R \bmod w$ result.

The $R \bmod w$ result is then added to the previously generated value (based on random walk approach). The output of this summation is set to a size of $\lceil \log_2 n \rceil$ bits, which is equivalent to performing a mod n operation if n is a power of 2. However, if n is not a power of 2, additional memory leakage prevention logic must be incorporated to ensure the range of $[0{:}n-1]$ is never exceeded. After the $R \bmod n$ operation, another LUT call is performed to retrieve the correct bit value of the PK. This bit value is then stored in RAM until the length of the SK is reached. An illustration of this implementation is provided in Figure 4.19.

Table 4.7: Scalability Comparisons of KDFs on MSP432

	Cycles/bit	
SK Length (bits)	PKDF	HKDF
73	276.4	1,051.6
128	274.5	599.7
256	273.2	299.9
264	273.2	388.7
336	272.9	305.4

After the design was tested and correct outputs verified, the design was then ported into hardware description language code in order to be implemented onto the FPGA. Then a core was synthesized for an Intel® Arria 10 10AS066N3F40E2SG FPGA [32]. The resulting core build required only a total of 275 adaptive logic modules (ALMs), 538 total registers, and 5 RAM blocks, resulting in approximately 0.1% overall area utilization of the Arria 10 10AS066N3F40E2SG. Based on the conversion of 1 ALM $\approx$ 2.7 logic elements (LEs) [32], a total of 743 LEs were needed in this design. As FPGA designs are not as efficient as ASIC implementations, there should only be improved performance after ASIC integrations. Therefore, this small area size is believed to make this design well suited for deployment in IoT devices.

The Quartus Prime build also indicated a maximum frequency, f_{max} of 469.92 MHz. Therefore, a throughput value, T, can be calculated based on the following:

$$T = \frac{k * f_{max}}{cycles}, \tag{4.8}$$

where k is the length of the SK and *cycles* is the total number of cycles required to produce a SK. With $k = 128$, the total number of cycles required in this design is $k + 8 = 136$, leading to a total throughput of 442.3 Mbps. Previous work on SHA implementations led to a throughput of 1009 Mbps[125] and 1087.8 Mbps [100] per each SHA-256 hash operation. This correlates to a minimum HKDF throughput of 168.2 Mbps and 181.3 Mbps respectively based on the need of a minimum of six sequential hash operations for HKDF. For a low-power device implementation, this represents an approximately 2.5x multiplier of key generation throughputs.

4.8 Summary

This chapter presented a candidate key derivation function for use in InfoSec functions for IoT devices. Overall, the proposed PKDF shows great promise as a low-power IoT solution compared to current industry techniques. Different design parameters were detailed with

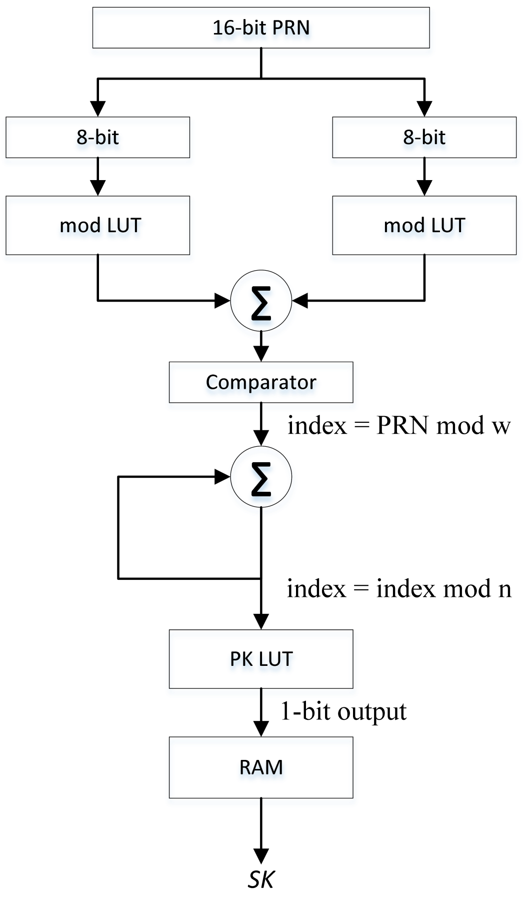

Figure 4.19: Illustration of hardware implementation of KDF.

two different algorithms presented. The simple design of the KDF allows for numerous types of PRNGs to be used based on device availability, while providing an extra layer of security. A non-standard key length derivation was evaluated for use in systems that do not require AES (128/192/256) levels of security. Key output streams passed the NIST tests for randomness, and further PRNG security analysis was also evaluated. Software comparisons of the PKDF and HKDF were performed on two different platforms highlighting the exceptional performance of the PKDF on a more resource-constrained device. Although the PKDF was not optimized for the more capable MSP432 architecture, it still achieved better results compared to the more optimized HKDF. An FPGA hardware implementation was described and preliminary build results show a very small resource utilization and excellent throughput.

Chapter 5

Lightweight Encryption using Galois Extension Field Arithmetic

The next main challenges that this dissertation focuses on are privacy and InfoSec. This chapter explores low-power, scalable encryption solution for data confidentiality. Moreover, maintaining proper confidentiality ensures privacy since an attack is unable to access the data. The previous chapter presented a low-power key derivation function for IoT devices. Those keys are then used in different cryptographic functions, such as symmetric encryption and decryption schemes. One of the highlights of the previous chapter's KDF is the ability to produce variable key lengths, allowing for greater design in cryptographic functions. One such function is detailed in this chapter: Galois extension field (GEF) arithmetic-based encryption and decryption. The basis for this fast, efficient stream cipher is the SWaP constraints from many IoT devices. As some platforms forgo encryption due to lack of resources, this chapter explores a solution using GEF multiplication techniques to provide a varying level of security based on the needs and capabilities of a device. The core extension field techniques are shown to be computationally efficient, scalable, and support a variety of contextual variations that make them suited for improving cryptography on low-power IoT devices. This chapter also explores the use in scalable, multi-party cryptographic algorithms, with a focus on the invertible transforms that may be applied in the extension field to rapidly encrypt and decrypt data. A key benefit of this technique is the ability to exploit the associative property of the underlying arithmetic to interchange the order of parties encrypting or decrypting the data stream without limitation, making the technique highly versatile, meeting a number of IoT use cases. Results show that this technique passes NIST's test suite for randomness indicating that the produced ciphertext is indistinguishable from a randomly generated sequence. Finally, an MSP430 implementation in C indicate that this encryption scheme is faster and more energy efficient than AES.

5.1 Acknowledgements

This chapter, in part, is a reprint of the material as it appears in the publication:

- Jason. M. McGinthy and Alan. J. Michaels. Lightweight internet of things encryption using Galois extension field arithmetic. In *2018 IEEE International Conference*

on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), pages 74–80, July 2018. doi: 10.1109/Cybermatics_2018.2018.00046.

The dissertation author was the primary author and Dr. Alan Michaels supervised the research which forms the basis of this chapter.

5.2 Background

Most cryptographic algorithms, such as AES, DES, and RSA [181], rely on a series of number theoretic transforms that ensure the underlying security of the data encrypted in case of observation by a non-approved third party who should not have access to the information. In general, the greater the protection of a "cryptographically secure" mathematical process, the higher the computational complexity of the algorithm(s). In many contexts, particularly those that are commercial in nature (e.g., low-power IoT) or of limited lifetimes (e.g., weakly encrypted data transmitted via time/frequency/code-based physical-layer hopping mechanisms), the computational expense and associated regulations of using a higher-security cryptographic algorithm may not be warranted. Multiple survey papers [89], [76] have highlighted the challenges and open research areas for overall IoT security, but few have focused on the very low-power, resource-constrained concerns of individual devices.

This research focuses on the implementation of a stream-based cryptographic algorithm using GEF multiplication techniques [136], building upon related PRNG sequence combination [129] and extension techniques. The specific focus of this research effort, and the ensuing cryptographic technique, is the use of invertible elements within the GEF structure to enable rapid encryption and decryption of the signals. This work also focuses on additional number theoretic techniques applied to further generalizations of the GEF techniques, such as those mapping $GF(p^k)$ into $GF(p^{k+d})$, $d \geq 2$ and p prime. Finally, this effort provides concrete examples using the GEF technique that are scalable to multi-party algorithms with interchangeable stream signing.

LFSRs are a simple logic circuit that can be used as a PRNG, stream-cipher, and as other cryptographic components [217], [172], [119]. The simplicity in their design and easy implementation have made them a reasonable choice for IoT security. However, an LFSR is linear, deterministic, and reliant on strong seeding. Therefore, vulnerabilities for LFSR-based designs are a serious concern [143],[87].

Other stream ciphers have implemented LFSRs into their designs. Grain-128 [92] and Grain-128a [28] (updated version of the original) are both stream ciphers based on LFSRs and NLFSRs to generate a key stream that is then XOR'd with the data stream that were part of the eSTREAM cipher competition [165]. Grain-128 was found to be vulnerable to multiple attacks [37], [71], and currently Grain-128a attacks have been possibly discovered [117]. These vulnerabilities highlight the challenges in designing a low-cost, fast stream cipher for IoT applications and systems.

RC4 is a well-known stream cipher introduced by Ron Rivest for RSA Security in 1987 [181] that does not implement a LFSR. It was shown to be very simple, perform very fast, and was implemented into WPA protocol for the IEEE 802.11 wireless LAN standard. However, vulnerabilities [31, 81, 82, 107, 120, 149] were discovered that have questioned the overall security of RC4 and removed it from being used in TLS [153, 155]. This has prompted the research into other stream cipher techniques for low-power, resource-constrained IoT devices. Due to the concerns of LFSR-based designs and RC4, new stream-cipher approaches must be researched.

Another popular stream cipher is AES running in counter mode [181]. AES is the standard in block ciphers, but when run in counter mode acts similarly to a stream cipher. This combines the strong security of AES with typically faster speeds of stream ciphers. However, since AES is a block cipher, it is unable to easily perform encryption on variable sized messages. Therefore, padding is needed to ensure the correct size for messages that are less than the block size. AES is widely used in many applications, but it is a relatively computationally expensive encryption scheme for IoT devices.

5.3 Stream-Cipher Cryptography using Galois Extension Field Multiplication

Extending the combination technique via GEFs from [136], given an input data stream, x_n, and time-synchronized PRNG output, y_n, the Galois field (GF) combination, $\Gamma(x_n, y_n)$, produces a k-bit output z_n. This combination process forms a mapping as such:

$$\Gamma(x_n \in GF(2^k), y_n \in GF(2^k)) \to z_n \in GF(2^k)$$

However, as discussed in [136], this combination process produces a poor distribution across the $GF(2^k)$ elements due to the aliasing of the even elements when reduced modulo 2^k. Therefore, an intermediate mapping from $GF(2^k)$ to $GF(2^{k+1})$ was introduced to allow all values from $GF(2^k)$ to be simply extended onto only the odd elements of $GF(2^{k+1})$.

This mapping process then can be used as a generalized stream combination cryptographic procedure:

$$z_n = \frac{((px_n + c_1) * (py_n + c_2) \bmod p^{k+1} - c_1c_2 \bmod p}{p}$$

where x_n and y_n are input PRNG sequences from $GF(p^k)$, c_1 and c_2 are non-zero rotational constants selected from $GF(p)$, and z_n is the output sequence selected from $GF(p^k)$. One of the most important observations is that all of the mapped elements within the resulting extension field $GF(p^{k+1})$ have a unique multiplicative inverse [167]. As a result, the process of encrypting and decrypting data, or equivalently the process of wrapping and unwrapping a data stream with one or multiple cover sequences may be performed as simply as a second multiplication by the multiplicative inverses of each element in the sequence. This cryptographic process may also be used to combine multiple sequence generators, each representing distinct processes or keys held by distinct parties that do not need to disclose the results to each other.

One limitation of this model for arbitrary p is that the two additive constants c_1, c_2 act as rotational elements on the opposite input(s) from the one to which they are added. Each component to the GEF multiplication is assured to be a non-zero element of $GF(p^{k+1})$, and thus is assured to possess a unique inverse within $GF(p^{k+1})$. When calculating the inverse, it should be noted that the multiplicative inverse selection is relative to $GF(p^{k+1})$, not $GF(p^k)$. These values are relatively easy to calculate via the extended Euclidean algorithm [181], though not in real-time, making memory-based storage of the values (for reasonable k) preferable; note that there exist only p^k possible inverses for a given c value, making this a bijective mapping.

The most common case is that of binary arithmetic ($p = 2$), leading to a simplification of the form

$$z_n = \frac{((2x_n + 1) * (2y_n + 1) \bmod 2^{k+1} - 1}{2} \tag{5.1}$$

in which case the only non-zero rotational elements are the values $c_1 = c_2 = 1$. The identification of the unique subset of $GF(2^{k+1})$ that is used by the injective mapping is also simplified since it is the odd elements, and the multiplicative inverses of all odd elements within $GF(2^{k+1})$ can trivially be seen to also be odd and exhaustive for the odd subset of $GF(2^{k+1})$. When stored in memory, these odd elements may be collapsed on $GF(2^k)$ by subtracting 1 and then dividing by 2. For reasonably small k, ($k < 12$), this processing is acceptable as a brute force storage.

To illustrate this further, consider the single-stage stream cipher cryptographic model shown in Figure 5.1. Taking the user data source as the x_n stream, the PRNG output as the y_n stream, and then the multiplicative inverse operator applied to the PRNG stream as y_n^{-1}, then a simple stream encryption/decryption operation is performed via the commutativity of the arguments within the GEF multiplication structure as follows.

$$x_n = \frac{[(2z_n + 1)(2y_n^{-1} + 1)] \bmod 2^{k+1} - 1}{2} \tag{5.2}$$

Substituting (5.1) in for z_n we get,

$$x_n = \frac{[(2x_n + 1)(2y_n + 1)(2y_n^{-1} + 1)] \bmod 2^{k+1} - 1}{2} \tag{5.3}$$

It should be noted that

$$1 = \frac{((2y_n + 1) * (2y_n^{-1} + 1) \bmod 2^{k+1} - 1}{2},$$

while $1 \neq y_n y_n^{-1} \bmod 2^k$, leading to,

$$\begin{aligned} x_n &= \frac{(2x_n + 1) \bmod 2^{k+1} - 1}{2} \\ &= \frac{2x_n + 1 - 1}{2} \\ &= x_n \end{aligned} \tag{5.4}$$

The simplest such counter-example is the 0 element, which does belong to $GF(2^k)$, but does not have a multiplicative inverse itself. Through the GEF technique, the $y = 0 \in GF(2^k)$ elements maps to $1 \in GF(2^{k+1})$, and thus has a multiplicative inverse of $1 \in GF(2^{k+1})$, returning a $y^{-1} = 0$ inverse element to be placed in storage. For binary scenarios, and all other scenarios where $c_1 = c_2$, the commutativity of the input sequences may be safely assumed. For $c_1 \neq c_2$, commutativity may not be assumed, so the $\{c_1, c_2\}$ values and ordering must be retained, and/or the stored collapse of inverse values from $GF(p^{k+1})$ onto $GF(p^k)$ are dependent on the selection of c.

Expanding beyond this single-stage stream cipher model, consider an expanded binary model that includes a number of R distinct input streams (one of which is chosen to be a user data source) and the continued simplification of $c_u = 1 \forall u$. All inputs $x_n, y_{n,u}$ are assumed to belong to $GF(2^k)$. This scaled version is depicted conceptually in Figure 5.2. The resulting arithmetic process for the ciphertext is:

$$z_n = \frac{[(2x_n + 1) \prod_{u=1}^{R-1} (2y_{n,u} + 1)] \bmod 2^{k+1} - 1}{2} \tag{5.5}$$

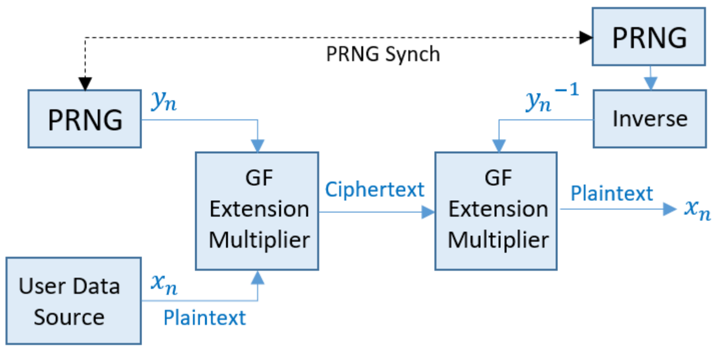

Figure 5.1: Single-stage cryptographic model using a GF extension multiplier.

If each of the distinct inputs streams are developed via independent PRNG processes and/or use different codes, then each and every additional sequence acts as a distinct layer of encryption to the multiplicative composite of the other streams. Further, the pre-calculation and storage costs of the multiplicative inverse elements is amortized over $(R-1)$ encrypting streams, lessening its overall impact to system resources if multiple addresses can be accessed simultaneously. The decryption arithmetic from (5.2)-(5.4) follows a similar process, with multiplicative inverses canceling to return plaintext x_n as shown in (5.6):

$$x_n = \frac{[(2z_n+1)\prod_{u=1}^{R-1}(2y_{n,u}^{-1}+1)] \bmod 2^{k+1}-1}{2} \tag{5.6}$$

The benefit of this proposed approach is that, while the plaintext sequence x_n is effectively wrapped in crypto cover of $(R-1)$ distinct PRNGs, multiple users may uniquely define the underlying process that occurs within their respective PRNGs as well as keep their keys defining any initial conditions private without other users observing anything except PRNG outputs. The decryption of the resulting ciphertext requires all parties to then supply their key and PRNG process again before the original plaintext can be recovered. Assuming the GEF multiplication process is managed by a trusted third-party (e.g., an entity trusted independently by each user), then all parties may have access to the ciphertext and nobody can independently decrypt it until all parties agree to supply their keys. Moreover, no user must disclose the details of their chosen PRNG generation process.

Likewise, this expanded binary multi-user model does not require that all PRNGs be available at the initial source since the GEF multiplication operator may be applied serially (and in any order of inputs). Moreover, since the stream cipher process is more like a sequence masking in many ways, this ability to selectively invert one of the process streams makes

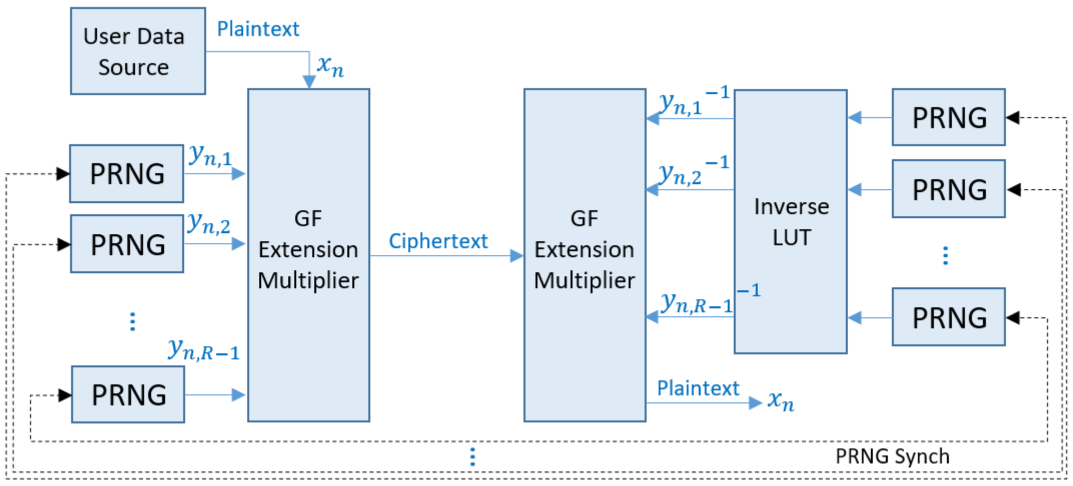

Figure 5.2: Multi-stage cryptographic model using GEF multipliers.

the order in which encryption or decryption occur negligible. Extensions of this core $GF(2^k)$ example may also be made to the broader $GF(p^k)$. When all c terms are chosen identically, the multiplicative inverse lookup tables (LUTs) may be shared, although choosing different c values per input user enables yet another free parameter to be chosen, although a simple to reverse one if the initial key and PRNG mapping are known.

A final extension of this approach is to consider GEF operations performed when the field extension is greater than a single power; i.e., $GF(p^k) \rightarrow GF(p^{k+d})$, where $d \geq 2$. The resulting extension mapping subdivides the larger extension domain by p^d, and enables the incorporation of additional free rotational parameters. Consider the modified GEF mapping for $d = 2$,

$$z_n = \frac{((p^2 x_n + v_{12})(p^2 y_n + v_{34}) \bmod p^{k+2} - v_{12} v_{34} \bmod p^2}{p^2}$$

where $v_{12} = pc_1 + c_2$ and $v_{34} = pc_3 + c_4$.

Here, an additional set of free parameters c_1, c_2, c_3, c_4 may effectively be used to select which of the p^d subsets of $GF(p^{k+d})$ are mapped onto. Extending to larger values of d may be performed without loss of generality.

5.4 Evaluation of Randomness

A candidate example of $GF(2^8)$ using a sample text of 1,043,760 characters (including spaces and punctuation) was tested to illustrate the GEF multiplication arithmetic. The sample text had the following letter frequency shown in Figure 5.3, and the output ciphertext had a distribution over the $GF(2^8)$ elements shown in Figure 5.4. The expected value for each element was 4077 (indicated by the black dashed line) and the standard deviation was 64.9. This distribution is nearly uniform which is crucial for a well designed cipher algorithm. The entropy for this distribution is 7.9998 compared to the maximal entropy, 8, indicating near perfect uniform distribution.

The ciphertext was also evaluated for randomness using the test suite from NIST [44]. The test ran the first 1,000,000 8-bit characters as 1,000 8,000-bit samples through the test suite to determine if the GEF multiplier produced random outputs. The results are displayed in Table 5.1. The test suite calculated that an overall test passed if at least 980 samples passed. These results show that the ciphertext appears random to an outside observer, which is integral to a successful stream cipher.

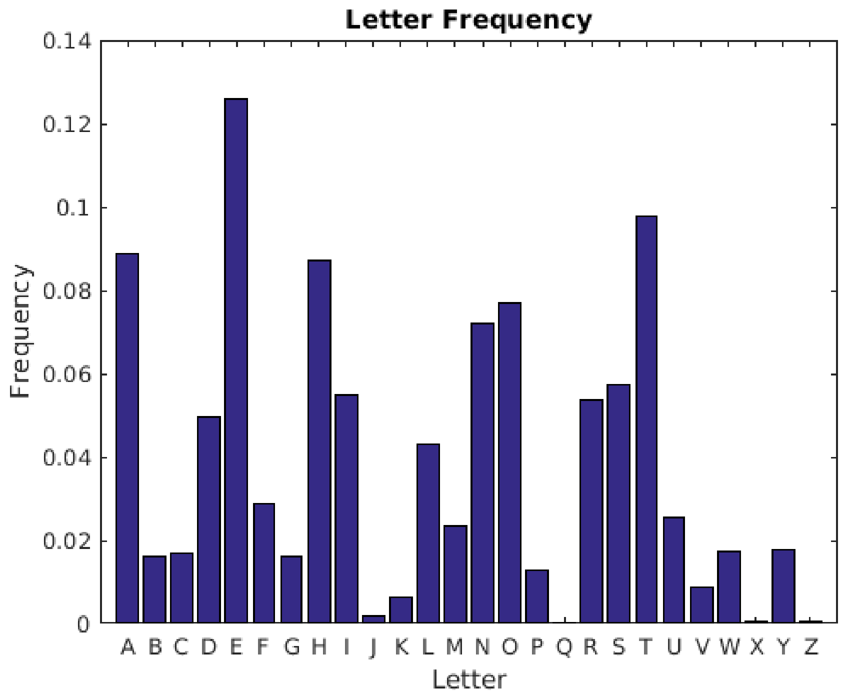

Figure 5.3: Probability distribution of the English alphabet from the sample text used for evaluation of the GEF technique.

Table 5.1: Results of NIST Test Suite

	GEF Multiplication	
Test	**Samples Passed**	**Pass Rate**
Frequency	989	98.9%
Block Frequency	990	99.0%
Cumulative Sums	990	99.0%
Runs	994	99.4%
Longest Run	985	98.5%
FFT	986	98.6%
Approx. Entropy	988	98.8%
Serial	993	99.3%

For more information on the individual tests, see [44].

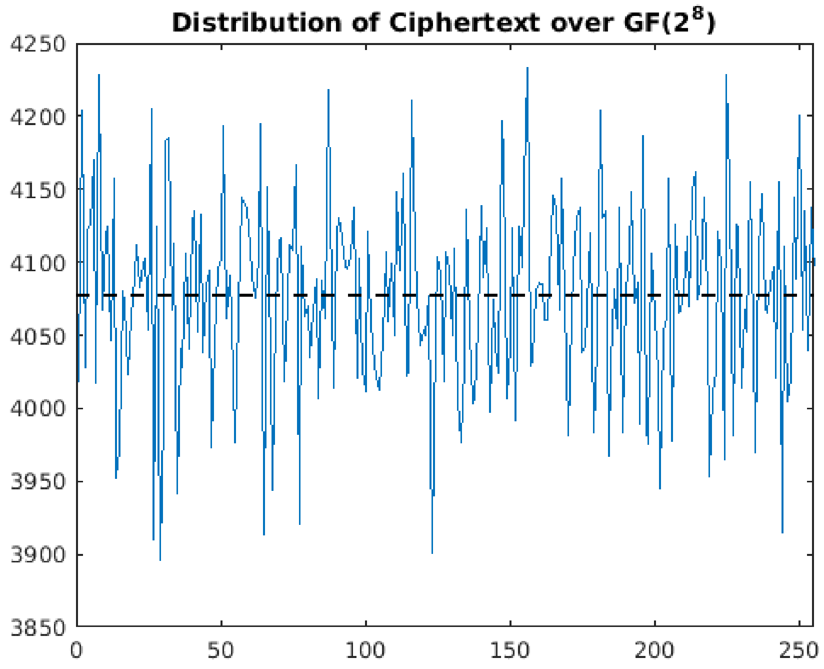

Figure 5.4: Distribution of the ciphertext over the 256-elements of $GF(2^8)$. The expected value for each element was 4077 (indicated by the black dashed line) and the standard deviation was 64.9.

5.5 MSP430 Implementation

In order to verify the low energy consumption claim, an implementation was written in C and was built on a MSP430FR5994 device [188] to illustrate the performance of this encryption scheme on an IoT-like device. This particular MSP430 device has 256 KB of non-volatile FRAM, 8 KB of volatile RAM and was clocked at 16 MHz, so this device is an ideal representation of IoT and WSN devices. This implementation was optimized to remove all `mod` and `div` operations by utilizing shifts. Based on 1,000 encryptions of 16 bytes of data, the average time for encryption was 1.167 ms and required 0.155 μJ/Byte of energy consumption. In order to compare to current encryption schemes, AES in counter mode (using AES as a stream cipher) was also implemented on the same device and ran 1,000 encryptions of 16 bytes of data. This resulted in an average encryption time of 6.379 ms and required 0.8835 μJ/Byte of energy consumption. All energy measurements were performed using the development kit's EnergyTrace software tool [3].

Due to the memory constraints of the MSP430, in order to validate the randomness of the implementation, a 512 8-bit character message was encrypted 512 times to produce 512 unique ciphertexts to generate a large sample size of random values for randomness testing. For each encryption, a new seed was provided to a PRNG [139] to produce a new cryptographic stream for the combination with the message. This provided 2,097,152 bits (2 Mbits) of data to test with the NIST randomness test suite. A distribution of the resulting ciphertexts similar to Figure 5.4 is shown in Figure 5.5. As expected, the distribution appears nearly uniform, indicating that the implementation is valid. The results of the NIST test suite are provided in Table 5.2. All tests passed above the NIST recommended threshold of successful tests (500) to validate the MSP430 implementation.

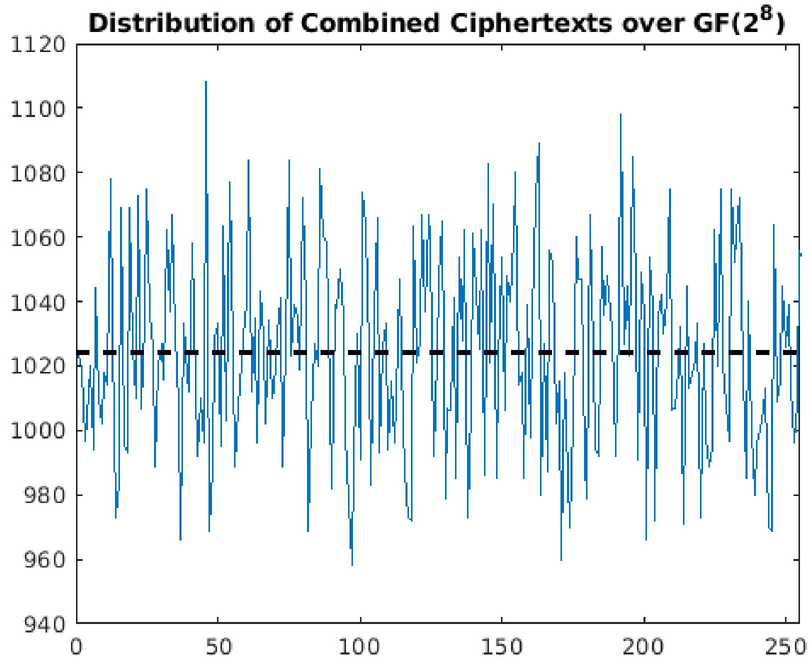

Figure 5.5: Distribution of the combined MSP430 implementation ciphertexts over the 256-elements of $GF(2^8)$. The expected value for each element was 1024 (indicated by the black dashed line) and the standard deviation was 31.02.

One final consideration for this implementation is pipelining of PRNG values. Unlike AES, which must perform block operations for each block of data, this GEF approach can perform the encryption operation once a PRN is available. This was simulated in the MSP430 by pre-loading PRNG outputs into an array (bypassing the PRNG) and encrypting a message. This was then tested by encrypting 10,000 16 byte messages resulting in an average encryption time of 0.305 ms and requiring approximately 40 nJ/Byte.

Table 5.2: Results of NIST Test Suite on MSP430 Implementation

	GEF Multiplication	
Test	**Samples Passed**	**Pass Rate**
Frequency	508	99.2%
Block Frequency	508	99.2%
Cumulative Sums	510	99.6%
Runs	509	99.4%
Longest Run	509	99.4%
FFT	509	99.4%
Approx. Entropy	502	98.1%
Serial	509	99.4%

For more information on the individual tests, see [44].

5.6 Applications of Selectively Invertible Galois Extension Field Techniques

A variety of simple low-computational complexity opportunities exist for the present methods, with an anticipated compatibility with micro-processor sized computational elements supporting adaptation to IoT devices, wireless avionics intra-communications, and related low-power systems. In the case of WSNs, if an edge node device needs to send an encrypted message to the network's central controller, the message may pass through multiple relays or access points. The intermediary devices may not have authorization to the data, but they can add their own encryption layer. This process may be repeated until the encrypted message arrives at the central controller. In this example, the central controller is trusted and has knowledge of all devices and is able to synchronize with each in order to produce the correct PRNG streams to decrypt the entire message. This provides a secure chain and can indicate if the message was tampered with during delivery as the message will not decrypt properly.

The ability to selectively encrypt and decrypt a shared message without consideration of cipher order also opens up additional applications in financial transactions, database management, and approval chains that benefit from not being sequential in nature. For example, in the real estate and home mortgage sector, the order of many of the signings is flexible. The only outcome that matters is that all parties have signed a contract. A trusted outside party (in this case the bank) has the ability to synchronize with all parties in order to decrypt and authenticate the entire transaction, but the individual signers do not need to have any knowledge of others' signatures. This example is illustrated in Figure 5.6 as multiple parties are involved in the home buying contract signing, but ultimately the bank must receive and decrypt the contract and authenticate the signatures.

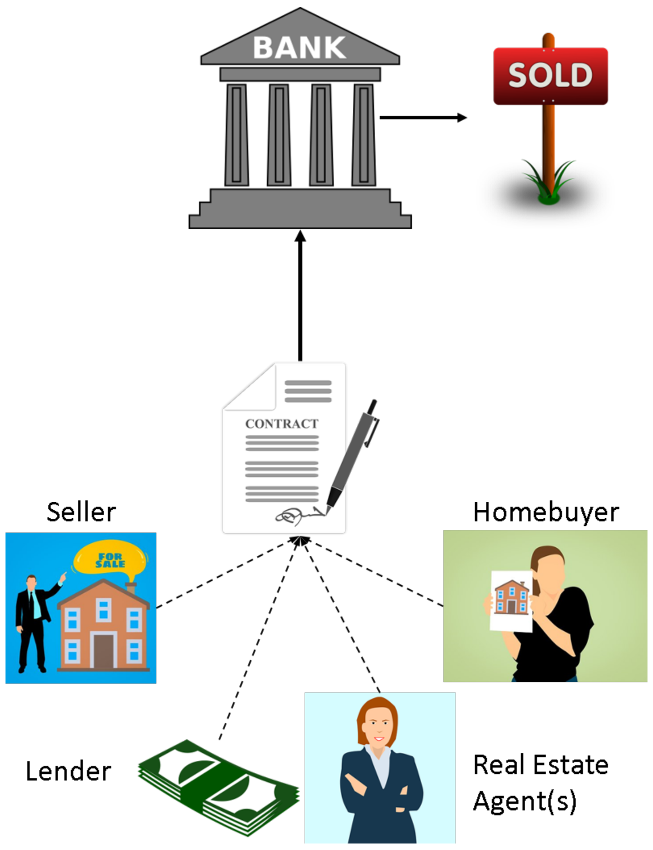

Figure 5.6: Illustration of multiple party signature encryption and authentication. The dashed lines indicate order-independent encryption/signature of a party. Once all parties have signed the contract, the bank is able to decrypt and authenticate all signatures.

5.7 Summary

This chapter adapts a series of computationally efficient PRNG sequence combination techniques built around a Galois extension field multiplication operation to stream-based cryptographic functions. Results from the NIST randomness test suite show promise for sufficient ciphertext randomness. Implementation on an MSP430 was shown to be trivial and easily scalable and produced ciphertexts that also passed the NIST tests. This cryptographic extension supports a variety of low-power moderate security use cases with the added advantages of (1) being scalable in the number of distinct layers of stream ciphers that may be applied and (2) being selectively invertible such that stream-based ciphers may be applied or inverted in any order, giving additional flexibility in the use cases.

Chapter 6

Semi-Coherent Transmission Security

The previous chapters have focused on many of the major challenges pertaining to IoT security. This chapter contributes another layer to aid in providing low-power defense-in-depth for IoT devices: a physical layer transmission security (TRANSEC) method. Historically, PHY layer security and TRANSEC have been shown through theoretical means [207] [124], [94], [190], [48]. Many different methods have been employed to achieve this security. Such work as presented in [219] relies on TDMA multi-user diversity based on channel state information (CSI). Another method described in [74] use the random positions of sub-carriers in orthogonal frequency division multiplexing. Other approaches rely on RF channel characteristics such as multipath or complete RF fingerprinting [180].

These methods mainly focus on using known channel characteristics to change the timing of transmissions. However, it may not be feasible to know the current CSI for low-power devices and the extra computation to continually monitor and adjust adds battery drain. Therefore, a pseudorandom process may be employed to change physical characteristics of the signal components reducing the probability of an attacker to reverse engineer.

By changing the physical transmission modulation properties of the signal in a pseudorandom, yet constrained, manner, the PHY layer security allows the transmitter and receiver to remain synchronized, but should not allow an eavesdropper to easily track the signal phase and subsequently use that phase to cryptanalyze the underlying pseudorandom number generator (PRNG) used to map the signal phase. By retaining the bulk phase of a spreading chip and only dithering over a small region, that phase dither term optionally being uncorrectable, the intended receiver may still receive and demodulate the signal with only a small loss.

The overall contribution of this chapter is the obfuscation of a signal by intentionally injecting noise into the phase mapping process of a spread spectrum signal to decrease an eavesdropper's ability to directly observe the true phase and reverse engineer the associated PRNG output or key, even at high SNR. The rest of the chapter provides an overview of a CDMA PHY layer communication system for this semi-coherent approach. It then describes the integration of the injected semi-coherent signal phases in detail, and presents an exemplary design implementation. Results based on simulations of the design are presented to highlight the effectiveness.

6.1 Acknowledgements

This chapter, in part, is a reprint of the material as it appears in the publication:

- Jason M. McGinthy and Alan J. Michaels. Semi-coherent transmission security for low power IoT devices. In *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 170–177, July 2018. doi: 10.1109/Cybermatics_2018.2018.00059.

The dissertation author was the primary author and Dr. Alan Michaels supervised the research which forms the basis of this chapter.

6.2 System Overview

6.2.1 Spread Spectrum Modulation

In spread spectrum baseband modulations, each data symbol is effectively spread over a larger bandwidth using a spreading sequence of N chips. For purposes of this effort, it is assumed, without loss of generality, that N is chosen as a fixed integer. In some applications, this process allows the signal to be hidden below the noise floor [178], while in commercial IoT applications, this spreading and despreading process offers a viable simplification to co-channel contention processes [151] and built in resilience [196] against natural and man-made interference. Similar sequence-based spread spectrum methods are used in GPS [53], secure digital chaotic sequence spread spectrum (CSSS) systems [138], and commercial datalinks like IEEE 802.11b [16] and CDMA2000 [85]. Since each data symbol is mixed with N chips, the chipping rate is much faster than the data rate. Therefore, a fast chipping rate, which ultimately defines the spread bandwidth, is ideal to allow a larger value of N chips per symbol.

On the receiving end, the signal is mixed with the complex conjugate of the synchronized spreading sequence to reconstruct the original data symbol. The more chips used in the spreading sequence (typically increasing the chip rate), the higher the processing gain [178]. The resulting expected energy per symbol, ε_s, is based on the number of chips used in the spreading sequence, N and the energy per chip, ε_c, as follows

$$E[\varepsilon_s] = N\varepsilon_c. \tag{6.1}$$

A conceptual view of the proposed process is illustrated in Figure 6.1 as the PRNGs of both the transmitter and receiver are synchronized with the same session key producing a

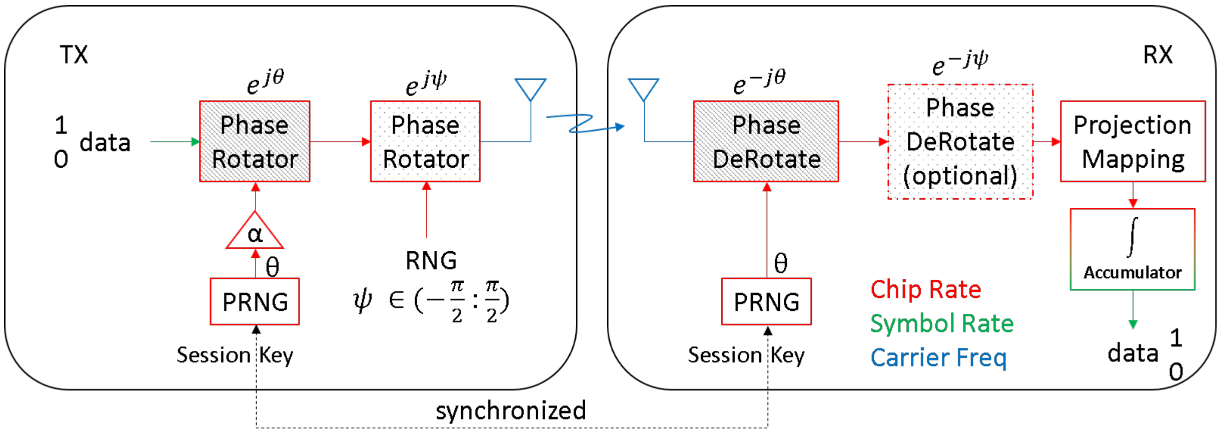

Figure 6.1: Conceptual view of baseband phase rotation with induced error, ψ.

coherent phase, θ. Each data symbol is mixed with a spreading code of N chips, each with a coherent phase, θ, and an induced phase error, ψ, and then transmitted to the receiver. The receiver than attempts to derotate the spreading chips' phases with the complex conjugate of θ. An optional phase derotation may occur if synchronization of the induced error is achievable. Finally, the signal is mapped and passed through an accumulator to produce the data symbol. Note α may be a function of time ($\alpha(t)$) or any other desired parameter, including SNR or other observables at the intended receiver.

One advantage of using a spread spectrum modulation with TRANSEC, is the added parameters for the chips and spreading sequence. This research generally assumes high order PSK signaling (HOPS) signaling techniques, where chip phases are drawn from relatively large M-ary PSK constellations on the unit circle [134]. As long as the transmitter and receiver use a synchronized spreading code, the individual phase of each chip can be altered to reduce the likelihood of an attacker reverse engineering the modulation scheme. The phase of each chip does not have to be the same and can be synchronized by the receiver. Therefore, adding a small error to each chip's phase will add an additional layer of obfuscation for the PRNG output and its associated key that is producing the true phase of each chip. The primary focus for this work is dithering the instantaneous phase of each chip. Since the transmitter and receiver will have a synchronized true phase and only a small error added on the transmitter side, the resultant phase error produces a semi-coherent signal for the receiver. Figure 6.1 highlights the addition of this phase error as well as the synchronization between a transmitter and receiver.

6.2.2 Semi-Coherent TRANSEC

Many TRANSEC approaches rely on the low probability of an attacker synchronizing with a transmitting device, but 0% probability is never a guarantee. A system needs sufficient security in the underlying TRANSEC architecture to reduce an attacker's ability to reverse engineer the process, yet in IoT, the computational burden borne by this processing must remain very low to be practically implementable. Many times a protected session key is used to derive the physical characteristics of the transmitted signal. One approach to reduce the likelihood of an attacker compromising the entire system, as presented in this chapter, is to add an induced error to the phase mapping that obfuscates the actual derived phase thereby making it more difficult for the PRNG output and thereby the session key to be determined. Moreover, this error should not unduly degrade the receiver's ability to receive the signal correctly, but the minor perturbation should make it increasingly difficult for an attacker to know the original value used in the parameter selection. Without that mapping value, cryptanalysis attempts explode into a stochastic search over a much larger search space.

6.2.3 Session Key Protection

Session keys and their synchronization amongst devices provides an authentication mechanism as only authenticated users should have access to the session keys. Therefore, any ability of an outside observer to reverse engineer any portion of this security scheme should be reduced. The main area of focus for this research is the phase of a transmitted sequence-based spread spectrum signal. An attacker must not be able to observe a stream of transmitted signal phases and determine any information that may be used to decipher a session key from those signals. Therefore, our proposed method to reduce the likelihood of this attack by introducing a semi-coherent[1] phase offset error to the actual phase, which should make it infeasible for an attacker to know what the original phase is, effectively obfuscating the associated session key.

6.3 System Design

This design is primarily focused on an induced error, the instantaneous phase error produced, and the impacts this error has on both chip and symbol energies. This section details the different sources of error and the energy calculations based on different distributions of the error.

[1]By choosing a constrained non-coherent phase value to be added to the coherent chip phase, the actual PRNG-driven signal is obfuscated and yet the resulting semi-coherent aggregate signal still contributes to lossy coherent gain in recovering the spread data symbol.

6.3.1 Perturbation Types

Many different approaches exist in the generation of the induced error. Both non-determinsitic and deterministic methods may be considered. The following examples highlight the advantages and disadvantages of certain types of error injection.

Truly Random Number

A TRNG may be implemented into the system to make it exceedingly difficult for an attacker to guess the non-deterministic outputs of the RNG. TRNGs should produce a stream of values with no pattern or periodicity confounding an attacker's ability to remove the error. Generally, TRNGs are unable to generate random numbers as quickly as other pseudorandom processes [200], [181]. Therefore, a TRNG may not be appropriate in contexts where random numbers are needed at a fast rate, since it binds the masking non-coherent phase rotations to a potentially low entropy process when taken at the rate of a spread spectrum communication system. Moreover, a TRNG cannot be synchronized at the intended receiver, eliminating any potential for reducing the effective SNR loss if desired/required.

Unsynchronized Pseudorandom Number

A PRNG may be used to generate the minor perturbations while making it very difficult for an attacker to guess the output. Although not truly random, a RNS PRNG [139] has a very high throughput, allowing the generation of numbers faster compared to many TRNGs. Numerous other high-rate PRNG processes may be used without loss of generality [217], [49], [42], recognizing that absolutely no effort is invested in synchronizing or making the PRNG repeatable. However, one must be mindful when choosing a PRNG as they are deterministic and periodic.

Synchronized Pseudorandom Number

A third option for generating the non-coherent phase addition to the signal chip phase is to use a synchronizable PRNG. This allows for selective dissemination of the additional PRNG parameters to trusted partners within the communication network so that they can achieve fully coherent process of the incoming signal (no loss) while semi-trusted nodes within the network proceed with a parametrically controlled amount of self interference within the signal. Such layered security techniques combine traditional TRANSEC protection with PHY-layer processing constraints, and may be useful for key transfer, network formation, key revocation, and adaptively controlled transmission of data in a unicast or sub-net specific fashion over a public channel. Similar self interference techniques have been proven for PHY-layer only physical processing [135].

Markovian Process

The final option identified is a Markovian process that would be stochastically state driven, configured based on given probabilities while still having some uncertainty of the next value. One prime example of this process is a random walk. Given the memoryless property of Markov chains, the past trajectory cannot be known and the future is still uncertain, yet the long-term stochastic averages may be predicted with prior knowledge of the transition matrix. Such methods may be used by an intended receiver to coarsely synchronize to the injected phase errors.

This research effort focuses on option 2, yet the design on this architecture should be independent of the chosen random number generation technique. All that may be known about this value is its distribution (i.e. uniform, normal, etc.). It shall have a mean, μ and variance, σ^2. An observer may be able to know the range of the random number and calculate the mean, but the range should be large enough that it is infeasible to easily guess the number. Therefore, an attacker may only be able to use the mean and variance to try to deduce the value of the random number.

6.3.2 Phase Modulation

An individual baseband spreading chip may be represented as

$$c[n] = e^{j\phi}$$

where ϕ is the phase for each chip. It is assumed that the center frequency and timing are synchronized, leaving only the signal phase to be considered. The selection of changing the phase of a signal has many benefits. A phase change can be instantaneous per chip and is very easy to control. The addition of a phase offset gains added security at a trade-off of reduction in received signal quality.

Continuous Phase

The chip's phase, ϕ, can be expanded as

$$\phi = \theta + \psi \tag{6.2}$$

where θ is the original phase component and ψ is the induced error. Both θ and $\psi \in [-\pi{:}\pi)$, resulting in ϕ also falling into the same range. Ideally the range of ψ is much smaller as to induce an acceptable error and still allow correct demodulation. Alternatively, the transmitted chip phase may be represented as

$$e^{j\phi} = e^{j\theta} e^{j\psi} \tag{6.3}$$

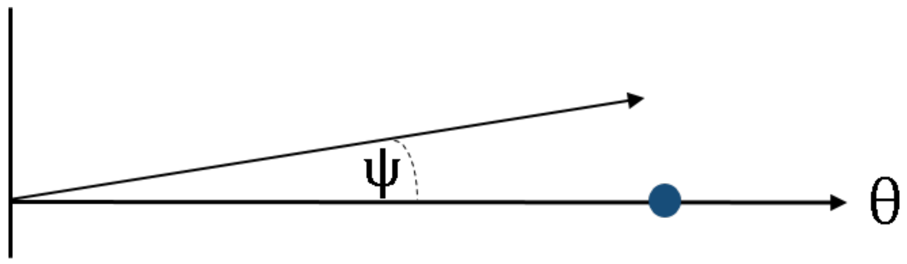

Figure 6.2: Illustration of phase angle error, ψ, caused by addition of small induced error for a fixed phase mapped angle, θ.

and the received chip phase after the complex conjugate multiplication by the coherent bulk phase, θ is

$$e^{j\phi} = e^{-j\theta} e^{j\theta} e^{j\psi} \tag{6.4}$$

which simplifies to a residual error term of $e^{j\phi}$ instead of the ideal location after despreading on the real axis ($\psi = 0$).

The receiver has no way to counteract the dithered phase error, ψ, so the receiver expects the transmitted signal to have a phase of θ. Therefore, the value of θ is arbitrary and can be simplified to 0 for further discussions and calculations through the complex conjugate multiplication with the receiver generated coherent phase, θ. This assumption is displayed in Figure 6.2 as $e^{j\theta}(e^{j\theta})^*$ can be dropped out of the equation when synchronized and the following calculations are based on $\phi = \psi$.

It will also be assumed that the following derivations are absent of noise losses when calculating performance loss between the transmitter and receiver. The actual received energy per chip, ε_c, is dependent only on the phase error, ψ, and defined as

$$\varepsilon_c = \varepsilon_0[\cos\psi + i\sin\psi], \tag{6.5}$$

where ε_0 is an SNR driven amplitude term. Due to an expected zero mean for the induced error, ψ, based on a symmetrical distribution, the imaginary component converges to 0 in expectation as the number of chips per symbol grows larger. Therefore, the average expected coherent energy per chip is only dependent on the real component of the signal and defined as

$$E[\varepsilon_c] = \varepsilon_0 \int_{-\psi_{max}}^{\psi_{max}} f(\psi)\cos\phi \; d\phi \tag{6.6}$$

where $f(\psi)$ is the probability density function (pdf) based on the distribution of ψ. Therefore, for a uniform distribution from $[-\psi_{max} : \psi_{max}]$ we get the following:

$$E[\varepsilon_c] = \varepsilon_0 \frac{1}{2\psi} \int_{-\psi_{max}}^{\psi_{max}} \cos\phi \; d\phi, \tag{6.7}$$

which can be simplified to

$$\frac{E[\varepsilon_c]}{\varepsilon_0} = \frac{\sin\psi}{\psi} \quad \forall\psi \in (0:\pi]. \tag{6.8}$$

The ratio of $\frac{E[\varepsilon_c]}{\varepsilon_0}$ can be considered as an average normalized expected energy per chip, $\hat{\varepsilon_c}$, and will help determine the expected average loss in energy per chip, L (in dB), and is calculated as

$$L = -10\log_{10}\hat{\varepsilon_c}. \tag{6.9}$$

Based on this loss, a maximum loss angle, ψ_{max}, can be calculated by substituting (6.8) into (6.9), which results in

$$L = -10\log_{10}\frac{\sin\psi_{max}}{\psi_{max}}. \tag{6.10}$$

The results of these maximum angles are presented in Table 6.1 based on average acceptable energy loss. Figure 6.3 shows the average expected energy per chip for $\psi_{max} \in (0{:}\pi]$.

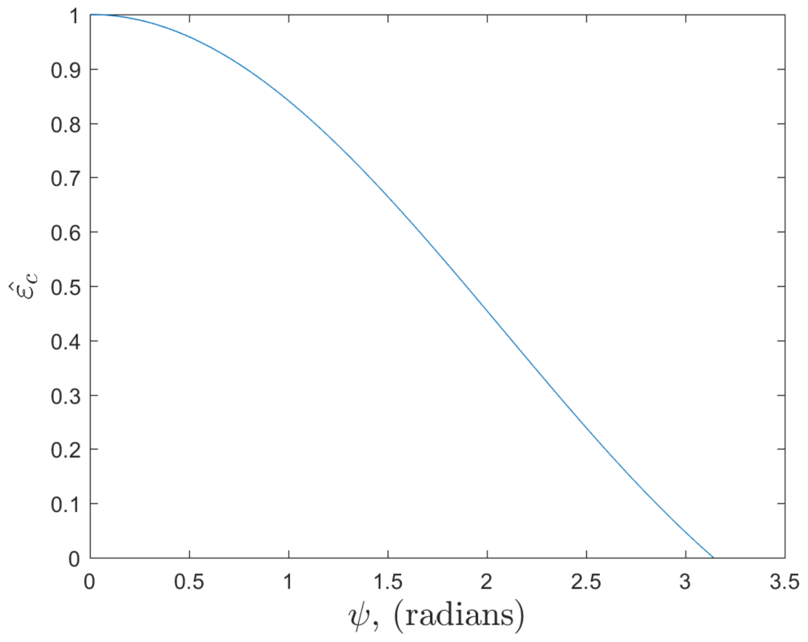

Figure 6.3: Average expected normalized energy per chip for a range bounded by a uniformly distributed range of [-ψ:ψ].

A second option for the distribution for the induced error is a normal distribution. This will weight more of the values closer to the mean and, dependent on the variance, will still allow

Table 6.1: Maximum Phase Angle Offset based on Expected Average Energy Loss

Loss (dB)	ψ_{max}(radians)
0.1	0.3709
0.25	0.5843
0.5	0.822
1	1.149
2	1.585
3	1.893

for smaller probabilities of outliers to occur, further compounding cryptanalysis attempts. This would allow a design to possibly allow the tails of the distribution to extend beyond the 3dB loss limit, but with very low probabilities. The pdf of a normal distribution is defined as

$$f(\psi) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(\psi-\mu)^2}{2\sigma^2}}, \tag{6.11}$$

where μ is the mean and σ is the standard deviation of the distribution. By substituting this pdf into (6.6), the average expected energy per chip over a normally distributed phase offset on the range of $[-\psi_{max} : \psi_{max}]$ is defined as

$$E[\varepsilon_c] = \varepsilon_0 \frac{\sqrt{2\pi\sigma^2}}{2\psi} \int_{-\psi_{max}}^{\psi_{max}} \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(\phi-\mu)^2}{2\sigma^2}} \cos\phi \; d\phi, \tag{6.12}$$

with a scaling factor of $\frac{\sqrt{2\pi\sigma^2}}{2\psi}$ to ensure the pdf integrates to 1. Finally, due to symmetry (6.12) can be rewritten as

$$E[\varepsilon_c] = \varepsilon_0 \frac{1}{\psi} \int_{0}^{\psi_{max}} e^{\frac{-(\phi-\mu)^2}{2\sigma^2}} \cos\phi \; d\phi. \tag{6.13}$$

Ideally, $\mu = 0$ and σ would be chosen such that the design still has a large number of phase states for an acceptable average energy loss. Figure 6.4 shows the effects of σ on the average energy at a given angle, ψ. For small values of σ, the distribution falls off quickly, allowing the average chip energy to converge to a value that additional increases in ψ won't have a large influence. Moreover, as the variance grows larger, the curve approaches the limit of a uniform distribution as shown in Figure 6.3. This is due to the fact that as $\sigma^2 \to \infty$, $e^{\frac{-(\phi-\mu)^2}{2\sigma^2}} \to 1$, and (6.12) simplifies to (6.7).

Similar to Table 6.1 for a uniform distribution, the maximum angle, ψ_{max}, for a given average chip energy can be determined for a normal distribution, as shown in Table 6.2. However, there is no easy closed-form solution, so ψ_{max} can be approximated through simulations such as presented in Figure 6.4. For smaller values of σ, only smaller acceptable losses are achievable. However, as previously mentioned, as σ increases, the values of ψ_{max} approach the values of a uniform distribution (Table 6.1). Additional probability distributions on ψ may be chosen without loss of generality, and that pdf choice may even be made into a time-varying feature of the layered security approach.

Table 6.2: Maximum Phase Angle Offset based on Expected Average Energy Loss for a Normal Distribution

	σ			
Loss (dB)	$\frac{\pi}{8}$	$\frac{\pi}{4}$	$\frac{\pi}{2}$	8π
0.1	0.398	0.378	0.373	0.371
0.25	0.752	0.609	0.589	0.5855
0.5	-	0.891	0.83	0.82
1	-	1.48	1.197	1.156
2	-	-	1.733	1.591
3	-	-	2.207	1.899

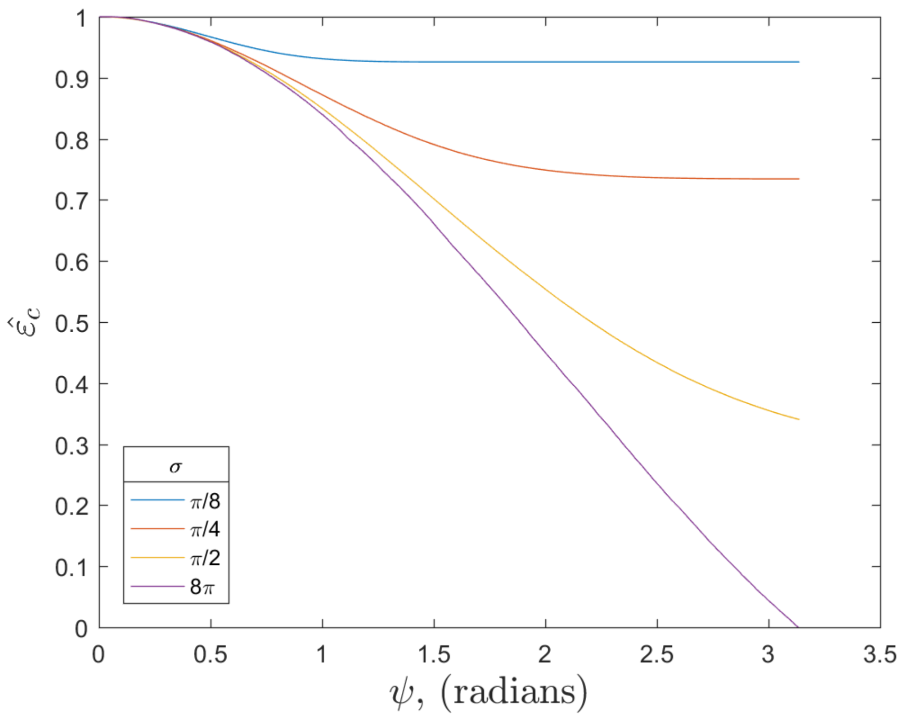

Figure 6.4: Average expected normalized energy per chip for a range bounded by a normally distributed range of $|\psi| \in (0 : \pi]$ at different values of σ.

6.3.3 Discrete Phase State Mapping

All the previous examples were done assuming a continuous range of values, but in reality, this TRANSEC architecture will have an integer number of discretized phase-mapped states, M.[2] These phase states are illustrated in Figure 6.5 with $M = 8$.

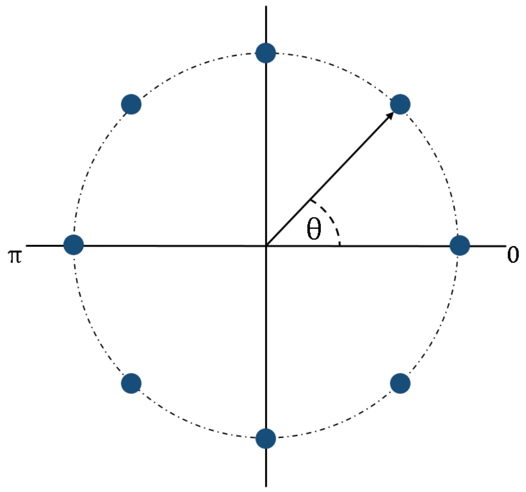

Figure 6.5: Phase mapping example.

The total number of possible phase states changes, m, can be calculated based on the maximum angle of loss, ψ_{max}, and the angle between M-ary phase mapping states, $\angle M$ as follows,

$$m = 2\left\lfloor \frac{\psi_{max}}{|\angle M|} \right\rfloor + 1. \tag{6.14}$$

For example, with $M = 64$ phase states, an acceptable average loss, $L = 2$ dB, and $\psi_{max} = 1.22$ radians, m is 25 allowing for the original phase state and ± 12 possible different phase states offsets. Table 6.3 shows the number of possible state changes of the transmitter based on the acceptable performance loss for different values of phase-discretized M-ary PSK constellations.

Figure 6.6 is an example of a M-ary system with $M = 64$ that illustrates the number of phases states allowed by the average acceptable loss. For example, the red lines bound a 0.25 dB acceptable loss with 11 states, the yellow lines show the limit of a loss of 1 dB and includes 23 states, and the purple lines contain 39 states for an acceptable loss of 3 dB. Note that the range over which the phase perturbations may be made is quite large, and yet retain an acceptable average signal loss.

[2]A nearly continuous phase perturbation may be used without loss of generality simply by allowing the transmitter-side phase mapping of ψ to have a larger phase word resolution than the receiver phase, thereby further increasing security against an observer. The residual phase error will naturally be less than one LSB of the receiver-side representation of θ.

Table 6.3: Number of Possible Phase States based on ψ_{max} with Uniform Distribution

	M-ary Phase Mapping				
Loss (dB)	8	16	64	256	2^{16}
0.1	1	1	7	31	7,745
0.25	1	3	11	47	12,189
0.5	3	5	17	67	17,137
1	3	5	23	93	23,949
2	5	9	33	129	33,063
3	5	9	39	155	39,541

After determining the total number of phase states, the probability of an attacker knowing the true phase value, θ, given that the attacker has the correct transmitted phase, ϕ, is

$$P(\theta|\phi) = \frac{1}{m} \tag{6.15}$$

for a uniformly distributed induced phase error, ψ. Therefore it becomes a design trade-off for the number of desirable states, m, and the acceptable average energy loss, L.

6.3.4 Energy Per Symbol Calculations

All previous calculations were done at the chip level. Now the discussion will shift to spread spectrum symbol energy. The symbol energy, ε_s, was defined in (6.1), but due to the addition of the phase error, ψ, the average expected symbol energy, $\hat{\varepsilon_s}$, must be considered and is defined as

$$\hat{\varepsilon_s} = \varepsilon_c \sum_{n=1}^{N} \cos \psi_n. \tag{6.16}$$

The expected value of $\hat{\varepsilon_s}$ is defined as

$$E[\hat{\varepsilon_s}] = \frac{\hat{\varepsilon_s}}{N\varepsilon_c}. \tag{6.17}$$

As the number of chips, N, increases, the expected value of $\hat{\varepsilon_s}$ shall converge based on the acceptable loss, L, towards $10^{\frac{-L}{10}}$.

6.4 Exemplary Design

An exemplary design of this induced error approach is shown in Figure 6.7. The actual phase state is determined from the RNS PRNG as an 8-bit value representing a single phase on

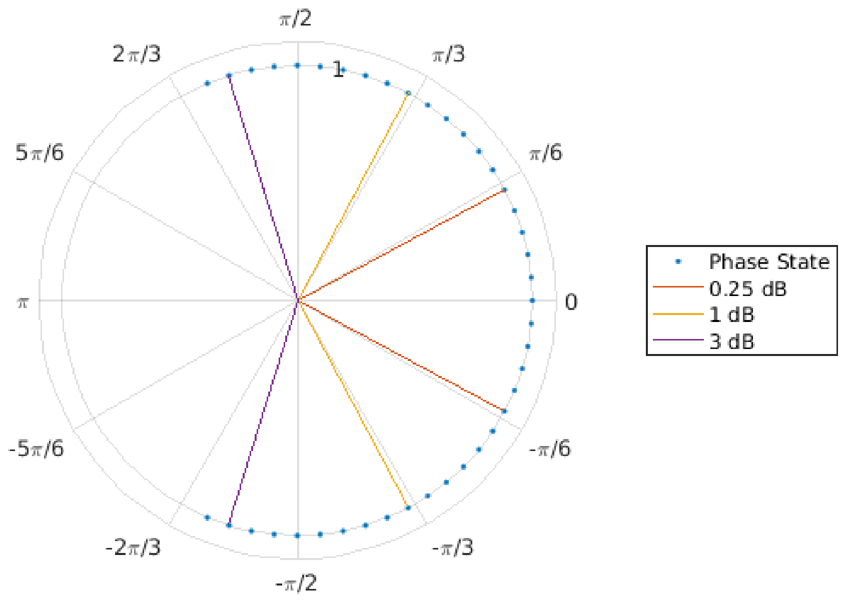

Figure 6.6: Example of 64-ary phase states bounded by the average acceptable loss. Each blue dot represents a discrete phase error that can be added to the true phase. The red, yellow, and purple lines show bounds based on the acceptable performance loss.

a $M = 256$ phase constellation. Then a random number is generated, in this case from a PRNG as a 12-bit value. This 12-bit error is then reduced modulo m, where $m = 67$ to allow an average error loss of 0.5 dB. This reduction only allows a new range of 67 values.[3] Then the value is mapped to the correct phase state range by subtracting 33 ($\lfloor \frac{m}{2} \rfloor - 1$) to map the error range to [-33:33] which is represented by 7 signed bits. Finally, the true phase 8-bit value and the 7-bit error value are added together (accounting for overflow) to produce the TRANSEC modulation phase state value.

6.5 Simulation Results

6.5.1 Calculated Performance Loss

The design shown in Figure 6.7 was modeled in MATLAB to determine the performance loss at different number of phase states, m. 10,000,000 uniformly distributed samples were used

[3]This will not be a perfect uniform distribution. However, it will be approximately uniform due to the large amount of aliasing as described in [129]

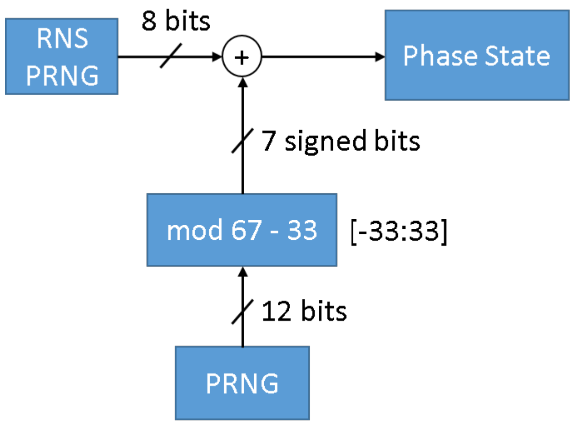

Figure 6.7: Design example with $M = 256$, $L = 0.5$ dB, and $m = 67$.

in the simulation, and the results are shown in Table 6.4. These results reinforce the values of m shown in Table 6.3. For $m = 67$, the percentage of samples that did not have a phase offset (which represents an attacker guessing the actual phase) was $0.0149 \approx \frac{1}{67}$ as expected from (6.15). Similar results were found for the other values of M and m.

Table 6.4: Simulation Results of 256-ary Phase with Uniform Distribution

m	Expected Average Loss	Calculated Average Loss
31	0.1 dB	0.1053 dB
47	0.25 dB	0.2440 dB
67	0.5 dB	0.5021 dB
93	1 dB	0.9888 dB
129	2 dB	1.9797 dB
155	3 dB	3.0493 dB

Again, 10,000,000 chips were simulated for a given acceptable performance loss with phase errors with a normal distribution. For $M = 256$ and $\sigma = \frac{\pi}{2}$, the results are shown in Table 6.5.

6.5.2 Symbol Energy Calculations

Average symbol energy per chip, $\hat{\varepsilon_s}$, calculations were performed in MATLAB for 10 symbols created with increasing values of N chips with both uniform and normal distributions with the results presented in Figure 6.8. The blue dots represent the uniformly distributed errors, and the green dots are the normally distributed phase errors. As is clearly visible, as the

Table 6.5: Simulation Results of 256-ary Phase with Normal Distribution ($\sigma = \frac{\pi}{2}$)

	Expected	Calculated
m	Average Loss	Average Loss
31	0.1 dB	0.0922 dB
47	0.25 dB	0.2207 dB
67	0.5 dB	0.4568 dB
97	1 dB	0.9565 dB
141	2 dB	1.9525 dB
179	3 dB	2.9468 dB

number of chips increased, the values converged toward the expected values, $10^{\frac{-L}{10}}$, due to the law of large numbers. Both the distributions are bounded in the figure by $\pm 3\sigma$ as the red and black lines for each respective distribution.

6.6 Summary

This chapter examined a low-power technique of obfuscating the actual instantaneous phase of each chip with a small induced phase error. Both uniform and normal error distribution were considered, but this design should be agnostic of distribution. The calculations indicate that M-ary PSK constellations can be used to assume an acceptable performance loss based on the number of states of the constellation used. The results presented show that as the number of chips per spread spectrum symbol increase, the average energy per symbol converges towards the expected value.

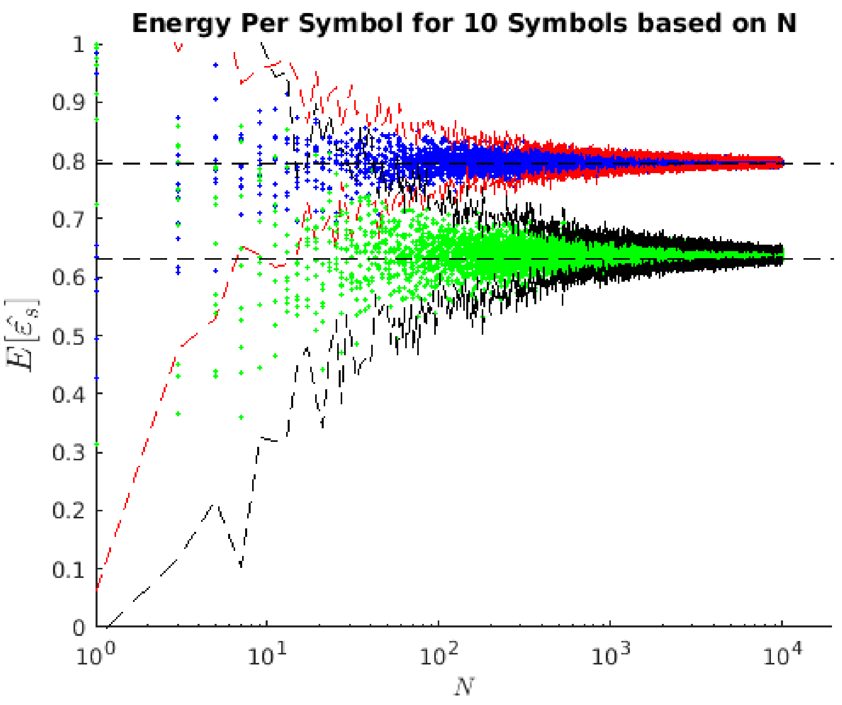

Figure 6.8: Average expected energy per symbol for 10 symbols based on the number of chips, N. Blue dots represent symbols created with uniformly distributed chip phase errors with $M = 256$ and $m = 93$ and an expected energy per chip loss of 1 dB. The green dots represent symbols with normally distributed chip phase errors with the following parameters: $M = 256$, $\sigma = \frac{\pi}{2}$, and $m = 141$ with an expected energy per chip loss of 2 dB. Both simulations converge towards $10^{\frac{-L}{10}}$ based on their respective values of L.

Chapter 7

Conclusions and Future Research

The Internet of Things is currently in its infancy, and there appears to be no upper bound on its potential. Technology typically advances faster than the security measures to protect it, and IoT is no different. As IoT continues to grow into more sectors of our lives, proper security should be in place before the scalability of IoT passes a point of no return. Six IoT scenarios were examined in Section 1.3 to highlight the different security challenges that are prevalent today in those areas. Since IoT aims to help collect and analyze data for safety and security benefits, many of these scenarios face the same issues, but the level of impact is different among them. These challenges are shown again in Figure 7.1 to indicate the severity of these concerns in their respective scenarios. The corresponding publications and dissertation locations are also included for reference of the contributions. Although these challenges are well known in IoT security, efforts are still fractured among researchers, designers, and manufacturers causing a major security concern. The research presented in this dissertation focuses on solutions for what were identified as the four most critical concerns: standardization, trust and authentication, privacy, and information security all while keeping SWaP as a constraint for these solutions due to the limited resources and capabilities of many IoT devices.

Many current security solutions that have been applied to IoT are still based on traditional schemes that don't prioritize the resource constraints facing many of the sensor and devices in IoT systems. Figure 7.2 provides a pictorial representation of IoT and the security areas presented in this dissertation and their approximate locations in the layers of the TCP/IP network security stack. As can be seen from this figure, there are still gaps in IoT security stemming from the application of these solutions that are based on a different security paradigm. These lower layers are the main focus of this dissertation due to the limited capabilities that many IoT devices have. Currently, if higher layer solutions are applied to these devices, some functions are restricted or removed, reducing the effective security for these devices. Therefore, a shift from the old paradigms must be made to focus on the constraints of IoT nodes. Due to the immaturity of IoT, few areas have grasped the techniques to apply solutions at the lower layers of the TCP/IP stack, yet these low layers offer excellent potential for low-power security solutions, as presented in this research.

With these challenges in mind, the main goal of this research was to develop a unified framework based on security features that can apply to all areas of IoT as it focuses on characterizing individual devices independent of their application. Defining these security

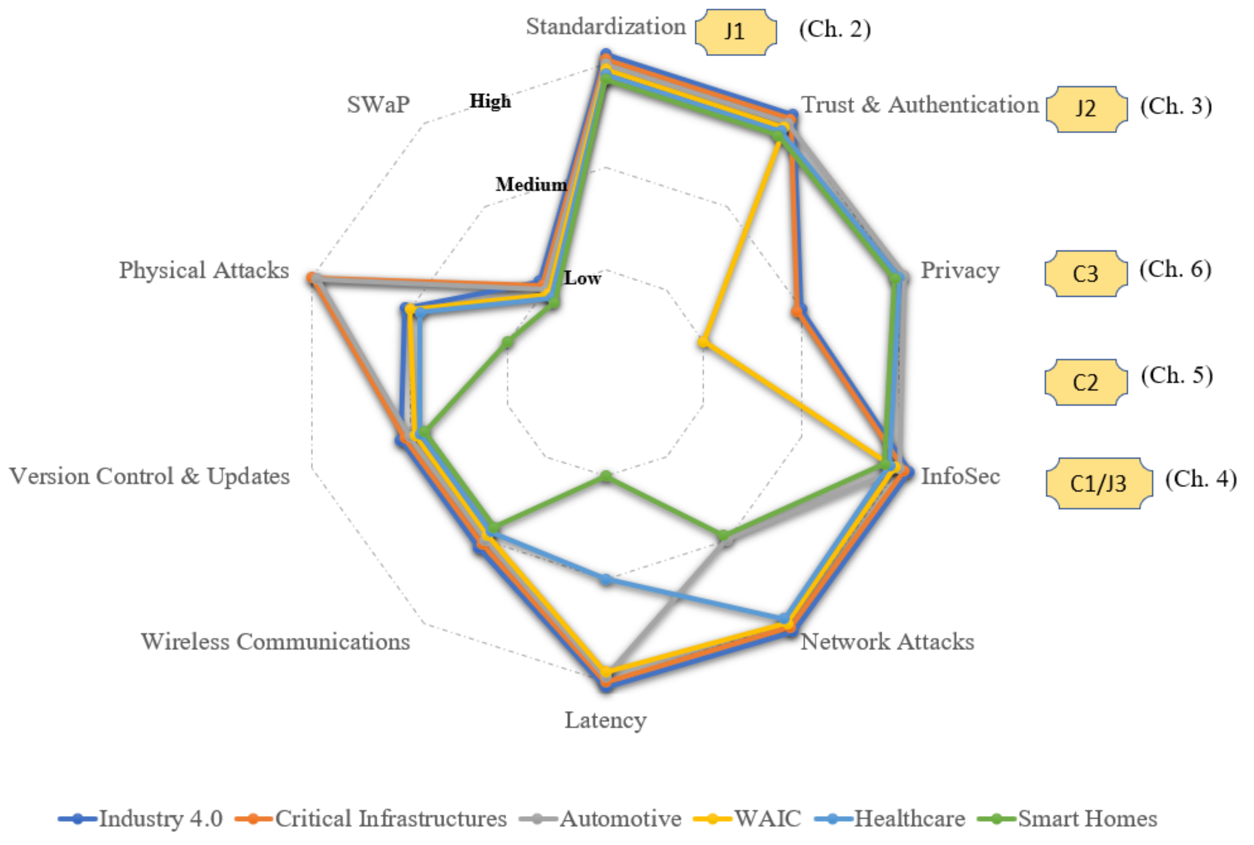

Figure 7.1: Severity of IoT challenges based on different scenarios. Corresponding publications and chapter locations are also included.

features allowed further research into low-power security functions that could be applied at the lowest layers of the TCP/IP network stack. Although the research presented in this dissertation provides a security level framework and component level tools for low-power security features, industry experts and manufacturers must agree on a standardization solution to provide secure longevity for IoT applications. This goes beyond a simple document that outlines levels, as it begins to become a solution based on technology, cost, and political concerns. Industry leaders such as NIST and Cisco must continue to garner support for IoT security at the lowest possible levels. Manufacturers must not shy away from security at the expense of smaller profits because a single security breach can cause major fallout in both reputation and bottom lines. However, if manufacturers must certify devices, it may be worth considering discounts to the cost of such a process in order to have a properly secured system. This can benefit all parties as the true profit in IoT is not from producing low-cost devices, but it is in the data collection and analysis to drive future innovative decisions based on the results. With security upfront, businesses can reduce risk and invest in their products for future gains.

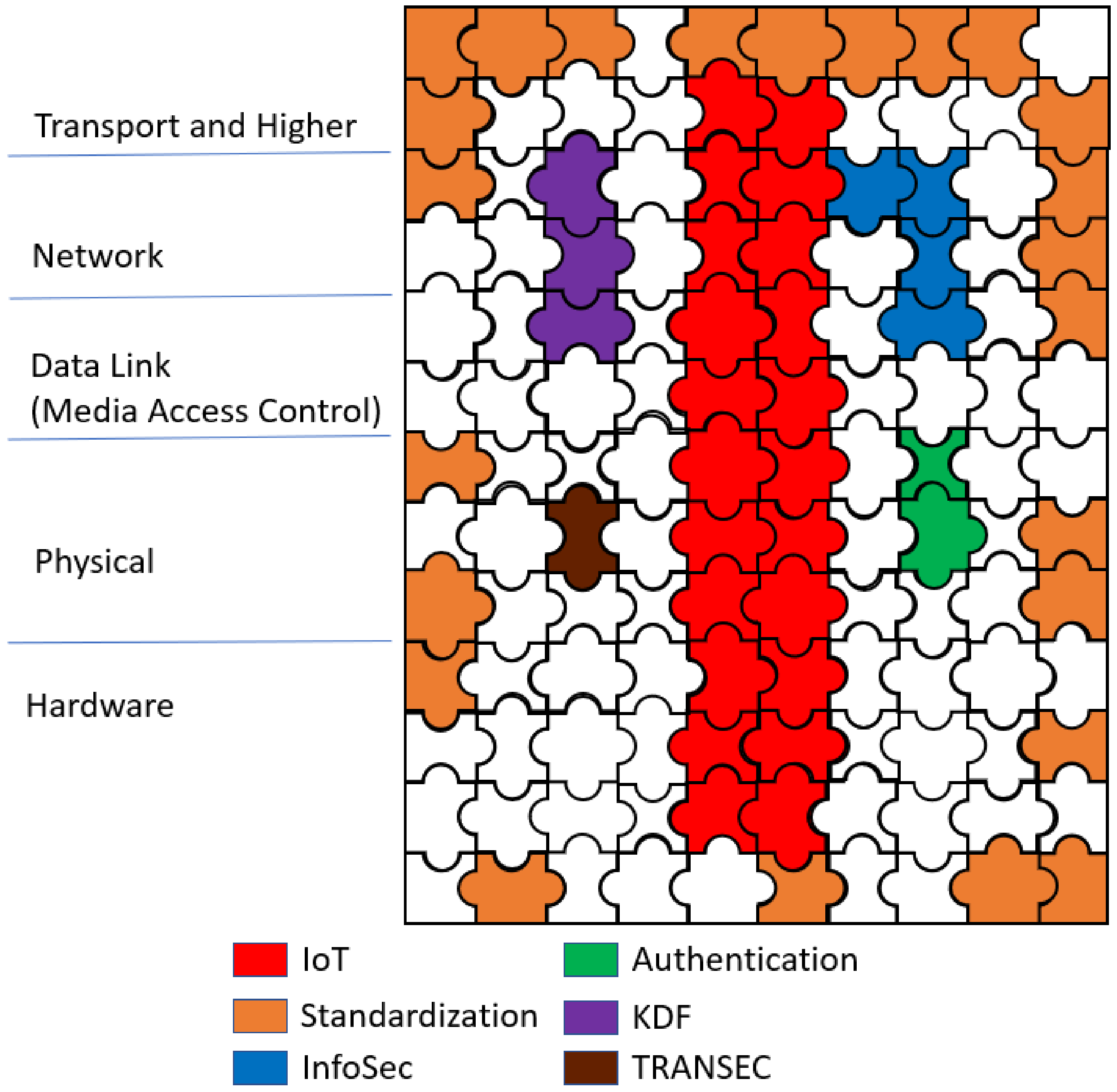

Figure 7.2: Pictorial representation of IoT and lower layer TCP/IP network security and hardware solutions.

7.1 Security Level-based Standardization Framework

The lack of standardization across IoT areas is a crucial security risk as currently there are no checks-and-balances for devices across the IoT landscape. Without formalized standards, security vulnerabilities can arise from unknown incompatibilities among devices. Therefore, the main research contribution presented in Chapter 2 provides a framework for characterizing heterogeneous IIoT devices through defined security levels based on device capabilities due to the expanding nature of IoT and current security concerns at the edge device. By focusing on security at these furthest nodes, the risk of a greater system or overall Internet compromise is greatly reduced. Therefore, more focus on security needs to be included at the lowest layer of IoT (perception) in order to ensure security through an entire system. However, due to the resource constraints of many of these edge nodes, special attention must be taken to allow longevity of remote devices with proper security. Hence, in this work a security level-based architecture was presented with varying degrees of security features. Four classes of devices were characterized based on their available security features. This classification framework allows different systems to identify the types of devices on their networks and implement policies that reduce the risk associated with the differing types of devices. Standardizing security levels of IoT devices will allow a greater sense of trust amongst devices and will ensure larger networks are more secure because security was designed as a foundation instead of an afterthought, and this standardization is also backwards compatible, allowing current implementations to be better characterized and proper security policies put in place. In order to achieve widespread implementation of these standardized classes of devices, designers and manufacturers must ensure that security is a major focus of all IoT devices. Chapters 3-6 aided in developing this overall framework by providing low-power security solutions that may be implemented on very limited devices. The implementation of these security features provides at least a medium level of encryption strength and physical layer security. The benefit of the solutions proposed in this dissertation is the flexibility afforded to allow even low-level options to devices that previously had no security in place. Chapter 2 also presented diagrams of example architectures based on different classes of devices. Figure 7.3 is a similar diagram, but the solutions presented in Chapters 3-6 are labeled to help map the overall contributions to an actual device design.

Expanding on the device security level framework, network scalability must continued to be researched. The interactions of the different device classes must be considered and well understood. There is current research that is exploring these interactions and developing a framework for networks of varying devices. For example, special considerations must be taken to allow numerous class 0 nodes with extremely limited security capabilities. The trust interaction amongst differing classes needs to be robust to possibly only allow periodic access to the network until higher trust metrics are met. These metrics may be determined through the specific emitter identification process presented in Chapter 3.

7.2 Neural Network-based Specific Emitter Identification

Chapter 3 discussed the groundwork for neural network-based specific emitter identification in IoT networks. The need for such authentication is warranted by the ever-growing number of devices in IoT. Current multi-factor authentication methods rely mainly on non-device

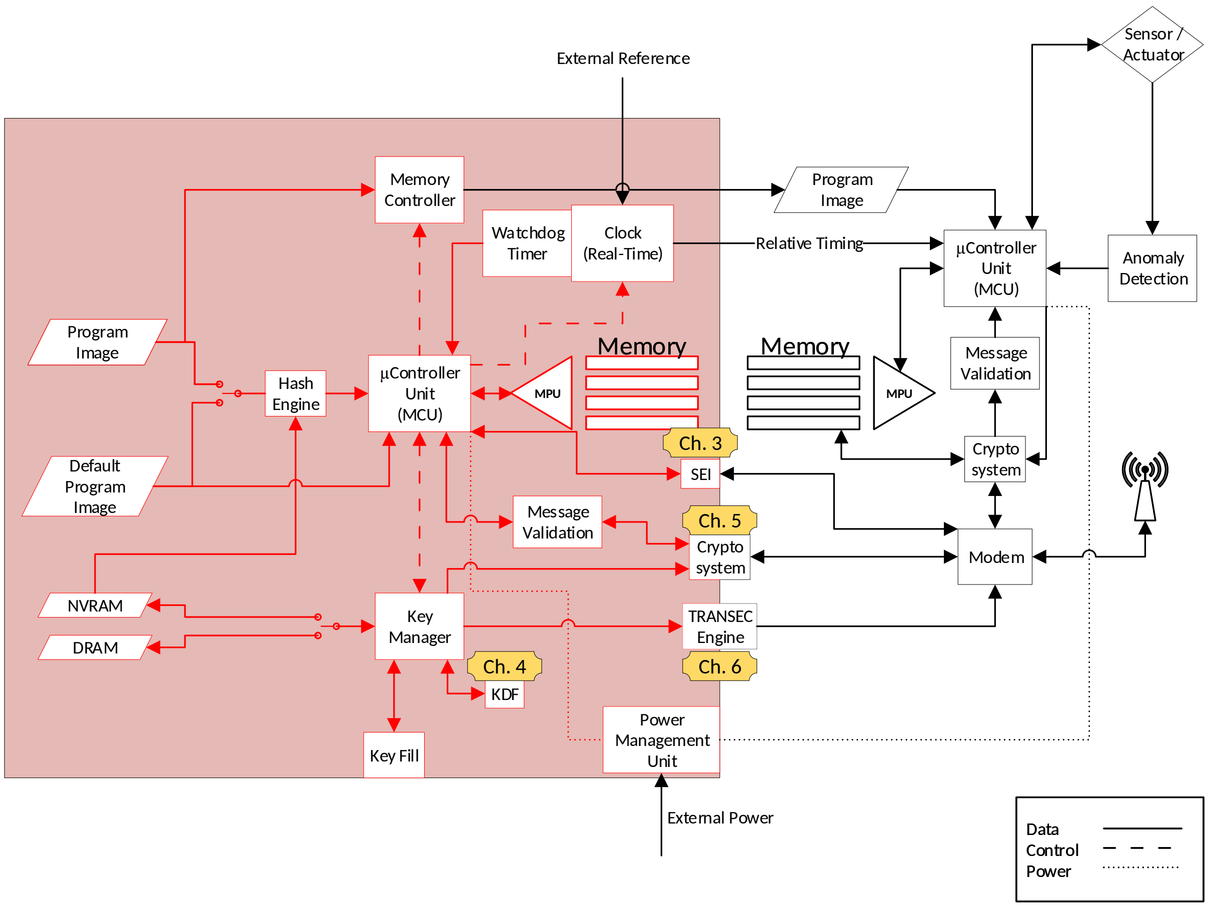

Figure 7.3: Example device architecture with dissertation chapter contributions highlighted.

specific parameters that can be compromised or spoofed. Although MFA is a good approach to authentication, it can become complex when small remote devices are the main focus. Incorporating the wide range of imperfections of the physical components, due to lower tolerances of low-cost devices, allows more unique identifiers for these devices to aid in identification and authentication. By allowing the neural network to extract the best features for device identification, the process can remove the need for human-selected “expert features” and improve overall efficiency. This proved that traditionally computationally expensive techniques, such as RF fingerprinting to perform SEI/node authentication can be efficiently mapped down to SWaP constrained edge nodes. Other applications were also discussed to allow authentication-as-a-service using a separate device or hardware module as additional flexible design options.

Since these SEI techniques can be performed on these constrained devices, future work must examine the scalability of network sizes that can effectively use this type of NN-based authentication. A small network comprised of Raspberry Pi Zero Ws or similar types of devices should be built to test the feasibility of performing SEI at the IoT-device level. The plan includes testing different transmitters and receivers to identify any limitations based on the quality of the components that IoT devices may typically be manufactured with using lower tolerances resulting in larger variances. These large variances should produce more identifying features for the neural networks to discover and use to identify different devices. Results will provide latency, memory footprint, and further insights into probable network sizes.

This SEI approach is one example of a traditionally computationally expensive process. Further research must examine other typically high-power, low duty cycle security mechanisms that can be tailored into IoT solutions. For example, trust-building algorithms may be periodically ran by waking up a receiver. This process may need to compute multiple sources of information to help determine a current level of trust, but the process does not need to be constantly run on a resource-constrained device. This type of process may benefit mesh network type systems where devices may enter and leave, complicating the trust relationship of the entire system.

7.3 PRNG-based Key Derivation Function

Chapter 4 presented a psuedorandom number generator-based key derivation function. Key management is critical in many security functions, and IoT's scalability further highlights the importance of keys. Due to the size, scale, and amount of data collected in IoT, cryptographic keys must be continuously generated to update security functions used in order to reduce the likelihood of an attacker collecting enough repeated observations to reverse engineer the mechanisms. The current technologies are mostly based on more computationally expensive operations such as hash functions. Therefore, this PKDF was designed with resource-constrained IoT edge devices as the main target. This work aimed to remove the need for over-the-air transmission of keying material to reduce the overall energy use by performing the key derivation locally on synchronized devices. This method does not employ a hash-based core function to reduce the energy consumption even further. Instead, a PRNG is used to pseudorandomly select bits from a pre-shared secret master key. The design also allows extremely fast key derivation, allowing keys to be generated orders of magnitudes faster than they can be possibly broken. The variable keys that can be produced can be used in a litany of security functions, even allowing shorter keys for devices that previously may not have been able to include any security features.

Although this PKDF implementation shows great promise for SWaP-constrained devices, the overall security strength has not been fully validated. The design has many parameters in place to protect the key derivation function and provide forward secrecy, but there are concerns that some applications may value proven security (such as the HKDF) over the resource savings. Further efforts must be accomplished to validate the security of the PKDF before it is ready to be used as a full replacement on certain devices and applications.

However, if the speed and energy savings outweigh the need for fully proven security, then the PKDF should be implemented over more resource-heavy solutions. Therefore, the PKDF is an excellent solution for those devices that are unable to implement other more complex KDFs. As is the case for many IoT security solutions, the risks and trade-offs must be scrutinized before designs are implemented.

7.4 Galois Extension Field Cryptographic Functions

Chapter 5 introduced component level tools for trusted interaction between devices as minor contributions in order to help illustrate different types of security features that may be present in different classes of IoT devices. These low-power security primitives were introduced to highlight options that these devices may employ and showed impressive results to provide sufficient security of IoT devices. A Galois extension field cryptographic scheme was developed as a low-power alternative encryption protocol for more efficient information security in IoT. Although there are numerous solutions for encryption, some do not provide addition flexibility warranted by the lack of resources of some IoT nodes. This GEF encryption stream cipher utilizes a variable key length, making it a very flexible option for IoT devices that may not currently have any encryption capabilities due to the higher resource costs of AES or other cryptographic functions. This flexibility allows implementation on many different classes of devices, and it also allows a design feature where the key length can change during operation, making it even more difficult for an attacker to reverse engineer the feature as the function can change before the attacker has enough knowledge to compromise. Also, as a stream cipher, there should not be any lost energy due to padding needed for messages that do not conform to typical block cipher approaches, adding more flexibility for use in different IoT devices.

Another benefit of this scheme is the support for multi-party encryption that is not order-dependent for decryption. This can allow multiple parties to all sign a document and allow a trusted third part to decrypt the file in any order and still recover the original file. Although this multi-party encryption is possible, further research needs to be done to realize other important benefits and uses. For instance, a robust scheme must be devised that allows a trusted outside party to receive and store the keys used in the encryption process. The scheme should also employ a message authentication check that can be used in an audit to ensure that all the parties are encrypting the correct document.

7.5 Semi-Coherent Transmission Security

A final minor effort detailed in Chapter 6 is research focusing on reducing an attacker's ability to repeatedly observe transmitted signal by way of a semi-coherent transmission security technique. The obfuscation of the transmitted signal adds extra privacy protections as the signal is more difficult to be observed. This method allows the introduction of an induced error (either truly random or pseudorandom) to each phase chip of a spread spectrum modulated signal. The amount of error added to each chip could be controlled and designed to be within allowable limits based on the amount of signal loss. This would allow the signal to be degraded, yet still recoverable by the receiver as both the transmitter and receiver would be synchronized. This synchronization allows the receiver to know the expected signal (prior to the induced error), yet an outside observer would not be able to have the same reference, increasing the difficulty of repeatedly correct observations and reducing the likelihood of reverse engineering of the underlying security mechanisms. Two different error distributions were examined and the results matched the expected losses. Employing this technique only requires minor additions to a system in order to add an induced error, making it an excellent choice for an added security layer in a low-power system.

7.6 Ideal Contribution Use Cases

All of the contributions presented in this dissertation are designed for use in IoT devices. However, IoT is diverse in the types of products and the risks associated in different applications. The following bullets highlight the ideal uses for the contributions described in this document:

- Standardization is critical for all security in IoT and should not be limited in its use. The sooner standards are in place, the sooner users can place more confidence in the devices within their networks.
- Neural Network-based Specific Emitter Identification should also not be limited to certain types of devices. Access points can perform the SEI without any additional costs incurred to the edge nodes, and there are solutions that can be implemented to allow edge nodes to also perform SEI periodically.
- PRNG-based Key Derivation Function shows great promise for SWaP-constrained devices such as the MSP430, yet the performance gains drastically drops when used on a more capable device. Therefore, the ideal use for the PKDF is on an 8-bit or 16-bit MCU with limited resources.
- Galois Extension Field Cryptographic Functions provide a fast, efficient stream cipher solution for resource-constrained devices. Ideally, this solution should be implemented on SWaP-constrained devices that do not currently have any cryptographic functions in order to provide a low-cost security solution, while traditional AES-like techniques remain more appropriate for capable devices.
- Semi-Coherent Transmission Security ideally should be used in high reliability systems that employ direct sequence spread spectrum modulation (typically for LPD/I

purposes) and are able to accept slight performance loss in order to obfuscate the underlying security functions.

Figure 7.4 now shows the contributions and future work of the efforts discussed in this dissertation. The labeled areas correspond to the journal and conference publications on those specific topics. The black filled pieces represent future work that is expected to be carried out in continuation. More pieces to the puzzle have been added, but there are still remaining gaps in other areas of challenges regarding IoT security. As IoT is still a *new* technology, time will be needed to develop more security solutions that are included as a foundation.

7.7 Final Remarks

The applications of IoT are extremely broad and there are still future applications that have not even been proposed yet. Just as the Internet shrunk the world and became the central nervous system to billions of people, IoT will take this a step farther and begin to integrate massive amounts of data about our lives, appliances, healthcare, and vehicles. This dissertation provides solutions for just a few of the highest priority challenges facing IoT security. However, there are still other issues remaining that must be researched and solved before IoT security is a commonplace instead of an afterthought. There is no doubt that as more research is done, new IoT-specific solutions will arise and become the foundation for future IoT technologies and applications. The work performed throughout this dissertation provides a basis for these future endeavors.

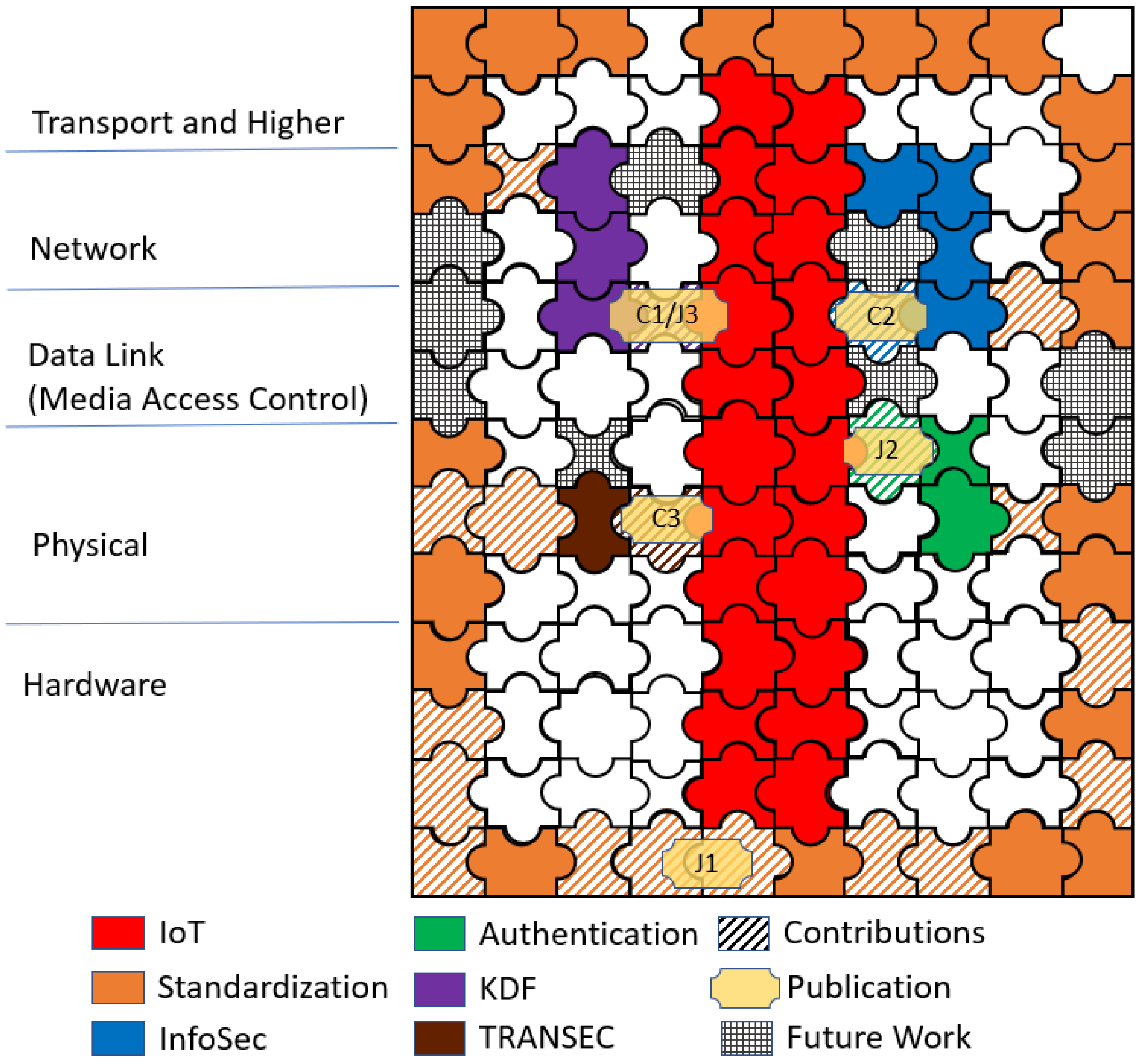

Figure 7.4: Future research will continue to develop new technologies to continue to fill the current gaps in IoT security.

Bibliography

[1] Cortex-m23 - arm developer, . URL https://developer.arm.com/products/processors/cortex-m/cortex-m23.

[2] Arm trustzone, . URL https://www.arm.com/products/security-on-arm/trustzone.

[3] MSP EnergyTrace technology. URL http://www.ti.com/tool/ENERGYTRACE.

[4] Intel sgx homepage. URL https://software.intel.com/sgx.

[5] Home page | LoRa Alliance™. URL https://www.lora-alliance.org/.

[6] Trusted platform module (TPM) summary, . URL https://trustedcomputinggroup.org/trusted-platform-module-tpm-summary/.

[7] Trusted platform module (TPM) 2.0, . URL https://www.trustedcomputinggroup.org/wp-content/uploads/TPM-2.0-A-Brief-Introduction.pdf.

[8] ARM Information Center. URL http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.set.arm11/index.html&_ga=2.116198021.2080172585.1534330215-1141557499.1534330215.

[9] Intel xeon processor e5-1620 v4 (10m cache, 3.50 ghz) product specifications. URL https://ark.intel.com/products/92991/Intel-Xeon-Processor-E5-1620-v4-10M-Cache-3$_$50-GHz.

[10] Cisco Wireless Gateway for LoRaWAN Data Sheet, . URL https://www.cisco.com/c/en/us/products/collateral/se/internet-of-things/datasheet-c78-737307.html.

[11] Devil's Ivy: Flaw in Widely Used Third-party Code Impacts Millions, . URL http://blog.senr.io/1/post/2017/07/devils-ivy-flaw-in-widely-used-third-party-code-impacts-millions.html. accessed 2018-08-08.

[12] RANDOM.ORG - True Random Number Service. URL https://www.random.org/.

[13] Zigbee Alliance. URL http://www.zigbee.org/.

[14] Technical characteristics and operational objectives for wireless avionics intra-communications (WAIC). Tech. Rep M.2197, Geneva, 2010.

[15] IEEE standard for information technology– local and metropolitan area networks– specific requirements– part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications amendment 6: Wireless access in vehicular environments. *IEEE Std 802.11p-2010 (Amendment to IEEE Std 802.11-2007 as amended by IEEE Std 802.11k-2008, IEEE Std 802.11r-2008, IEEE Std 802.11y-2008, IEEE Std 802.11n-2009, and IEEE Std 802.11w-2009)*, pages 1–51, July 2010. doi: 10.1109/IEEESTD.2010.5514475.

[16] Ieee standard for information technology–telecommunications and information exchange between systems local and metropolitan area networks–specific requirements part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications. *IEEE Std 802.11-2012 (Revision of IEEE Std 802.11-2007)*, pages 1–2793, March 2012. doi: 10.1109/IEEESTD.2012.6178212.

[17] Health Information Privacy, August 2015. URL https://www.hhs.gov/hipaa/index.html.

[18] Raspberry Pi Zero: the $5 computer, November 2015. URL https://www.raspberrypi.org/blog/raspberry-pi-zero/.

[19] TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL http://tensorflow.org/. Software available from tensorflow.org.

[20] IEEE standard for low-rate wireless networks. *IEEE Std 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011)*, pages 1–709, April 2016. doi: 10.1109/IEEESTD.2016.7460875.

[21] NHS services hit by cyber-attack, May 2017. URL https://www.bbc.com/news/health-39899646.

[22] The state of IoT security, October 2017. URL https://www.gemalto.com/m2m/documents/iot-security-report.

[23] China Used a Tiny Chip in a Hack That Infiltrated U.S. Companies, October 2018. URL https://www.bloomberg.com/news/features/2018-10-04/the-big-hack-how-china-used-a-tiny-chip-to-infiltrate-america-s-top-companies.

[24] Amazon now has more than 100,000 warehouse robots on its payroll, June 2018. URL http://www.dailymail.co.uk/sciencetech/article-5808319/Amazon-100-000-warehouse-robots-company-insists-replace-humans.html.

[25] Medical Device and Health IT Joint Security Plan. Jan 2019. URL https://healthsectorcouncil.org/the-joint-security-plan/.

[26] M. Abomhara and G. M. Køien. Security and privacy in the Internet of Things: Current status and open issues. In *2014 International Conference on Privacy and Security in Mobile Systems (PRISMS)*, pages 1–8, May 2014. doi: 10.1109/PRISMS.2014.6970594.

[27] Carlisle Adams, Guenther Kramer, Serge Mister, and Robert Zuccherato. On the security of key derivation functions. Springer.

[28] Martin Ågren, Martin Hell, Thomas Johansson, and Willi Meier. Grain-128a: A new version of grain-128 with optional authentication. *Int. J. Wire. Mob. Comput.*, 5(1): 48–59, December 2011. ISSN 1741-1084. doi: 10.1504/IJWMC.2011.044106. URL http://dx.doi.org/10.1504/IJWMC.2011.044106.

[29] R. N. Akram, K. Markantonakis, R. Holloway, S. Kariyawasam, S. Ayub, A. Seeam, and R. Atkinson. Challenges of security and trust in avionics wireless networks. In *2015 IEEE/AIAA 34th Digital Avionics Systems Conference (DASC)*, pages 4B1–1–4B1–12, Sep. 2015. doi: 10.1109/DASC.2015.7311416.

[30] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys Tutorials*, 17(4):2347–2376, Fourthquarter 2015. ISSN 1553-877X. doi: 10.1109/COMST.2015.2444095.

[31] Nadhem J AlFardan, Daniel J Bernstein, Kenneth G Paterson, Bertram Poettering, and Jacob CN Schuldt. On the security of RC4 in TLS. In *USENIX Security Symposium*, pages 305–320, 2013.

[32] *Intel® Arria® 10 Device Overview*. Altera, April 2018. URL https://www.altera.com/en_US/pdfs/literature/hb/arria-10/a10_overview.pdf.

[33] Jim Alves-Foss, Paul W Oman, Carol Taylor, and W Scott Harrison. The MILS architecture for high-assurance embedded systems. *International journal of embedded systems*, 2(3-4):239–247, 2006.

[34] E. Aras, G. S. Ramachandran, P. Lawrence, and D. Hughes. Exploring the Security Vulnerabilities of LoRa. In *2017 3rd IEEE International Conference on Cybernetics (CYBCONF)*, pages 1–6, June 2017. doi: 10.1109/CYBConf.2017.7985777.

[35] Kevin Ashton et al. That 'Internet of Things' thing. *RFID journal*, 22(7):97–114, 2009.

[36] *ATmega328/P Datasheet*. Atmel, 2016.

[37] Jean-Philippe Aumasson, Itai Dinur, Luca Henzen, Willi Meier, and Adi Shamir. Efficient FPGA implementations of high-dimensional cube testers on the stream cipher grain-128. *SHARCS'09 Special-purpose Hardware for Attacking Cryptographic Systems*, page 147.

[38] S. Babar, A. Stango, N. Prasad, J. Sen, and R. Prasad. Proposed embedded security framework for internet of things (IoT). In *2011 2nd International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace*

Electronic Systems Technology (Wireless VITAE), pages 1–5, Feb 2011. doi: 10.1109/WIRELESSVITAE.2011.5940923.

[39] Stephanie B Baker, Wei Xiang, and Ian Atkinson. Internet of things for smart healthcare: Technologies, challenges, and opportunities. *IEEE Access*, 5:26521–26544, 2017.

[40] S. Bang, J. Wang, Z. Li, C. Gao, Y. Kim, Q. Dong, Y. P. Chen, L. Fick, X. Sun, R. Dreslinski, T. Mudge, H. S. Kim, D. Blaauw, and D. Sylvester. 14.7 a 288 μw programmable deep-learning processor with 270kb on-chip weight storage using non-uniform memory hierarchy for mobile intelligence. In *2017 IEEE International Solid-State Circuits Conference (ISSCC)*, pages 250–251, Feb 2017. doi: 10.1109/ISSCC.2017.7870355.

[41] Elaine Barker and Allen Roginsky. Recommendation for cryptographic key generation. *NIST Special Publication*, 800, 2012.

[42] Elaine B. Barker and John M. Kelsey. SP 800-90A Rev. 1. recommendation for random number generation using deterministic random bit generators. Technical report, Gaithersburg, MD, United States, 2012.

[43] A. Barki, A. Bouabdallah, S. Gharout, and J. Traoré. M2m security: Challenges and solutions. *IEEE Communications Surveys Tutorials*, 18(2):1241–1254, Secondquarter 2016. ISSN 1553-877X. doi: 10.1109/COMST.2016.2515516.

[44] Lawrence E. Bassham, III, Andrew L. Rukhin, Juan Soto, James R. Nechvatal, Miles E. Smid, Elaine B. Barker, Stefan D. Leigh, Mark Levenson, Mark Vangel, David L. Banks, Nathanael Alan Heckert, James F. Dray, and San Vo. SP 800-22 Rev. 1a. a statistical test suite for random and pseudorandom number generators for cryptographic applications. Technical report, Gaithersburg, MD, United States, 2010.

[45] C. Bertoncini, K. Rudd, B. Nousain, and M. Hinders. Wavelet fingerprinting of radio-frequency identification (RFID) tags. *IEEE Transactions on Industrial Electronics*, 59 (12):4843–4850, Dec 2012. ISSN 0278-0046. doi: 10.1109/TIE.2011.2179276.

[46] Eli Biham and Adi Shamir. *Differential cryptanalysis of the data encryption standard.* Springer Science & Business Media, 2012.

[47] Surupa Biswas, Thomas Carley, Matthew Simpson, Bhuvan Middha, and Rajeev Barua. Memory overflow protection for embedded systems using run-time checks, reuse, and compression. *ACM Transactions on Embedded Computing Systems (TECS)*, 5(4):719–752, 2006.

[48] M. Bloch, J. Barros, M. R. D. Rodrigues, and S. W. McLaughlin. Wireless information-theoretic security. *IEEE Transactions on Information Theory*, 54(6):2515–2534, June 2008. ISSN 0018-9448. doi: 10.1109/TIT.2008.921908.

[49] L. Blum, M. Blum, and M. Shub. A simple unpredictable pseudo-random number generator. *SIAM Journal on Computing*, 15(2):364–383, 1986. doi: 10.1137/0215025. URL https://doi.org/10.1137/0215025.

[50] Eli Blumenthal and Elizabeth Weise. Hacked home devices caused massive internet outage. *USA Today*, Oct 2016. URL https://www.usatoday.com/story/tech/2016/10/21/cyber-attack-takes-down-east-coast-netflix-spotify-twitter/92507806/.

[51] Andrey Bogdanov, Lars R Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew JB Robshaw, Yannick Seurin, and Charlotte Vikkelsoe. PRESENT: An ultra-lightweight block cipher. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 450–466. Springer, 2007.

[52] K. Bourzac. Speck-size computers: Now with deep learning [News]. *IEEE Spectrum*, 54(4):13–15, April 2017. ISSN 0018-9235. doi: 10.1109/MSPEC.2017.7880447.

[53] M. S. Braasch and A. J. van Dierendonck. GPS receiver architectures and measurements. *Proceedings of the IEEE*, 87(1):48–64, Jan 1999. ISSN 0018-9219. doi: 10.1109/5.736341.

[54] Jordan Valinsky Business, CNN. Amazon reportedly employs thousands of people to listen to your Alexa conversations. URL https://www.cnn.com/2019/04/11/tech/amazon-alexa-listening/index.html.

[55] J. Cañedo and A. Skjellum. Using machine learning to secure iot systems. In *2016 14th Annual Conference on Privacy, Security and Trust (PST)*, pages 219–222, Dec 2016. doi: 10.1109/PST.2016.7906930.

[56] S. Chang, R. Chiang, S. Wu, and W. Chang. A context-aware, interactive m-health system for diabetics. *IT Professional*, 18(3):14–22, May 2016. ISSN 1520-9202. doi: 10.1109/MITP.2016.48.

[57] B. Chatterjee, D. Das, S. Maity, and S. Sen. RF-PUF: Enhancing IoT security through authentication of wireless nodes using in-situ machine learning. *IEEE Internet of Things Journal*, pages 388–398, 2018. doi: 10.1109/JIOT.2018.2849324.

[58] Lily Chen. Recommendation for key derivation using pseudorandom functions. *NIST special publication*, 800:108, 2008.

[59] Jiangfeng Cheng, Weihai Chen, Fei Tao, and Chun-Liang Lin. Industrial IoT in 5G environment towards smart manufacturing. *Journal of Industrial Information Integration*, 10:10–19, 2018.

[60] Howard C Choe, Robert E Karlsen, Grant R Gerhart, and Thomas J Meitzler. Wavelet-based ground vehicle recognition using acoustic signals. In *Wavelet Applications III*, volume 2762, pages 434–446. International Society for Optics and Photonics, 1996.

[61] François Chollet. Keras. https://github.com/fchollet/keras, 2015.

[62] Bill Clark. Efficient waveform spectrum aggregation for algorithm verification and validation, Sep 2016. URL https://gnuradio.org/grcon-2016/talks/.

[63] CNBC. US charges Chinese companies in trade secrets theft, November 2018. URL https://www.cnbc.com/2018/11/01/us-charges-chinese-companies-in-trade-secrets-theft.html.

[64] Louis Columbus. 2018 Roundup Of Internet Of Things Forecasts And Market Estimates. URL https://www.forbes.com/sites/louiscolumbus/2018/12/13/2018-roundup-of-internet-of-things-forecasts-and-market-estimates/.

[65] Henry Corrigan-Gibbs, Wendy Mu, Dan Boneh, and Bryan Ford. Ensuring high-quality randomness in cryptographic key generation. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 685–696. ACM, 2013.

[66] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, 2006. ISBN 0471241954.

[67] Robert K Crane. *Propagation handbook for wireless communication system design*. CRC press, 2003.

[68] Adéle Da Veiga and Jan HP Eloff. A framework and assessment instrument for information security culture. *Computers & Security*, 29(2):196–207, 2010.

[69] Quynh H Dang. NIST FIPS PUB 180-4 secure hash standard. Technical report, 2015.

[70] J. Deogirikar and A. Vidhate. Security attacks in iot: A survey. In *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, pages 32–37, Feb 2017. doi: 10.1109/I-SMAC.2017.8058363.

[71] Itai Dinur, Tim Güneysu, Christof Paar, Adi Shamir, and Ralf Zimmermann. An experimentally verified attack on full grain-128 using dedicated reconfigurable hardware. In *ASIACRYPT*, volume 7073, pages 327–343. Springer, 2011.

[72] Morris J Dworkin. NIST FIPS PUB 202 SHA-3 standard: Permutation-based hash and extendable-output functions. Technical report, 2015.

[73] Morris J Dworkin. SHA-3 standard: Permutation-based hash and extendable-output functions. Technical report, 2015.

[74] D. Efstathiou, G. D. Papadopoulos, D. Tsipouridou, and F. N. Pavlidou. Enhancement of transmission security for OFDM based systems. In *2017 IEEE Symposium on Computers and Communications (ISCC)*, pages 546–551, July 2017. doi: 10.1109/ISCC.2017.8024585.

[75] Ted Eisenberg, David Gries, Juris Hartmanis, Don Holcomb, M Stuart Lynn, and Thomas Santoro. The cornell commission: on morris and the worm. *Communications of the ACM*, 32(6):706–710, 1989.

[76] S. Elbouanani, M. A. E. Kiram, and O. Achbarou. Introduction to the internet of things security: Standardization and research challenges. In *2015 11th International Conference on Information Assurance and Security (IAS)*, pages 32–37, Dec 2015. doi: 10.1109/ISIAS.2015.7492741.

[77] K. J. Ellis and N. Serinken. Characteristics of radio transmitter fingerprints. *Radio Science*, 36(4):585–597, July 2001. ISSN 1944-799X. doi: 10.1029/2000RS002345.

[78] A. Esfahani, G. Mantas, R. Matischek, F. B. Saghezchi, J. Rodriguez, A. Bicaku, S. Maksuti, M. G. Tauber, C. Schmittner, and J. Bastos. A lightweight authentication mechanism for M2M communications in industrial IoT environment. *IEEE Internet of Things Journal*, 6(1):288–296, Feb 2019. ISSN 2327-4662. doi: 10.1109/JIOT.2017.2737630.

[79] EU Commission. Communication from the commission on a european programme for critical infrastructure protection. *COM (2006)*, 786, 2006.

[80] Dave Evans. The internet of things: How the next evolution of the internet is changing everything. *CISCO white paper*, 1(2011):1–11, 2011.

[81] S. Fluhrer and D. McGrew. Statistical analysis of the alleged RC4 key stream generator. In *Proceedings, Fast Software Encryption 2000*, 2000.

[82] Scott Fluhrer, Itsik Mantin, and Adi Shamir. Weaknesses in the key scheduling algorithm of RC4. In *Proceedings, Workshop in Selected Areas of Cryptography*, 2001.

[83] Aurelien Francillon, Boris Danev, and Srdjan Capkun. Relay Attacks on Passive Keyless Entry and Start Systems in Modern Cars. Technical Report 332, 2010. URL http://eprint.iacr.org/2010/332.

[84] M. Frustaci, P. Pace, G. Aloi, and G. Fortino. Evaluating critical security issues of the IoT world: Present and Future challenges. *IEEE Internet of Things Journal*, PP(99): 1–1, 2017. doi: 10.1109/JIOT.2017.2767291.

[85] Vijay K Garg. *IS-95 CDMA and CDMA2000: Cellular/PCS systems implementation.* Pearson Education, 1999.

[86] Blaise Gassend, Dwaine Clarke, Marten van Dijk, and Srinivas Devadas. Silicon physical random functions. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, CCS '02, pages 148–160, New York, NY, USA, 2002. ACM. ISBN 1-58113-612-9. doi: 10.1145/586110.586132. URL http://doi.acm.org/10.1145/586110.586132.

[87] G. Gong, S. Ronjom, T. Helleseth, and H. Hu. Fast discrete Fourier spectra attacks on stream ciphers. *IEEE Transactions on Information Theory*, 57(8):5555–5565, Aug 2011. ISSN 0018-9448. doi: 10.1109/TIT.2011.2158480.

[88] Q. Gou, L. Yan, Y. Liu, and Y. Li. Construction and Strategies in IoT Security System. In *2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing*, pages 1129–1132, August 2013. doi: 10.1109/GreenCom-iThings-CPSCom.2013.195.

[89] J. Granjal, E. Monteiro, and J. Sá Silva. Security for the internet of things: A survey of existing protocols and open research issues. *IEEE Communications Surveys Tutorials*, 17(3):1294–1312, thirdquarter 2015. ISSN 1553-877X. doi: 10.1109/COMST.2015.2388550.

[90] Nathaniel A. Hall, William C. Headley, Thomas G. Krauss, and Alan J. Michaels. Modeling the probability of 802.11g frame collisions in a hidden terminal environment. In *MILCOM 2017 - 2017 IEEE Military Communications Conference*, 2017.

[91] Shaikh Shahriar Hassan, Soumik Das Bibon, Md Shohrab Hossain, and Mohammed Atiquzzaman. Security threats in Bluetooth technology. *Computers & Security*, 74: 308 – 322, 2018. ISSN 0167-4048. doi: https://doi.org/10.1016/j.cose.2017.03.008.

[92] M. Hell, T. Johansson, A. Maximov, and W. Meier. A stream cipher proposal: Grain-128. In *2006 IEEE International Symposium on Information Theory*, pages 1614–1618, July 2006. doi: 10.1109/ISIT.2006.261549.

[93] M. Hermann, T. Pentek, and B. Otto. Design principles for industrie 4.0 scenarios. In *2016 49th Hawaii International Conference on System Sciences (HICSS)*, pages 3928–3937, Jan 2016. doi: 10.1109/HICSS.2016.488.

[94] A. O. Hero. Secure space-time communication. *IEEE Transactions on Information Theory*, 49(12):3235–3249, Dec 2003. ISSN 0018-9448. doi: 10.1109/TIT.2003.820010.

[95] M. Hicks. Clank: Architectural support for intermittent computation. In *2017 ACM/IEEE 44th Annual International Symposium on Computer Architecture (ISCA)*, pages 228–240, June 2017. doi: 10.1145/3079856.3080238.

[96] G. Hospodar, R. Maes, and I. Verbauwhede. Machine learning attacks on 65nm arbiter PUFs: Accurate modeling poses strict bounds on usability. In *2012 IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 37–42, Dec 2012. doi: 10.1109/WIFS.2012.6412622.

[97] G. Huang, Y. Yuan, X. Wang, and Z. Huang. Specific emitter identification based on nonlinear dynamical characteristics. *Canadian Journal of Electrical and Computer Engineering*, 39(1):34–41, winter 2016. ISSN 0840-8688. doi: 10.1109/CJECE.2015.2496143.

[98] Carnegie Mellon University Software Engineering Institute. The CERT Division, 2019. URL https://www.sei.cmu.edu/about/divisions/cert/index.cfm.

[99] Qi Jing, Athanasios V. Vasilakos, Jiafu Wan, Jingwei Lu, and Dechao Qiu. Security of the Internet of Things: perspectives and challenges. *Wireless Networks*, 20(8):2481–2501, November 2014. ISSN 1022-0038, 1572-8196. doi: 10.1007/s11276-014-0761-7. URL http://link.springer.com/10.1007/s11276-014-0761-7.

[100] Marcio Juliato and Catherine Gebotys. FPGA implementation of an HMAC processor based on the SHA-2 family of hash functions. Technical report, 2011.

[101] S. K. K, S. Sahoo, A. Mahapatra, A. K. Swain, and K. K. Mahapatra. Security Enhancements to System on Chip Devices for IoT Perception Layer. In *2017 IEEE International Symposium on Nanoelectronic and Information Systems (iNIS)*, pages 151–156, December 2017. doi: 10.1109/iNIS.2017.39.

[102] Masanobu Katagi and Shiho Moriai. Lightweight cryptography for the internet of things. *Sony Corporation*, pages 7–10, 2008.

[103] D. K. Kilcoyne, S. Bendelac, J. M. Ernst, and A. J. Michaels. Tire pressure monitoring system encryption to improve vehicular security. In *MILCOM 2016 - 2016 IEEE Military Communications Conference*, pages 1219–1224, Nov 2016. doi: 10.1109/MILCOM.2016.7795497.

[104] K. Kim, C. M. Spooner, I. Akbar, and J. H. Reed. Specific emitter identification for cognitive radio with application to IEEE 802.11. In *IEEE GLOBECOM 2008 - 2008 IEEE Global Telecommunications Conference*, pages 1–5, Nov 2008. doi: 10.1109/GLOCOM.2008.ECP.404.

[105] Kyou Woong. Kim. *Exploiting cyclostationarity for radio environmental awareness in cognitive radios*. PhD thesis, Virginia Polytechnic and State University, 2008.

[106] Yoongu Kim, Ross Daly, Jeremie Kim, Chris Fallin, Ji Hye Lee, Donghyuk Lee, Chris Wilkerson, Konrad Lai, and Onur Mutlu. Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors. *SIGARCH Comput. Archit. News*, 42(3):361–372, June 2014. ISSN 0163-5964. doi: 10.1145/2678373.2665726. URL http://doi.acm.org/10.1145/2678373.2665726.

[107] Lars R. Knudsen, Willi Meier, Bart Preneel, Vincent Rijmen, and Sven Verdoolaege. *Analysis Methods for (Alleged) RC4*. 1998.

[108] Neal Koblitz and Alfred Menezes. A riddle wrapped in an enigma. *IEEE Security & Privacy*, 14(6):34–42, 2016.

[109] Kirsten Koepsel. *Commercial Aviation and Cyber Security: A Critical Intersection*. SAE, 2017.

[110] Hugo Krawczyk and Pasi Eronen. HMAC-based extract-and-expand key derivation function (HKDF). 2010.

[111] Hugo Krawczyk, Mihir Bellare, and Ran Canetti. HMAC: Keyed-hashing for message authentication. RFC 2104, February 1997.

[112] Shancang Li and Li Da Xu. *Securing the internet of things*. Syngress, 2017.

[113] Shancang Li, Li Da Xu, and Shanshan Zhao. 5G internet of things: A survey. *Journal of Industrial Information Integration*, 10, February 2018. doi: 10.1016/j.jii.2018.01.005.

[114] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao. A survey on internet of things: Architecture, enabling technologies, security and privacy, and applications. *IEEE Internet of Things Journal*, 4(5):1125–1142, Oct 2017. ISSN 2327-4662. doi: 10.1109/JIOT.2017.2683200.

[115] Y. Lu and L. Da Xu. Internet of things (IoT) cybersecurity research: A review of current research topics. *IEEE Internet of Things Journal*, 2018. ISSN 2327-4662. doi: 10.1109/JIOT.2018.2869847.

[116] Brandon Lucia and Benjamin Ransford. A simpler, safer programming and execution model for intermittent systems. *SIGPLAN Not.*, 50(6):575–585, June 2015. ISSN 0362-1340. doi: 10.1145/2813885.2737978. URL http://doi.acm.org/10.1145/2813885.2737978.

[117] Z. Ma, T. Tian, and W. F. Qi. Conditional differential attacks on grain-128a stream cipher. *IET Information Security*, 11(3):139–145, 2017. ISSN 1751-8709. doi: 10.1049/iet-ifs.2016.0060.

[118] R. Mahmoud, T. Yousuf, F. Aloul, and I. Zualkernan. Internet of things (IoT) security: Current status, challenges and prospective measures. In *2015 10th International Conference for Internet Technology and Secured Transactions (ICITST)*, pages 336–341, December 2015. doi: 10.1109/ICITST.2015.7412116.

[119] S. Maitra and E. Pasalic. Further constructions of resilient boolean functions with very high nonlinearity. *IEEE Transactions on Information Theory*, 48(7):1825–1834, Jul 2002. ISSN 0018-9448. doi: 10.1109/TIT.2002.1013128.

[120] Itsik Mantin and Adi Shamir. A practical attack on broadcast RC4. In *International Workshop on Fast Software Encryption*, pages 152–164. Springer, 2001.

[121] MATLAB and Simulink Toolbox. *version 9.3 (R2017b)*. The MathWorks Inc., Natick, Massachusetts, 2016.

[122] Makoto Matsumoto and Takuji Nishimura. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 8(1):3–30, 1998.

[123] J. Matuszewski and K. Sikorska-Łukasiewicz. Neural network application for emitter identification. In *2017 18th International Radar Symposium (IRS)*, pages 1–8, June 2017. doi: 10.23919/IRS.2017.8008202.

[124] Ueli M Maurer. Secret key agreement by public discussion from common information. *IEEE transactions on information theory*, 39(3):733–742, 1993.

[125] R. P. McEvoy, F. M. Crowe, C. C. Murphy, and W. P. Marnane. Optimisation of the SHA-2 family of hash functions on FPGAs. In *IEEE Computer Society Annual Symposium on Emerging VLSI Technologies and Architectures (ISVLSI'06)*, pages 6–pp, March 2006. doi: 10.1109/ISVLSI.2006.70.

[126] J. M. McGinthy, L. J. Wong, and A. J. Michaels. Groundwork for neural network-based specific emitter identification authentication for IoT. *IEEE Internet of Things Journal*, 2019. ISSN 2327-4662. doi: 10.1109/JIOT.2019.2908759.

[127] Jason M. McGinthy and Alan J. Michaels. Further analysis of PRNG-based key derivation function. *IEEE Access*.

[128] Jason. M. McGinthy and Alan. J. Michaels. Lightweight internet of things encryption using Galois extension field arithmetic. In *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 74–80, July 2018. doi: 10.1109/Cybermatics_2018.2018.00046.

[129] Jason M. McGinthy and Alan J. Michaels. Session key derivation for low power IoT devices. In *2018 IEEE 4th International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing, (HPSC) and IEEE International Conference on Intelligent Data and Security (IDS)(BIGDATASECURITY/HPSC/IDS)*, pages 194–203, May 2018. doi: 10.1109/BDS/HPSC/IDS18.2018.00050. URL `doi.ieeecomputersociety.org/10.1109/BDS/HPSC/IDS18.2018.00050`. [**BEST PAPER**].

[130] Jason M. McGinthy and Alan J. Michaels. Semi-coherent transmission security for low power IoT devices. In *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 170–177, July 2018. doi: 10.1109/Cybermatics_2018.2018.00059.

[131] Jason M. McGinthy and Alan. J. Michaels. Secure industrial internet of things critical infrastructure node design. *IEEE Internet of Things Journal*, 2019. ISSN 2327-4662. doi: 10.1109/JIOT.2019.2903242.

[132] John M McQuillan and David C Walden. The arpa network design decisions. *Computer Networks (1976)*, 1(5):243–289, 1977.

[133] K. Merchant, S. Revay, G. Stantchev, and B. Nousain. Deep learning for RF device fingerprinting in cognitive communication networks. *IEEE Journal of Selected Topics in Signal Processing*, 12(1):160–167, Feb 2018. ISSN 1932-4553. doi: 10.1109/JSTSP.2018.2796446.

[134] A. J. Michaels. High-order PSK signaling (HOPS) techniques for low-power spread spectrum communications. In *2018 IEEE 19th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*, pages 1–7, June 2018. doi: 10.1109/WoWMoM.2018.8449732.

[135] A. J. Michaels and D. B. Chester. Featureless chaotic spread spectrum modulation of arbitrary data constellations. In *2011 IEEE 12th International Workshop on Signal Processing Advances in Wireless Communications*, pages 36–40, June 2011. doi: 10.1109/SPAWC.2011.5990432.

[136] A. J. Michaels, M. Meadows, and J. Ernst. PRNG sequence combination techniques via Galois extension fields. In *MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM)*, pages 841–845, Oct 2017. doi: 10.1109/MILCOM.2017.8170729.

[137] A.J. Michaels and D.B. Chester. Mixed radix conversion with a priori defined statistical artifacts, 2011. US Patent 7,970,809.

[138] Alan J Michaels. *Digital chaotic communications*. Georgia Institute of Technology, 2010.

[139] Alan J Michaels. A maximal entropy digital chaotic circuit. In *Circuits and Systems (ISCAS), 2011 IEEE International Symposium on*, pages 717–720. IEEE, 2011.

[140] R. Fernandez Molanes, K. Amarasinghe, J. Rodriguez-Andina, and M. Manic. Deep learning and reconfigurable platforms in the internet of things: Challenges and opportunities in algorithms and hardware. *IEEE Industrial Electronics Magazine*, 12(2):36–49, June 2018. ISSN 1932-4529. doi: 10.1109/MIE.2018.2824843.

[141] Kathleen Moriarty, Burt Kaliski, and Andreas Rusch. PKCS# 5: Password-based cryptography specification version 2.1. 2017.

[142] A. Mosenia and N. K. Jha. A Comprehensive Study of Security of Internet-of-Things. *IEEE Transactions on Emerging Topics in Computing*, 5(4):586–602, October 2017. doi: 10.1109/TETC.2016.2606384.

[143] D. Mukhopadhyay, S. Banerjee, D. RoyChowdhury, and B. B. Bhattacharya. CryptoScan: A secured scan chain architecture. In *14th Asian Test Symposium (ATS'05)*, pages 348–353, Dec 2005. doi: 10.1109/ATS.2005.42.

[144] Lindsey O'Donnell. IoT Security Concerns Peaking – With No End In Sight. URL `https://threatpost.com/iot-security-concerns-peaking-with-no-end-in-sight/131308/`.

[145] O. Olawumi, K. Haataja, M. Asikainen, N. Vidgren, and P. Toivanen. Three practical attacks against ZigBee security: Attack scenario definitions, practical experiments, countermeasures, and lessons learned. In *2014 14th International Conference on Hybrid Intelligent Systems*, pages 199–206, December 2014. doi: 10.1109/HIS.2014.7086198.

[146] John Padgette. Guide to Bluetooth security. *NIST Special Publication*, 800:121, 2017.

[147] C. F. Pasluosta, H. Gassner, J. Winkler, J. Klucken, and B. M. Eskofier. An emerging era in the management of parkinson's disease: Wearable technologies and the internet of things. *IEEE Journal of Biomedical and Health Informatics*, 19(6):1873–1881, Nov 2015. ISSN 2168-2194. doi: 10.1109/JBHI.2015.2461555.

[148] H. J. Patel, M. A. Temple, and R. O. Baldwin. Improving ZigBee device network authentication using ensemble decision tree classifiers with radio frequency distinct native attribute fingerprinting. *IEEE Transactions on Reliability*, 64(1):221–233, March 2015. ISSN 0018-9529. doi: 10.1109/TR.2014.2372432.

[149] Goutam Paul and Subhamoy Maitra. Permutation after RC4 key scheduling reveals the secret key. In *International Workshop on Selected Areas in Cryptography*, pages 360–377. Springer, 2007.

[150] E. E. Petrosky, A. J. Michaels, and D. B. Ridge. Network scalability comparison of ieee 802.15.4 and receiver-assigned cdma. *IEEE Internet of Things Journal*, 2019. ISSN 2327-4662. doi: 10.1109/JIOT.2018.2884455.

[151] Eric E. Petrosky, Alan J. Michaels, and Joesph M. Ernst. A low power IoT medium access control for receiver-assigned CDMA. In *2018 Wireless Telecommunications Symposium (WTS)*, April 2018.

[152] Benjamin M. Piccarreta. Draft NISTIR 8200, Interagency Report on Status of International Cybersecurity Standardization for the Internet of Things (IoT). page 187.

[153] Tim Polk, Kerry McKay, and Santosh Chokhani. Guidelines for the selection, configuration, and use of transport layer security (TLS) implementations. *NIST special publication*, 800(52):32, 2014.

[154] H Vincent Poor. *An introduction to signal detection and estimation*. Springer Science & Business Media, 2013.

[155] Andrey Popov. Prohibiting RC4 cipher suites. *Computer Science*, 2355:152–164, 2015.

[156] T. Popp, S. Mangard, and E. Oswald. Power analysis attacks and countermeasures. *IEEE Design Test of Computers*, 24(6):535–543, Nov 2007. ISSN 0740-7475. doi: 10.1109/MDT.2007.200.

[157] PPD-21. Presidential policy directive: Critical infrastructure security and resilience. 2013.

[158] T. Pultarova. Webcam hack shows vulnerability of connected devices. *Engineering Technology*, 11(11):10–10, December 2016. ISSN 1750-9637. doi: 10.1049/et.2016.1112.

[159] Z. Quan, F. Gui, D. Xiao, and Y. Tang. Trusted architecture for farmland wireless sensor networks. In *4th IEEE International Conference on Cloud Computing Technology and Science Proceedings*, pages 782–787, December 2012. doi: 10.1109/CloudCom.2012.6427496.

[160] M. T. Rahman, F. Rahman, D. Forte, and M. Tehranipoor. An aging-resistant RO-PUF for reliable key generation. *IEEE Transactions on Emerging Topics in Computing*, 4(3):335–348, July 2016. ISSN 2168-6750. doi: 10.1109/TETC.2015.2474741.

[161] S. Ur Rehman, K. Sowerby, C. Coghill, and W. Holmes. The analysis of rf fingerprinting for low-end wireless receivers with application to ieee 802.11a. In *2012 International Conference on Selected Topics in Mobile and Wireless Networking*, pages 24–29, July 2012. doi: 10.1109/iCOST.2012.6271285.

[162] K. A. Remley, C. A. Grosvenor, R. T. Johnk, D. R. Novotny, P. D. Hale, M. D. McKinley, A. Karygiannis, and E. Antonakakis. Electromagnetic signatures of wlan cards and network security. In *Proceedings of the Fifth IEEE International Symposium on Signal Processing and Information Technology, 2005.*, pages 484–488, Dec 2005. doi: 10.1109/ISSPIT.2005.1577145.

[163] Eric Rescorla. The transport layer security (TLS) protocol version 1.3. URL https://tools.ietf.org/html/draft-ietf-tls-tls13-26.

[164] Eric Rescorla and Nagendra Modadugu. Datagram transport layer security version 1.2. 2012.

[165] Matthew Robshaw and Olivier Billet, editors. *New Stream Cipher Designs: The eSTREAM Finalists*. Springer-Verlag, Berlin, Heidelberg, 2008. ISBN 978-3-540-68350-6.

[166] E. Ronen, A. Shamir, A. Weingarten, and C. O'Flynn. IoT goes nuclear: Creating a ZigBee chain reaction. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 195–212, May 2017. doi: 10.1109/SP.2017.14.

[167] K.H. Rosen. *Elementary Number Theory and Its Applications*. Addison-Wesley, 2011. ISBN 9780321500311.

[168] Ron Ross, Patrick Viscuso, Gary Guissanie, KELLEY DEMPSEY, and Mark Riddle. Protecting controlled unclassified information in nonfederal systems and organizations. *NIST Special Publication*, 800:171, 2016.

[169] Ishtiaq Rouf, Rob Miller, Hossen Mustafa, Travis Taylor, Sangho Oh, Wenyuan Xu, Marco Gruteser, Wade Trappe, and Ivan Seskar. Security and privacy vulnerabilities of in-car wireless networks: a tire pressure monitoring system case study. In *Proceedings of the 19th USENIX conference on Security*, pages 21–21. USENIX Association, 2010.

[170] A. De Rubertis, L. Mainetti, V. Mighali, L. Patrono, I. Sergi, M. L. Stefanizzi, and S. Pascali. Performance evaluation of end-to-end security protocols in an internet of things. In *2013 21st International Conference on Software, Telecommunications and Computer Networks - (SoftCOM 2013)*, pages 1–6, Sept 2013. doi: 10.1109/SoftCOM.2013.6671893.

[171] Markku Juhani Saarinen and Jean-Philippe Aumasson. The BLAKE2 cryptographic hash and message authentication code (MAC). RFC 7693, November 2015.

[172] Dilip V Sarwate and Michael B Pursley. Crosscorrelation properties of pseudorandom and related sequences. *Proceedings of the IEEE*, 68(5):593–619, 1980.

[173] P. Scanlon, I. O. Kennedy, and Y. Liu. Feature extraction approaches to RF fingerprinting for device identification in femtocells. *Bell Labs Technical Journal*, 15(3):141–151, Dec 2010. ISSN 1089-7089. doi: 10.1002/bltj.20462.

[174] Karen Seo and Stephen Kent. Security architecture for the internet protocol. 2005.

[175] Claude E Shannon. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55, 2001.

[176] Cong Shi, Jian Liu, Hongbo Liu, and Yingying Chen. Smart user authentication through actuation of daily activities leveraging WiFi-enabled IoT. In *Proceedings of the 18th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, Mobihoc '17, pages 5:1–5:10, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4912-3. doi: 10.1145/3084041.3084061. URL http://doi.acm.org/10.1145/3084041.3084061.

[177] Taizo Shirai, Kyoji Shibutani, Toru Akishita, Shiho Moriai, and Tetsu Iwata. The 128-bit blockcipher CLEFIA. In *Proceedings of the 14th International Conference on Fast Software Encryption*, FSE'07, pages 181–195, Berlin, Heidelberg, 2007. Springer-Verlag. ISBN 3-540-74617-X, 978-3-540-74617-1. URL http://dl.acm.org/citation.cfm?id=2394499.2394516.

[178] Bernard Sklar. *Digital communications*, volume 2. Prentice Hall Upper Saddle River, 2001.

[179] C. Song, Y. Zhan, and L. Guo. Specific emitter identification based on intrinsic time-scale decomposition. In *2010 6th International Conference on Wireless Communications Networking and Mobile Computing (WiCOM)*, pages 1–4, Sept 2010. doi: 10.1109/WICOM.2010.5600772.

[180] C. Sperandio and P. G. Flikkema. Wireless physical-layer security via transmit precoding over dispersive channels: optimum linear eavesdropping. In *MILCOM 2002. Proceedings*, volume 2, pages 1113–1117 vol.2, Oct 2002. doi: 10.1109/MILCOM.2002.1179633.

[181] William Stallings. *Cryptography and Network Security: Principles and Practice*. Pearson Education, 7th edition, 2017. ISBN 0134444280.

[182] Kevin T Sterne, Joseph M Ernst, Deirdre K Kilcoyne, Alan J Michaels, and Geffrey Moy. Tire pressure monitoring system sensor radio frequency measurements for privacy concerns. Technical report, 2017.

[183] G. E. Suh and S. Devadas. Physical unclonable functions for device authentication and secret key generation. In *2007 44th ACM/IEEE Design Automation Conference*, pages 9–14, June 2007.

[184] B. Sunar, W. J. Martin, and D. R. Stinson. A provably secure true random number generator with built-in tolerance to active attacks. *IEEE Transactions on Computers*, 56(1):109–119, Jan 2007. ISSN 0018-9340. doi: 10.1109/TC.2007.250627.

[185] Charles Suslowicz, Archanaa S. Krishnan, and Patrick Schaumont. Optimizing cryptography in energy harvesting applications. In *Proceedings of the 2017 Workshop on Attacks and Solutions in Hardware Security*, ASHES '17, pages 17–26, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-5397-7. doi: 10.1145/3139324.3139329. URL http://doi.acm.org/10.1145/3139324.3139329.

[186] R. Tahir, H. Hu, D. Gu, K. McDonald-Maier, and G. Howells. A scheme for the generation of strong icmetrics based session key pairs for secure embedded system applications. In *2013 27th International Conference on Advanced Information Networking and Applications Workshops*, pages 689–696, March 2013. doi: 10.1109/WAINA.2013.143.

[187] Kenneth I Talbot, Paul R Duley, and Martin H Hyatt. Specific emitter identification and verification. *Technology Review*, 113, 2003.

[188] *MSP430FR58xx, MSP430FR59xx, and MSP430FR6xx Family User Guide*. Texas Instruments, June 2017.

[189] *MSP432P4xx SimpleLink^{TM} Microcontrollers*. Texas Instruments, December 2017.

[190] Andrew Thangaraj, Souvik Dihidar, A Robert Calderbank, Steven McLaughlin, and Jean-Marc Merolla. Capacity achieving codes for the wiretap channel with applications to quantum key distribution. *arXiv preprint cs.IT/0411003*, 2004.

[191] F. Tobagi and L. Kleinrock. Packet switching in radio channels: Part II - the hidden terminal problem in carrier sense multiple-access and the busy-tone solution. *IEEE Transactions on Communications*, 23(12):1417–1433, December 1975. ISSN 0090-6778. doi: 10.1109/TCOM.1975.1092767.

[192] J. Toonstra and W. Kinsner. A radio transmitter fingerprinting system ODO-1. In *Proceedings of 1996 Canadian Conference on Electrical and Computer Engineering*, volume 1, pages 60–63 vol.1, May 1996. doi: 10.1109/CCECE.1996.548038.

[193] Meltem Sönmez Turan, Elaine Barker, William Burr, and Lily Chen. Recommendation for password-based key derivation. *NIST special publication*, 800:132, 2010.

[194] Mathy Vanhoef and Frank Piessens. Practical verification of WPA-TKIP vulnerabilities. In *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security*, ASIA CCS '13, pages 427–436, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-1767-2. doi: 10.1145/2484313.2484368. URL http://doi.acm.org/10.1145/2484313.2484368.

[195] Mathy Vanhoef and Frank Piessens. Key reinstallation attacks: Forcing nonce reuse in WPA2. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1313–1328. ACM, 2017.

[196] Andrew J Viterbi. *CDMA: principles of spread spectrum communication*, volume 122.

[197] Paul Voigt and Axel Von dem Bussche. The EU general data protection regulation (GDPR). *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, 2017.

[198] Yongge Wang. The law of the iterated logarithm for random sequences. In *Proceedings of Computational Complexity (Formerly Structure in Complexity Theory)*, pages 180–189, May 1996. doi: 10.1109/CCC.1996.507680.

[199] Yongge Wang and Tony Nicol. On statistical distance based testing of pseudo random sequences and experiments with PHP and Debian OpenSSL. *Computers & Security*, 53:44 – 64, 2015. ISSN 0167-4048. doi: https://doi.org/10.1016/j.cose.2015.05.005. URL http://www.sciencedirect.com/science/article/pii/S0167404815000693.

[200] K. Wold and S. Petrovic. Optimizing speed of a true random number generator in FPGA by spectral analysis. In *2009 Fourth International Conference on Computer Sciences and Convergence Information Technology*, pages 1105–1110, Nov 2009. doi: 10.1109/ICCIT.2009.95.

[201] M. Wollschlaeger, T. Sauter, and J. Jasperneite. The future of industrial communication: Automation networks in the era of the internet of things and industry 4.0. *IEEE Industrial Electronics Magazine*, 11(1):17–27, March 2017. ISSN 1932-4529. doi: 10.1109/MIE.2017.2649104.

[202] L. J. Wong, W. C. Headley, S. Andrews, R. M. Gerdes, and A. J. Michaels. Clustering learned cnn features from raw I/Q data for emitter identification. In *MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM)*, Oct 2018.

[203] L. J. Wong, W. C. Headley, and A. J. Michaels. Estimation of transmitter I/Q imbalance using convolutional neural networks. In *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 948–955, Jan 2018. doi: 10.1109/CCWC.2018.8301715.

[204] L. J. Wong, W. C. Headley, and A. J. Michaels. Specific emitter identification using convolutional neural network-based IQ imbalance estimators. *IEEE Access*, 7:33544–33555, 2019. ISSN 2169-3536. doi: 10.1109/ACCESS.2019.2903444.

[205] A. D. Wood and J. A. Stankovic. Denial of service in sensor networks. *Computer*, 35 (10):54–62, Oct 2002. ISSN 0018-9162. doi: 10.1109/MC.2002.1039518.

[206] Joshua Wright. Killerbee: practical zigbee exploitation framework. In *11th ToorCon conference, San Diego*, volume 67, 2009.

[207] A. D. Wyner. The wire-tap channel. *Bell System Technical Journal*, 54(8):1355–1387, 1975. ISSN 1538-7305. doi: 10.1002/j.1538-7305.1975.tb02040.x. URL http://dx.doi.org/10.1002/j.1538-7305.1975.tb02040.x.

[208] L. Xiao, Q. Yan, W. Lou, G. Chen, and Y. T. Hou. Proximity-based security techniques for mobile users in wireless networks. *IEEE Transactions on Information Forensics and Security*, 8(12):2089–2100, Dec 2013. ISSN 1556-6013. doi: 10.1109/TIFS.2013.2286269.

[209] M. Xiao, J. Wu, and L. Huang. Time-sensitive utility-based single-copy routing in low-duty-cycle wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 26(5):1452–1465, May 2015. ISSN 1045-9219. doi: 10.1109/TPDS.2014.2321136.

[210] L. D. Xu, W. He, and S. Li. Internet of things in industries: A survey. *IEEE Transactions on Industrial Informatics*, 10(4):2233–2243, Nov 2014. ISSN 1551-3203. doi: 10.1109/TII.2014.2300753.

[211] Q. Xu, R. Zheng, W. Saad, and Z. Han. Device fingerprinting in wireless networks: Challenges and opportunities. *IEEE Communications Surveys Tutorials*, 18(1):94–104, Firstquarter 2016. ISSN 1553-877X. doi: 10.1109/COMST.2015.2476338.

[212] Y. Yang, L. Wu, G. Yin, L. Li, and H. Zhao. A survey on security and privacy issues in internet-of-things. *IEEE Internet of Things Journal*, 4(5):1250–1258, Oct 2017. ISSN 2327-4662. doi: 10.1109/JIOT.2017.2694844.

[213] H. M. Yassine and W. R. Moore. Improved mixed-radix conversion for residue number system architectures. *IEE Proceedings G - Circuits, Devices and Systems*, 138(1):120–124, Feb 1991. ISSN 0956-3768. doi: 10.1049/ip-g-2.1991.0022.

[214] C. E. Yin and G. Qu. Improving puf security with regression-based distiller. In *2013 50th ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–6, May 2013.

[215] M. D. Yu, M. Hiller, J. Delvaux, R. Sowell, S. Devadas, and I. Verbauwhede. A lockdown technique to prevent machine learning on PUFs for lightweight authentication. *IEEE Transactions on Multi-Scale Computing Systems*, 2(3):146–159, July 2016. doi: 10.1109/TMSCS.2016.2553027.

[216] H. L. Yuan and A. Q. Hu. Preamble-based detection of Wi-Fi transmitter RF fingerprints. *Electronics Letters*, 46(16):1165–1167, August 2010. ISSN 0013-5194. doi: 10.1049/el.2010.1220.

[217] K. Zeng, C. H. Yang, D. Y. Wei, and T. R. N. Rao. Pseudorandom bit generators in stream-cipher cryptography. *Computer*, 24(2):8–17, Feb 1991. ISSN 0018-9162. doi: 10.1109/2.67207.

[218] Z. Zhang, M. C. Y. Cho, C. Wang, C. Hsu, C. Chen, and S. Shieh. Iot security: Ongoing challenges and research opportunities. In *2014 IEEE 7th International Conference on Service-Oriented Computing and Applications*, pages 230–234, Nov 2014. doi: 10.1109/SOCA.2014.58.

[219] Y. Zou, J. Zhu, X. Wang, and V. C. M. Leung. Improving physical-layer security in wireless communications using diversity techniques. *IEEE Network*, 29(1):42–48, Jan 2015. ISSN 0890-8044. doi: 10.1109/MNET.2015.7018202.