APPLICATION OF SPRITZ ENCRYPTION FOR IMPROVING CYBER SECURITY AND PRIVACY FOR ELECTRICAL SMART METERS

LINCOLN KAMAU KIARIE

MASTER OF SCIENCE

(Telecommunication Engineering)

JOMO KENYATTA UNIVERSITY OF

AGRICULTURE AND TECHNOLOGY

2019

Application of Spritz Encryption for Improving Cyber Security and Privacy for Electrical Smart Meters

LINCOLN KAMAU KIARIE

A thesis submitted in partial fulfilment for the Degree of Master of Science in Telecommunication Engineering in the Jomo Kenyatta University of Agriculture and Technology

2019

DECLARATION

This thesis is my original work and has not been presented for the award of a degree in any other university.

Signature: ..... Date: .....

Lincoln Kamau Kiarie

This thesis has been submitted for examination with our approval as university supervisors:

Signature: ..... Date: .....

Dr Kibet Langat, PhD

JKUAT, Kenya

Signature: ..... Date: .....

Prof. Christopher Maina, PhD

Murang'a University of Technology, Kenya

DEDICATION

To my dear wife Luseno – you constantly encourage me to do better than I thought possible.

To Lemuel, Luke and Lennox – you bring me great joy with your curiosity and energy.

ACKNOWLEDGEMENTS

The saying that it takes a village to raise a child could, to some extent, also apply to completing a thesis. Many people have made direct and indirect contributions.

My supervisors were of immense help. Dr Kibet set aside time to guide me in the middle of demanding work schedules and invited me to helpful research presentations. Prof. Maina's patience and optimism in the face of challenges helped me keep pushing forward even when I was stuck. You are both great role models to me.

I am grateful for my colleague Lawrence Chege introducing me to the fascinating field of Cyber Security which is at the heart of this work. I also appreciate Dr. Ron Rivest, a pioneer and trailblazer in modern Cryptography, whose algorithms were helpful in this work.

To my fellow postgraduate students – especially Roy Orenge and Louis Njoroge– for much needed feedback, support and pointing me to useful research tools, thank you.

My parents James and Esther Kiarie have made many sacrifices through the years for my education. I am grateful for the prayers and that through your examples I am reminded of the vital need to work hard in order to be helpful to others.

I appreciate my wife, Luseno, for being there through the good and the bad, in times of progress and when everything seemed to have stalled. Thank you for loving me well and taking care of our home so that I am better placed to research. I would not be where I am without you.

Finally, I thank my Lord and Saviour, Jesus Christ, who pulled me "out of the miry clay, and set my feet upon a rock" and without whom all my efforts would be striving after the wind.

TABLE OF CONTENTS

DECLARATION ..... **ii**

DEDICATION ..... **iii**

ACKNOWLEDGEMENTS ..... **iv**

LIST OF TABLES ..... **x**

LIST OF FIGURES ..... **xi**

LIST OF APPENDICES ..... **xiii**

LIST OF ACRONYMS ..... **xiv**

ABSTRACT ..... **xvi**

CHAPTER ONE **1**

INTRODUCTION **1**

1.1 Background 1

1.2 Problem Statement 3

1.3 Justification 3

1.4 Objectives 4

1.4.1 General Objective 4

1.4.2 Specific Objectives 4

1.5 Thesis Organization 5

CHAPTER TWO 6

LITERATURE REVIEW 6

2.1 Introduction 6

2.2 Smart Grid 7

2.2.1 Features and benefits of Smart Grid 9

2.2.2 Smart Grid Communication Requirements 12

2.2.3 Challenges and Barriers to Smart Grid 13

2.3 Smart Grid Communication 14

2.3.1 Communications Options for Smart Grid 14

2.3.2 Smart Grid data Compared to Internet Traffic 15

2.3.3 Challenges in Smart Grid Communications 18

2.3.4 Use Cases in Smart Grid Communication 20

2.3.5 Smart Meters 23

2.4 Cyber Security and Privacy 25

2.4.1 Security objectives and requirements 26

2.4.2 Types of security attacks....28

2.4.3 Privacy Concerns in Smart Grid....30

2.4.4 Solutions to Privacy Leakage....33

2.5 Encryption....35

2.5.1 Symmetric versus Asymmetric Ciphers....38

2.5.2 Block versus Stream Ciphers....39

2.5.3 Key Selection....41

2.5.4 RC4 Encryption....44

2.5.5 Spritz Encryption....45

2.5.6 Qualitative improvements of Spritz over RC4....46

2.5.7 Statistical tests for Cryptography....46

2.6 Summary of Literature survey....48

CHAPTER THREE....50

METHODOLOGY....50

3.1 Problem formulation....50

3.2 Extraction of data for Encryption using Image Processing....52

3.3 Encryption of smart meter data....54

3.3.1 Key Selection and analysis 54

3.3.2 RC4 Encryption/Decryption 55

3.3.3 Spritz Encryption/Decryption 56

3.3.4 Attempted decryption using wrong keys 57

3.4 Comparing effectiveness of Spritz against RC4 58

3.4.1 Time taken to run RC4 and Spritz 58

3.4.2 Statistical analysis of RC4 and Spritz 58

3.4.3 Attacks on attenuated Ciphers 60

CHAPTER FOUR 62

RESULTS AND ANALYSIS 62

4.1 Data extracted using Image Processing 62

4.2 Results for Encryption 63

4.2.1 Key Selection and analysis 63

4.2.2 RC4 Encryption/Decryption 65

4.2.3 Spritz Encryption/Decryption 66

4.2.4 Attempted decryption using wrong keys 68

4.2.5 Time Taken for RC4 and Spritz Encryption 73

4.3 Results for Statistical Analysis of RC4 and Spritz 74

4.3.1 Monobit test 75

4.3.2 Poker test.....75

4.3.3 Runs test.....75

4.3.4 Long run test 79

4.3.5 Conclusion from Statistical Test results 79

4.4 Attacks on attenuated Ciphers 80

CHAPTER FIVE 82

CONCLUSION AND RECOMMENDATIONS.....82

5.1 Conclusion 82

5.2 Thesis Contributions 83

5.3 Recommendations 84

REFERENCES 86

APPENDICES.....92

LIST OF TABLES

Table 2.1: Domains and Actors in the NIST SG Conceptual Model..... 8
Table 2.2: A brief comparison of the conventional grid and the smart grid. 12
Table 2.3: Summary of differences between smart grid and internet traffic.....17
Table 2.4: Comparison of various privacy enhancing techniques 35
Table 2.5: Estimated time for successful brute-force attacks on different key lengths ..37
Table 2.6: Main differences between symmetric and asymmetric ciphers 39
Table 2.7: Range of intervals for passing the runs test 48
Table 4.1: Test data points for RC4 and Spritz 67
Table 4.2: Wrong keys used to attempt decryption 69
Table 4.3: Time taken for RC4 encryption on 10 repeated trials 74
Table 4.4: Time taken for Spritz encryption on 10 repeated trials 74
Table 4.5: Sample bits for testing RC4 and Spritz 74
Table 4.6: Results for RC4 runs test 75
Table 4.7: Results for Spritz runs test 77

LIST OF FIGURES

Figure 2.1: Interaction of Actors in Different Smart Grid Domains 8
Figure 2.2: Redistribution of load from peak to off-peak hours 10
Figure 2.3: Household profile with devices identified. 20
Figure 2.4: Use cases in Smart Grid distribution and transmission systems 22
Figure 2.5: Use cases in the AMI and home-area networks 23
Figure 2.6: (a) Conventional energy meter and (b) smart meter architecture.....24
Figure 2.7: Household electricity demand profile from a one-bedroom apartment 26
Figure 2.8: Power consumption over time showing different appliance events 31
Figure 2.9: Battery-based Load Hiding Algorithm 33
Figure 2.10: Block diagram of Symmetric Encryption.....36
Figure 2.11: A summary of classification of Encryption Techniques 40
Figure 2.12: Block Diagram showing the operation of a stream cipher 41
Figure 2.13: Block diagram describing a general hash function, H 43
Figure 3.1: Household profile with devices identified 52
Figure 3.2: Applying RC4 to electrical usage data.....56
Figure 3.3: Applying Spritz to electrical usage data.....57
Figure 3.4: Flow chart for runs test and long run test.....60
Figure 4.1: Typical household energy usage, plotted from extracting data points.....62
Figure 4.2: Percentage variation in hamming distance as meter number increases 64
Figure 4.3: RC4 Encrypted data using *Key1* 65
Figure 4.4: Data restored through RC4 decryption using *Key1* 66
Figure 4.5: Spritz encryption using *Key1*.....67
Figure 4.6: Data restored through Spritz decryption using *Key1* 68
Figure 4.7: Attempted Spritz Decryption using *Key2*.....70
Figure 4.8: Attempted Spritz Decryption using *Key3*.....70
Figure 4.9: Attempted Spritz Decryption using *Key4*.....71
Figure 4.10: Attempted RC4 Decryption using *Key2*.....72

Figure 4.11: Attempted RC4 Decryption using *Key3*.....72
Figure 4.12: Attempted RC4 Decryption using *Key4*.....73
Figure 4.13: Results from 0s runs for RC4.....76
Figure 4.14: Results from 1s runs for RC4.....77
Figure 4.15: Results from 0s runs for Spritz.....78
Figure 4.16: Results from 1s runs for Spritz.....79
Figure 4.17: Comparison of time taken to break RC4 and Spritz using brute force.....81

LIST OF APPENDICES

Appendix 1: RC4 Algorithm and code 92
Appendix 2: Spritz Algorithm and code 95
Appendix 3: Code for extracting data points from an Image graph 102
Appendix 4: Miscellaneous Code.....104
Appendix 5: Conferences and Publications from this work.....109

LIST OF ACRONYMS

AMI	Advanced Metering Infrastructure
AMR	Automatic Meter Reading
AES	Advanced Encryption Standard
BEMS	Building Energy Management System
BLH	Battery-based Load Hiding
DES	Data Encryption Standard
DMS	Distribution Management Systems
DoS	Denial-of-service
DR	Demand Response
DRM	Digital Rights Management
DSL	Digital Subscriber Line
DSM	Demand Side Management
EMS	Energy Management System
HEMS	Home Energy Management System
ICT	Information and Communications Technology
IED	Intelligent Electronic Devices
IEEE	Institute of Electrical and Electronic Engineers
LAN	Local Area Network
MDMS	Meter Data Management System
NALM/NIALM	Nonintrusive Appliance Load Monitoring
NIST	National Institute of Science and Technology
OSGP	Open Smart Grid Protocol
PHEV	Plug-in Hybrid Electrical Vehicles
PLC	Power Line Communication
PMU	Phasor Measurement Units
PV	Photovoltaic
QoS	Quality of Service

RC4	Rivest Cipher 4 (or Ron's Code 4)
RSA	Rivest-Shamir-Adleman
SCADA	Supervisory Control and Data Acquisition
SDH	Synchronous Data Hierarchy
SHA	Secure Hash Algorithm
SONET	Synchronous Optical Network
ToU	Time of Use
TLS	Transport Layer Security
WEP	Wired Equivalent Privacy
WPA	Wi-Fi Protected Access

ABSTRACT

The electrical grid has been undergoing improvements by integrating advanced communication to convert it to a better system known as Smart Grid. Better communication makes Smart Grid better, but it also makes it vulnerable to problems associated to telecommunications. This thesis aims to reduce the problem of privacy leakage and cyber security in electrical smart meters by applying fast and secure cryptography. Smart meters have limited computational capacity, thus care is needed when choosing an algorithm to use, without compromising on security. Researchers have found severe weaknesses in the first version of Open Smart Grid Protocol (OSGP) which used the popular *RC4* (Rivest Cipher 4) encryption algorithm to secure smart meters. With OSGP being broken, there is need for a more secure encryption algorithm for smart meters. In this thesis, *Spritz*, an encryption algorithm developed in 2014, was examined and found to have suitable properties for smart meters. It offers good security while introducing fairly low computational overhead. Encryption was tested using data from a typical household over a 24-hour period. This data was extracted from a graph using image processing. After encryption, a plot of the data showed that it appeared random and thus obscuring a customer's usage patterns. Data privacy was thus achieved since data mining techniques can no longer work. Spritz was compared to RC4 in terms of performance speed and found to be slower by a small margin. Encryption with Spritz took on average 0.851 milliseconds (compared to 0.449ms for RC4), which would not adversely affect the operation of a smart meter, being less than 2 times slower. A key generation method is also presented for creating cryptographic keys from meter numbers. A test of 100 keys from consecutive meter numbers revealed that adjacent keys had, on average, about half the bits being different bits (49.39%) being different. To compare the strength of Spritz to RC4, attacks were mounted against weakened versions of the ciphers. The time taken to break Spritz was found to be considerably higher than RC4. For an 8 bit key, it took over 3 times longer to break Spritz than RC4. Thus although Spritz is slightly slower, it is significantly more secure than RC4 and is thus a better candidate for smart meter encryption.

CHAPTER ONE

INTRODUCTION

1.1 Background

As Kenya seeks to attain “Vision 2030”, growth in development and population will keep pushing the demand for electricity higher and higher. The cost of meeting rising energy demand is a challenge. Merely increasing power generation plants is not a sustainable option, both in terms of cost and efficiency. Improving the processes of generation, transmission and distribution is essential in attaining this goal. This has been the subject of much research and recent advances in field of electrical power, especially by applying telecommunications. This is leading to the gradual replacement of the traditional power grid by a more efficient system known as the *Smart Grid*.

Smart Grid has two-way flow of power and information giving it several advantages over the traditional grid. Improved two way communication benefits both power utility companies and customers. Smart Grid offers better fault detection and response to problems occurring in the system. It facilitates distributed power generation in which there are many power producers instead of having only a few centralized locations, which can be crucial points of failure. Operational costs are reduced through automation of meter readings. It contains many sensors and is thus self-monitoring and self-healing. Customers receive better services, fewer blackouts and have more options to choose from [1]. They also enjoy better feedback, visualization of electricity usage and potentially lower power bills.

Although advanced communication makes Smart Grid superior, it is also one of its great sources of vulnerability. Having more data is very helpful in monitoring, planning power distribution and sending control signals, but it also introduces privacy and cyber security threats. A user’s consumption data can become a major source of *privacy* leakage. From

this data, it is possible to infer the electrical devices being used in a home or industry. This data can then be used to find out when a home is empty, which processes are used in a factory and other details which may be private. Thus robbers, business competitors and any malicious observer can use this data to the detriment of any electrical user.

Further problems occur in the form of *cyber security* threats. A malicious hacker can manipulate control signals in the power network causing system malfunctions such as blackouts. Pricing or billing information can also be changed, causing the utility company to suffer huge losses.

In telecommunication systems, cyber security and privacy problems are tackled through the use of cryptography. Cryptography involves changing of messages to a form in which they are unintelligible to anyone other than the intended recipient(s) and/or verifying that the correct messages are received from the right sender. It would at first appear to be a trivial matter to take cryptographic methods in telecommunications and apply them directly to Smart Grid, but there are two barriers to this approach. First, cryptography is generally computationally intensive. This increases the minimum hardware requirements needed to provide necessary computing power and storage. Smart Grid has many light-weight devices designed for large scale operations, including smart meters. Some cryptographic approaches may adversely affect the operation of these devices or cause them to stop working altogether. Second, some sections of smart grid have very little time delay allowances (i.e. latency), beyond which system errors occur and some functionality is lost. Cryptographic approaches with long processing times are thus unsuitable. It is thus highly desirable to assess the trade-off between security and time-criticality in order to adopt appropriate security schemes [2].

This work seeks to address the problem of privacy leakage in electrical smart meters. By considering various options for encryption, a relatively new technique called Spritz [3] was examined and found to be a suitable candidate for smart meter encryption.

1.2 Problem Statement

Smart meters collect and transmit usage data at a higher rate than previous meters. This provides accurate readings that can help improve energy efficiency, but it comes at the cost of user privacy. Data being sent can be illegally accessed, intercepted, modified, or used against the wishes of the electrical customer. It can then be analysed, resulting in the extraction of information about a household's activities or processes occurring in a manufacturing plant.

For example, a burglar would know what electrical devices are in a home and can determine if a house is occupied with an accuracy of over 80% [4]. With such information, robbers would figure out the best time to invade a house, the electrical gadgets they expect to find and whether an electrical alarm has been installed. Lifestyles of home owners can be tracked without their knowledge by any interested party including spies, stalkers, landlords, totalitarian governments, and so on.

Additionally, the encryption method deployed in wide use on smart meters has been broken. In aiming to solve the problems of privacy and cyber security, a scheme known as the Open Smart Grid Protocol (OSGP) was developed by the OSGP Alliance. It used RC4 encryption for its encryption but was found to have significant weaknesses [5][6] and was thus broken. A more secure cryptographic solution is needed which can run well on smart meters, in spite of their time and resource constrains.

1.3 Justification

Privacy protection is becoming more crucial as terrorist attacks, spying and large scale surveillance increase. A vulnerable electrical power grid would have severe consequences

due to the large number of people connected to it. Data being sent between customers and a utility company needs to be protected. Encryption is thus needed.

The popular encryption algorithm, RC4, is a good candidate for securing smart meters due to its low computational overhead and relatively good security. However, it has been implemented in the Open Smart Grid Protocol (OSGP) and found to be very weak. This work examines the use of a newer encryption algorithm know as Spritz, which was developed as a "drop-in-replacement" [3] for RC4. Though it is more computationally intensive, it is designed to overcome the known weaknesses of RC4. Spritz thus has potential to work well in smart meters.

It is also desirable to have multiple valid options for encryption algorithms in any system. With this in place, a replacement can be made whenever weaknesses are discovered in the currently implemented algorithm. This is known as *algorithm agility* and is crucial because algorithms might be broken during the lifetime of a system [5]. Thus, adding Spritz to the options usable in smart meters would be beneficial.

1.4 Objectives

1.4.1 General Objective

To enhance Cyber Security and Privacy of electrical smart meters through the use of Spritz encryption.

1.4.2 Specific Objectives

i) To extract household usage data for testing encryption from an image demonstrating privacy leakage through identification of electrical devices.
ii) To evaluate the effects of encryption on smart meter data.
iii) To compare the effectiveness of Spritz against RC4 in encryption for smart meters.

iv) To simulate encryption of actual electrical consumption data using Spritz in order to demonstrate data privacy.

1.5 Thesis Organization

The rest of this thesis is organized as follows:

Chapter 2 gives a literature survey of explores Smart Grid's features and operations, its cyber security and privacy considerations and solutions, especially encryption.

Chapter 3 describes the methodology used in this research.

Chapter 4 presents the results obtained together with their analysis and discussion.

Chapter 5 gives the conclusion and recommendations of this research.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

The electrical power grid is in the process of being improved into a more intelligent and efficient system known as the Smart Grid. Traditionally, the grid has been a broadcast system with a few centralized power stations providing a wide region with electricity. Its limited communication makes it difficult to respond dynamically to faults, changes in demand and integrate renewable energy sources. It also suffers from human inefficiencies such as manual meter reading, inputting prepaid codes in digital meters, having to call customer service to report power outages and utility employees needing to follow power lines looking for the cause of a fault. Smart Grid is designed to overcome these and other limitations.

Smart Grid uses two-way communication to improve system efficiency and reliability. Actions of all connected users are integrated and power can be better transmitted. Faults can be detected and corrected faster. There is better incorporation of renewable energy because Smart Grid enables smoother reconfiguration of network topology for more efficient power flow [7]. Power generation can be harmonized better with power demand, cutting down on losses. Smart Grid also allows for automatic meter reading (AMR), eliminating the need (and cost) of having employees travel to collect usage data from customer premises. With better monitoring, utility companies are able to plan better and even offer customers improved services, including some that were not previously possible.

However, introducing advanced communications brings its own set of challenges. Questions arise on which communication system to use for various sections and how to integrate them, how to configure the network, how to ensure scalability as customers

increase, the cost of installation and maintenance as well as privacy/cyber security concerns. This work focuses on privacy and cyber security.

This chapter will look at what Smart Grid is, its benefits, requirements, and challenges. This is followed by a discussion on Smart Grid communication and then a look at cyber security and privacy threats, and together with solutions to these problems.

2.2 Smart Grid

Smart Grid is an intelligent electricity network that integrates the actions of all users connected to it. It makes use of advanced information, control, and communication technologies to save energy, reduce cost and increase reliability and transparency [8]. Other terms that have been used to describe Smart Grid shed more light on this new and improved system. They include intelligent grid, Intelligrid, Futuregrid, Intergrid, Perfect Power Grid, and EmPowered Grid [1] [9].

Smart Grid is an enhancement of the 20th century grid. It encompasses the technology for the integration, interfacing and intelligent control of various innovations. These include enabling technology such as smart metering, wind turbines, plug-in hybrid electrical vehicles (PHEV), and solar arrays.

Smart Grid involves several key players each with a different role. National Institute of Science and Technology (NIST) in USA, organizes them into 7 logical domains: Transmission, Distribution, Operations, Bulk Generation, Markets, Customer, and Service Provider [10]. They are shown in Figure 2.1.

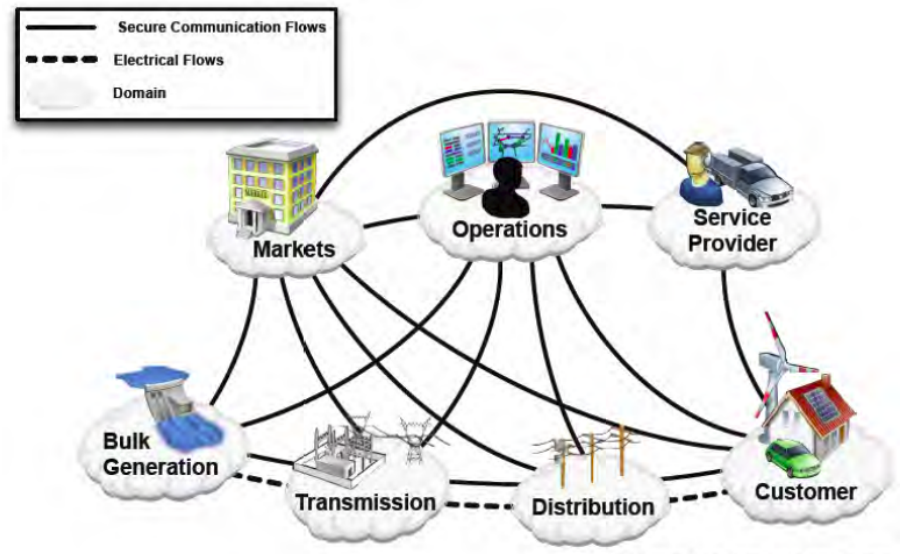


Figure 2.1 Interaction of Actors in Different Smart Grid Domains

A summary of the functions of these 7 layers is shown in Table 2.1 from [1]:

Table 2.1 Domains and Actors in the NIST SG Conceptual Model

Domain	**Actor in the Domain**
Customers	The end users of electricity. May also generate, store, and manage the use of energy.
Markets	The operators and participants in electricity markets.
Service Providers	The organizations providing services to electrical customers and utilities.
Operations	The managers of the movement of electricity.
Bulk Generation	The generators of electricity in bulk quantities. May also store energy for later distribution.
Transmission	The carriers of bulk electricity over long distances. May also generate electricity.
Distribution	The distributors of electricity to and from customers. May also store and generate electricity

2.2.1 Features and benefits of Smart Grid

Smart Grid aims to reduce the inefficiencies of the traditional electrical grid. Through improved communication, it reduces some of the problems within the traditional grid. This leads to better efficiency, lower production cost, a larger customer base and feasibility of large scale adoption of new technology (such as Electric vehicles).

As an example, consider what happens when there is a significant increase in the electricity demanded. While utility providers use load forecasting models to try predicting demand and thus aim to meet it, this is never perfectly accurate and energy demand can rise above average levels, as it happens at certain times of the day (peak hours). When the system is operating in peak load conditions, non-renewable energy sources (e.g. coal) can be relied on to generate additional supply of energy to meet a temporary shortfall in supply (lasting a few seconds up to a few minutes). This is both expensive and environmentally unfriendly. Smart Grid can reduce this by allowing demand to be matched to the available supply. This can happen by providing incentives to consumers to reschedule the load to when demand is lower (e.g. through variable pricing). This improves utilization of the available capacity. Due considerations must be made to ensure that the level of service still meets customer expectations, such as choosing which loads can be postponed [11]. Figure 2.2 from [11] shows how the peak load can be reduced, leading to cost savings.

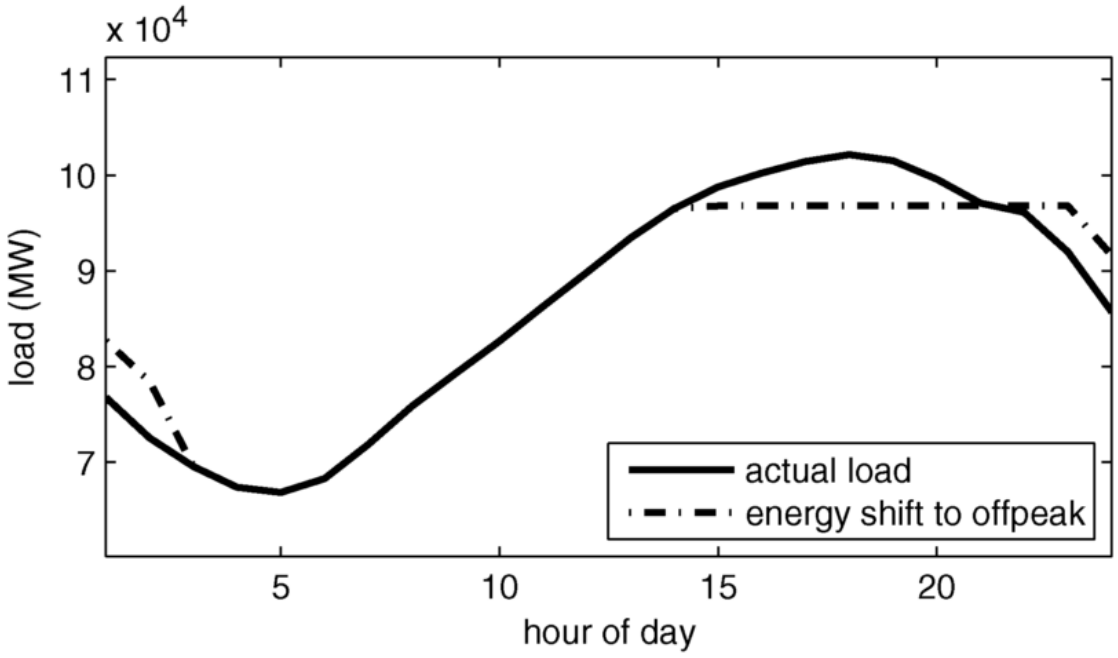


Figure 2.2 Redistribution of load from peak to off-peak hours

The benefits from Smart Grid are summarized below:

a) **Implementation of Demand Side Management:** Matching the consumption pattern with the generation pattern improves the grid's efficiency. Demand Side Management (DSM) attempts to reach this goal by providing incentive for the customer to reschedule their load [12]. This can be done by implementing Time of Use (ToU) pricing, where the charges vary at different times.
b) **Environmental conservation:** Less dependence on non-renewable energy generators leads to a reduction in environmentally harmful carbon dioxide emissions.
c) **Better distributed generation:** Improved communication allows for integration of different power generation sources, including renewable ones. Transmission losses are reduced by intelligently choosing the best supply option, increasing the efficiency.

d) **Increased potential for renewable energy:** With two-way communication, renewable energy sources can be easier incorporated to the grid, since it can allow for reconfiguration of network topology for more efficient power flow [7].

e) **Enhanced monitoring:** Power usage is measured and recorded using Advanced Metering Infrastructure (AMI). Both consumers and energy companies can access this information and use it to make better decisions. There is also wide-area monitoring for improved awareness in transmission systems [9]. Better monitoring means that faults are detected and corrected faster.

f) **Better Customer interaction:** Customer can participate more actively in the power grid and receive more feedback. There is clearer monitoring of power consumption, customers can sell generated power (e.g. from solar), and running devices can be managed better using Demand Response (DR) [13].

g) **Potential for new innovations:** Smart Grid's adoption will potentially lead to the innovation of many new applications that will improve the efficiency of the power grid as well as features that will help the customers to better manage their consumption. This include HEMSs, PHEV which would call for the efficiency of Smart Grids to handle the increase in energy demand [14].

Smart Grid is thus an enhancement to the traditional grid promising many benefits. Table 2.2, adapted from [1] compares the two in brief.

Table 2.2 A brief comparison of the conventional grid and the smart grid.

Feature	Conventional Grid	Smart Grid
Metering devices	Electromechanical	Digital
Communication	One-way	Two-way
Power Generation	Centralized	Distributed
Sensors	Few	Located throughout
Fault recovery	Manual restoration	Self-healing
Result of fault	Failures and blackouts	Adaptive and islanding
Control	Limited	Pervasive
Customer Choices	Few	Many

2.2.2 Smart Grid Communication Requirements

For Smart Grid to work, a number of factors need to be in place. Those related to communication include [15]:

i) **Acceptable network latency:** Latency is the maximum time within which a message should reach its destination through a communication channel. Different types of Smart Grid traffic have different amounts of acceptable latency.
ii) **Data delivery criticality:** Some applications, such as protection, are more critical than others and may need a protocol suite which meets their higher priority. Some data could require confirmation while in others, merely detecting data loss is sufficient.
iii) **Reliability:** The communication backbone should be reliable (i.e. have a low probability of error) for successful and timely message exchanges.
iv) **Time synchronization:** Some of the devices on power grid need to be synchronized in time.

v) **Multicast support:** This is helpful when a message is to be communicated to several peers at the same time. Instead of multiple individually addressed messages, a single multicast message is sent to a switch that forwards it to all outgoing ports. This reduces the network traffic.

vi) **Security:** Physical and cyber security from intruders is crucial. Smart Grid data traffic such as billing, signal prices and control messages is very important traffic that needs to be encrypted and prevented from malicious changes and unauthorized access.

2.2.3 Challenges and Barriers to Smart Grid

Each new technology brings with it a set of challenges, sometimes intertwined with the benefits it brings. An objective look at potential shortcomings is vital in overcoming and dealing with them. Many stakeholders are involved and an agreeable solution should be sought for. If customers are not satisfied with the new system, or they lose the trust they have for electric companies, then the full potential for Smart Grid will not be achieved. For the Smart Grid to work well, the following challenges need to be addressed:

a) **Cost-benefit tradeoffs:** Each metric can be improved, but almost always, only with a trade-off elsewhere. For example, installing smart meters has an upfront cost; however, there are future potential benefits from improved system reliability and efficiency [11].

b) **End-user acceptance:** Without acceptance, users may fail to subscribe to an optional program, or worse yet, those previously recruited may withdraw from a load control program, when their participation would be most needed [11]. Research and mass education is needed, so that expected benefits can better match reality. If users perceive that the new system will be a liability, say by exposing their privacy, they could protest its usage [16].

c) **Unanticipated user behaviour:** One might expect that users will change their power consumption patterns based on cheaper rates at less busy hours (i.e. ToU billing), but some could prefer convenience over savings.

d) **Conflicting stakeholder goals:** Consider a smart meter whose usage leads to lower costs as users become more conscious of their power consumption. This means less revenue to a utility company. Would the company hesitate to install the meter to reduce short term losses, or would they focus on the long term benefits the meter could bring?

e) **Challenges associated with communications:** These are problems inherent in communication standards and devices being used. They range from getting varying standards to work well together to privacy leakages due to increased data collection. This will be explored deeper in a later section.

2.3 Smart Grid Communication

Communication technology is a key component of the Smart Grid. Without it, many of the helpful features would be unattainable.

2.3.1 Communications Options for Smart Grid

There are a number of valid communication networks that can be used for Smart Grid. These vary based on the specific application. The options can be classified into three categories, namely, Power Line Communication (PLC), Wireline and Wireless networks [7][15].

a) Power Line Communications: It uses existing electrical wires to transport data. A modulated carrier signal is impressed on the wiring system. Typically data signals cannot propagate through transformers and hence the power line communication is limited within each line segment between transformers. PLC can be used in several

important applications such as utility metering and control, real-time pricing and distributed energy generation [17].

PLC offers extensive coverage, low installation cost, high speeds and is has less network congestion than wireless networks. It is however susceptible to noise, has limited bandwidth, offers low security and suffers from signal attenuation and distortion.

b) **Wireline Network:** Involves dedicated wireline cables, separate from the electrical power lines. They require extra investment on the cable deployment, but can offer higher communication capacity and shorter communication delay. It includes SONET/SDH, Ethernet, DSL, coaxial cable access network and Optical fibre. Optical fibre is one of the technically attractive communication infrastructures, providing extremely high data rates.

   Optical fibre has a high capacity, high noise immunity and suffers less attenuation thus is less dependent on repeaters. It does however involve a high cost of installation.

c) **Wireless Network:** Wireless networks can be used for either short range (e.g. ZigBee, Bluetooth and Wi-Fi) or long range communications (e.g. cellular network and WiMAX). Wireless networks offer convenience since extensive wires do not need to be laid, placement and installation of devices is more flexible and the cost can be relatively low. They do however have limited coverage, suffer attenuation and interference, have high power consumption and are exposed to more security vulnerabilities.

2.3.2 Smart Grid data Compared to Internet Traffic

Data in the Smart Grid has some significant differences with that of the internet. This is significant because solutions existing for solving problems in internet communication

need to be vetted and sometimes modified before adopting them to the Smart Grid. Without this, traffic requirements may not be met and the system will not be optimized. An important consideration in security solutions is that time delay is generally a more significant consideration in Smart Grid than in internet. Table 2.3 gives a summary of these differences [2].

Table 2.3 Summary of differences between smart grid and internet traffic

ASPECT	INTERNET	SMART GRID
1. Main Performance Metrics	*Throughput and fairness:* its basic goal is to provide users with data services (e.g., web surfing, music downloading)	*Latency (i.e. low Message delay):* it should ensure reliable, secure, and real-time message delivery and non-real time monitoring and management
2. Major traffic pattern	*Power-law:* many Internet traffic flows have the self-similarity property, e.g. WWW traffic. This is where few sites (e.g. google.com) have heavy traffic while many have little.	*Periodic:* most traffic is for consistent monitoring (e.g. raw data sampling in power substations) and periodic meter reading in home-area networks.
3. Timing requirement	Most IP traffic is best-effort traffic or has delay requirements of (100-150ms) for VoIP and media services.	Wide range of delay requirements from milliseconds to minutes, depending on application. (E.g. Trip protection messages have a 3 ms delay constraint.)
4. Communication model	*End-to-end:* It supports peer-to-peer communication between any pair of nodes in the world.	*Two-way:* top-down (centre to device) and bottom-up (device to centre). Peer-to-peer is restricted within the LAN.
5. Protocol stack	*IPv4/IPv6:* The internet is built upon the IP protocol	*Proprietary/heterogeneous:* The Smart Grid is expected to use IPv6, but can also have a mixture of protocols depending on application

2.3.3 Challenges in Smart Grid Communications

Several challenges need to be overcome in order to successfully deploy the smart grid.

i) **Interoperability:** Several communication options available for the transfer of the required information: fibre optics, copper-wire line, power line communications, and a variety of wireless technologies. There is need for communication standardization to ensure interoperability between different system components.

ii) **Scalable Internetworking Solutions:** Smart Grid implementation should be scalable, to allow for future network expansion (e.g. when new customers are introduced).

iii) **Cost:** Cost is an important factor especially due to the scope covered by smart grid.

iv) **Customer concerns:** Technological advances have made customers more willing to participate in decision making. Customers however do not want technology to interfere with their lifestyles, such as by demanding frequent settings modification. Interfaces need to be simple and accessible. They choose settings based on comfort, price and environment, with the rest being automated - an approach described as “set-it-and-forget-it” technology.

v) **Distinct characteristics of smart grid traffic:** Smart Grid traffic is different from most applications such as the internet. More on this later. Requirements for the Quality of service (QoS) will vary depending on the required service.

vi) **Privacy**: Large amounts of data from the user are required to optimize power distribution. The more information that is available, the smarter the decisions that can be made by a management system. On the other hand, more information could lead to privacy leaks [1]. Through analyzing how a user spends their power, their day to day activities can be known (e.g. when they have breakfast, when they watch TV or when they shower) [18]. It is also

possible to determine if there are people in a house and even achieve occupancy detection accuracies of more than 80% [4]. If such information would fall into the wrong hands (such as criminals), the effects would be devastating. Figure 2.3 shows one household's electricity profile recorded on a one-minute time base over a 24-hour period with many of the appliance events labelled [19]. The information on appliances and their usage can be inferred using a technique called Non-Intrusive Load Monitoring [20] which will be discussed later.

vii) **Cyber security:** The information should be transferred effectively without being accessed or manipulated by hackers. Unauthorized acquisition or alteration of such information can be potentially hazardous with great economic losses.

These last two problems – Cyber security and Privacy – are focus of this work. To address them, it is important to first distinguish the various communication applications in Smart Grid. These are referred to as use cases.

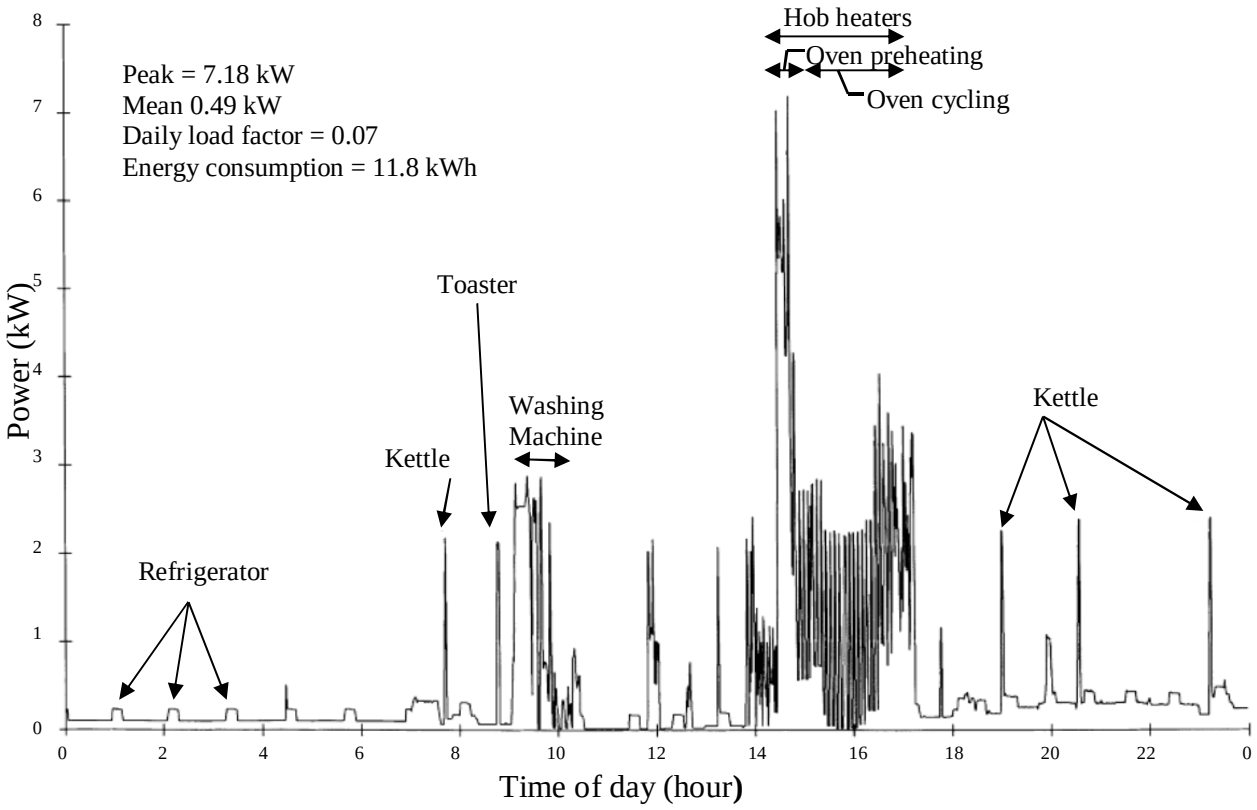


Figure 2.3 Household profile with devices identified.

2.3.4 Use Cases in Smart Grid Communication

Smart Grid is a complex network involving components with various functionality and requirements. It is an interconnection of many energy distribution networks, power generating sources, energy customers and all the devices in between [21]. Without distinguishing the components involved, one would inadvertently develop a system that works very well in one section, but is a serious liability in another.

Apart from Supervisory Control and Data Acquisition (SCADA) systems and other power substation systems, Smart Grid has other systems to be secured. NIST developed a report [22] which gives key use cases. They can be grouped into two: (i) Distribution and transmission operation as shown in Figure 2.4 (communication is time critical for

monitoring, control and protection), and (ii) Advanced Metering Infrastructure (AMI) and Home Area Networks as shown in Figure 2.5 (communication is mainly for interaction between customers and utility).

Five major use cases are discussed below [2]:

Case 1: Deals with ensuring reliable substation operations. It has tens of Intelligent Electronic Devices (IEDs) monitoring all feeder equipment and a substation computer serving as a gateway to outside networks. An attack on this would interfere with the protection system leading to damage of equipment or power loss.

Case 2: Works with a client-server communication model manage devices. It handles monitoring, control and protection beyond the local area systems. Device status and readings can be delivered to the SCADA centre. Authentication is a requirement, in order to avoid Man-in-the-middle attackers who would falsify data between two nodes.

Case 3: Provides a global snapshot of power signal quality by sending raw data samples of power signals to the SCADA centre. The samples need to be collected in a timely manner and need to be synchronized. The strict timing requirement makes it vulnerable to an attack which delays the arrival of the message (a weak Denial-of-Service attack).

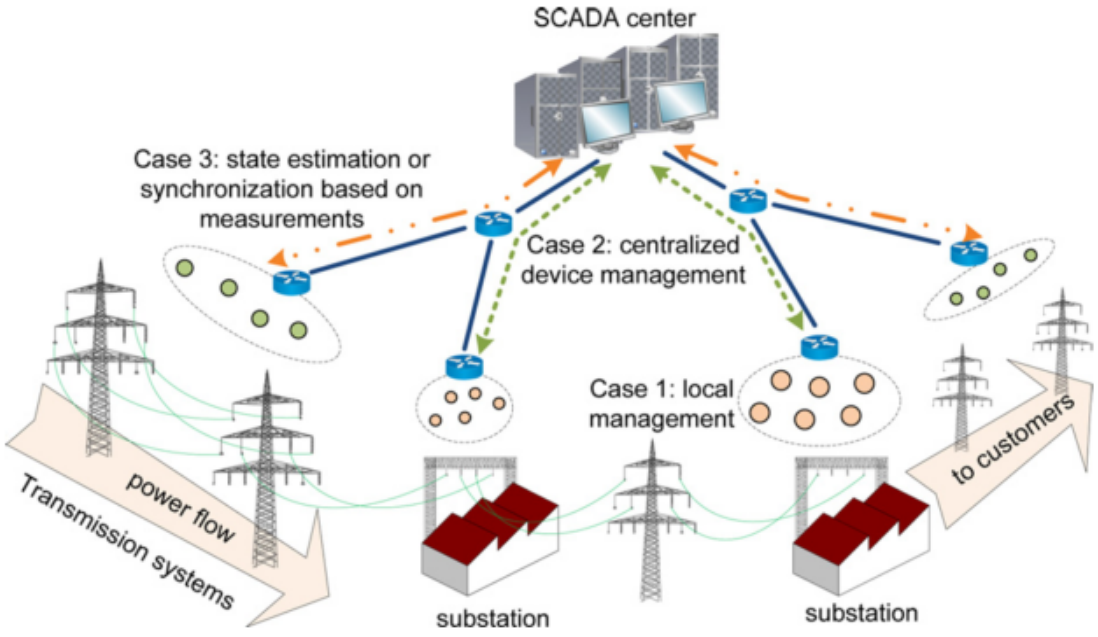


Figure 2.4 Use cases in Smart Grid distribution and transmission systems

The other two remaining cases focus on connecting utility companies to smart meters for scheduled energy management or demand/request response in customers' homes.

Case 4: Handles information exchange such as meter reading and maintenance between the utility centre and smart meters.

Case 5: is concerned with smart meter interactions to the electricity market such as real-time pricing and demand response.

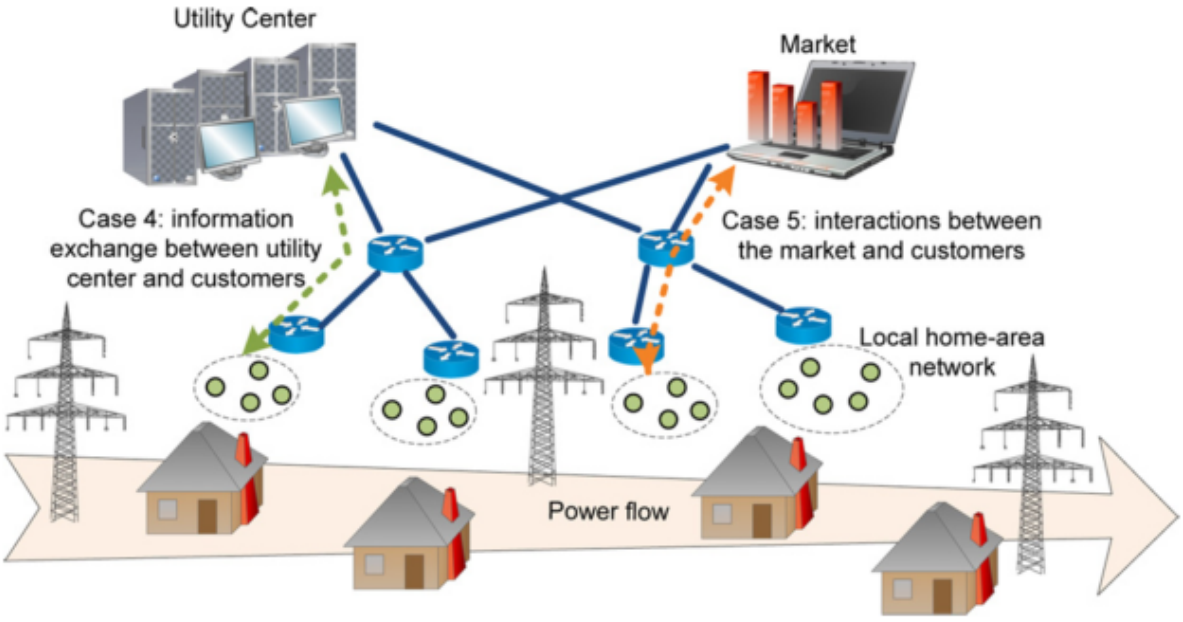


Figure 2.5 Use cases in the AMI and home-area networks

This work examines the security of the information exchange between the utility and customers which rely on smart meters.

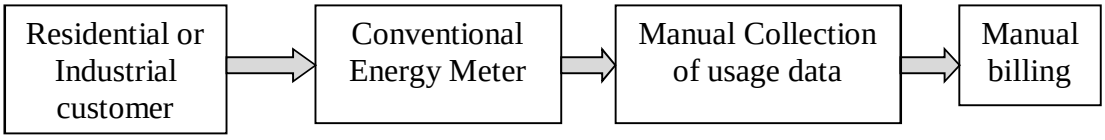
2.3.5 Smart Meters

A smart meter "is an advanced energy meter that measures the energy consumption of a consumer and provides added information to the utility company compared to a regular energy meter." [23] It is a crucial component of the Smart Grid, serving as the main communication between a utility company and its customers.

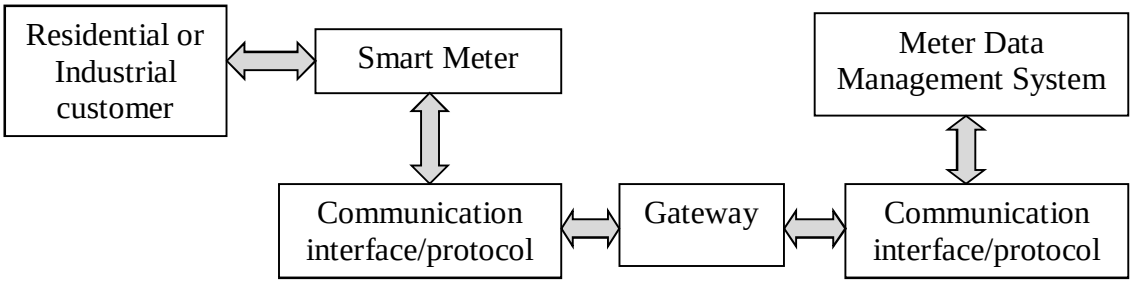
Smart meters offer utility companies several benefits. They reduce the manual labour needed to collect electrical consumption data while increasing the number of samples collected. They allow for remote disconnection and reconnection of electricity, based on whether a customer has paid their dues. They help utilities better plan on how to meet

demand, by providing more data. This reduces the losses incurred in meeting large amounts of unexpected demand.

Smart meters are contrasted from conventional meters in Figure 2.6, adapted from [23]. The Meter Data Management System (MDMS) works through storage servers and the collected data is used for three purposes: billing, operations and value-added services [24].



a) Conventional Energy Meter



b) Smart Meter

Figure 2.6 (a) Conventional energy meter and (b) smart meter architecture

Customers also benefit. Smart meters have the potential to lower the cost of electricity. A better awareness of how they consume power can make customers find ways of saving power. Smart meters can automatically modify operation schedules of certain home appliances moving them away from peak load and thus save costs. Users can also connect to other sources of energy (e.g. renewable sources) without being billed by the utility [23]. Due to the many benefits smart meters promise, the European Union (EU) aims to improve cost-effectiveness by replacing at least 80% of electricity meters with smart meters by 2020 [25].

As with any new technology, there are shortcomings that need to be addressed with the adoption of smart meters. Unlike the conventional meters, smart meters collect and transmit data at a high resolution. This data can be used to identify behavior patterns of the users [19] and thus lead to privacy leakage. With the usage of smart meters becoming mandatory in many countries across the globe [26], it is crucial to seek suitable solutions.

2.4 Cyber Security and Privacy

A successful attack on the Smart Grid can have severe consequences. Pricing and billing information could be modified, causing utility companies to incur major losses both in profit and customer confidence. Control signals can be manipulated, resulting in a cascade of failures, such as massive blackouts and destruction of infrastructure. Malware (malicious software) can be injected through communication channels to provide attackers access to control centres. Malicious commands can then be sent to trip critical lines leading to widespread blackouts as happened in Ukraine [27].

Privacy of customers can also be jeopardized. Information about what customers do with their electrical devices can expose their private lifestyle and habits (such as when they cook, watch TV, leave the house, etc.). Data mining on the overall power consumption can result in privacy leakage as shown in Figure 2.7. From the graph with the electrical

devices identified, it can be deduced that the home occupant(s) prepared their breakfast between 9 and 10 am, after which they left the house until around 5pm and prepared supper at 9pm. The electrical devices are also laid bare. Depending on who acquires this information, the risks on the electricity user can be far-reaching as expounded in section 2.4.3.

While cyber security and privacy problems have been addressed in various domains such as telecommunications and ICT, solutions from other fields cannot be borrowed directly to Smart Grid. This is because Smart Grid has some unique needs and a number of devices within it (e.g. smart meters) have limited computational power and storage capacity. Any existing solution thus needs to be vetted before applying it in Smart Grid.

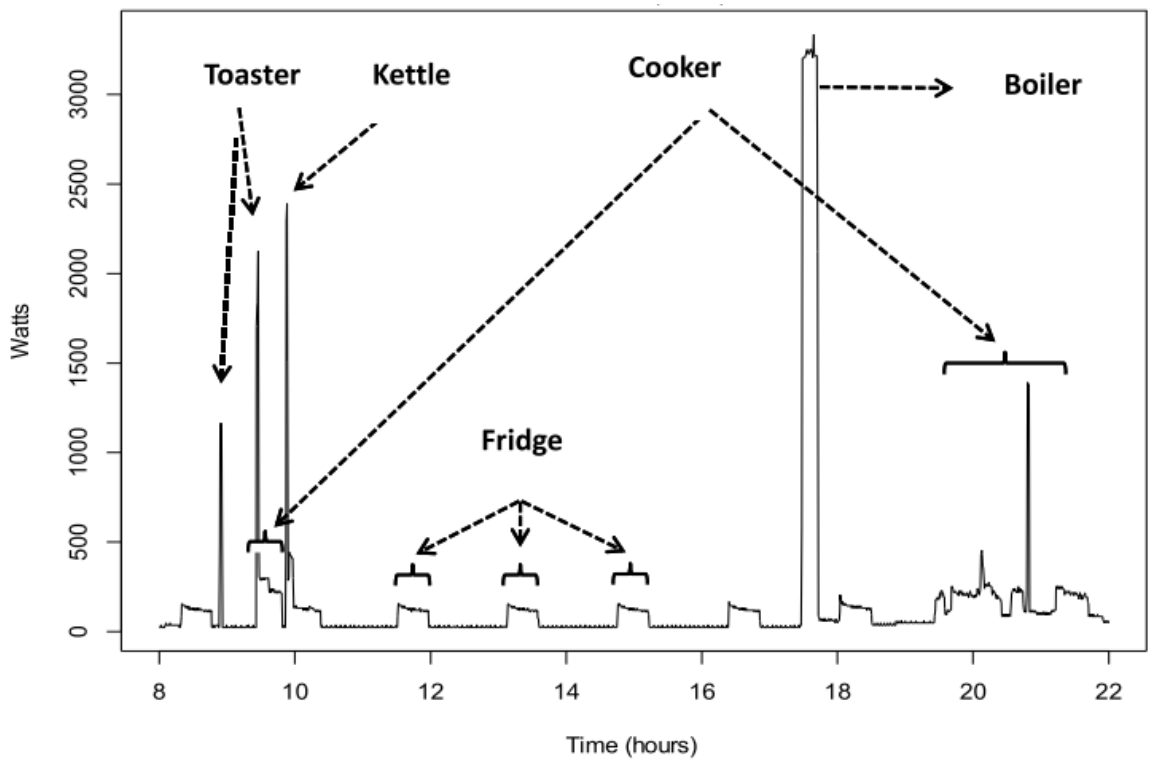


Figure 2.7 Household electricity demand profile from a one-bedroom apartment

2.4.1 Security objectives and requirements

Before attempting to protect a system, clear objectives are needed for establishing a basis on which to describe a system as secure. The cyber security working group in the NIST Smart Grid interoperability panel produced comprehensive guidelines for Smart Grid cyber security [22]. Three high level security objectives are given. These are:

a) **Availability:** Ensure timely and reliable information access.
b) **Integrity:** Protect against unauthorized modification or destruction of information.
c) **Confidentiality:** Guard against unauthorized information access, mainly to protect personal privacy and proprietary information.

The guidelines also give specific requirements for cyber security and physical security. Those relating to security of information are summarized as follows [2]:

a) *Attack detection and resilience operations:* with the large scope involved in Smart Grid, it is almost impossible to cover all possible attacks. The system should thus consistently check for any abnormal incidents due to attacks. It should further have self-healing ability to continue operations in the presence of attacks.
b) *Identification, authentication and access control:* ensures that resources are only accessed by the right users. It prevents unauthorized access to sensitive information or controlling critical infrastructure.
c) *Secure and efficient communication protocols:* Smart Grid needs to be both secure and time critical. These two goals tend to contradict because security is computationally intensive while time-criticality requires fast computation. Tradeoffs are required to balance communication efficiency and information security.

This work seeks to strike a balance between security and efficiency. This is important not just because many operations are time critical, but also because many Smart Grid devices tend to be embedded systems with limited computational abilities.

2.4.2 Types of security attacks

The dangers facing Smart Grid are many. While most cyber attacks have been aimed at financial institutions, other targets become more attractive as automation and interconnectivity increases. Smart Grid is prone to be attacked for two main reasons: we are heavily dependent on electricity for our day-to-day affairs and the electrical grid was designed without factoring some of the recent cyber threats (many of which did not exist before systems were interconnected).

Potential attackers can be classified into three categories:

a) **Selfish misbehaving users**: They aim to obtain more network resources by violating communication protocols [2]. Typically, they hoard bandwidth at the expense of other users to obtain higher download speeds. They are generally not a major concern for Smart Grid, where the focus is on monitoring and control.
b) **Malicious attackers**: These aim to illegally acquire, modify or disrupt information in the network [2]. They seek to harm the system without necessarily having any direct benefit themselves. This group includes disgruntled or former utility employees, young hackers trying to prove themselves to their community and ideologically motivated hackers. They may trigger catastrophic damage to power supplies and widespread power outage. These pose a serious threat to the Smart Grid.
c) **Attackers for monetary gain**: This group seek to get money through corrupting metering systems, falsifying billing information, acquiring services without paying

and even locking out users from a system until they receive a ransom payment [28]. Such attackers do not limit their scope to financial institutions but they keep expanding their scope of operations the more technology becomes part of our daily lives. Smart Grid thus provides new targets for attack.

There are many ways of attacking the grid. They generally involve violation of the major security objectives. Attacks can thus be classified based on which of the three security objectives is compromised:

- *Attacks targeting availability*: attempt to delay, block or corrupt communication in the grid. Also called denial-of-service (DoS) attacks. Delaying a time-critical message, such as a protection signal, can be as devastating as blocking it all together.
- *Attacks targeting integrity*: deliberately and illegally modify or disrupt data exchange.
- *Attacks targeting confidentiality*: acquire unauthorized information from network resources, including data that should be private.

On a practical level, these security attacks include:

i) A hacker can reduce his electrical bill or maliciously increase another's bill [29], leading to energy theft and a loss of trust by customers towards the utility.
ii) Some equipment have stringent latency requirements, (e.g. IEDs in substations should send their measurements to data aggregators within 4 ms [30]). A message delay on such time-sensitive data can lead to equipments' damage [31].
iii) Distribution Management Systems (DMS) and load forecasting will be affected if data falsification and injection target Advanced Metering Infrastructure (AMI) measurements [31].
iv) Denial-of-service attacks and the cyber vulnerabilities that are exploitable by malicious entities to disrupt smart grid operations on a large scale [32].

v) Eavesdropping on communication channels to obtain information, such as a customer's account number and electricity usage. This can be done by wiretapping [33] or traffic analysis [34].
vi) Privacy of the lives of customers can be heavily compromised. This is discussed in the following section.

2.4.3 Privacy Concerns in Smart Grid

The ability to tell when and how long a person uses various electrical devices can provide a surprisingly large amount of information about their lifestyle. With different devices serving varying roles in our personal lives, such information can be used to determine both your location and activities. It might be thought that getting detailed information about device usage might require installation of invasive monitoring hardware at all power outlets, but that is not the case. A technique was developed in the early 1990s that made it possible to extract information on appliance usage by reconstructing it from the overall load data [20]. This technique is known as Nonintrusive Appliance Load Monitoring (NALM or NIALM) and as the name suggests, it can be used to monitor loads without any intrusion on a household.

NALM works because different electrical loads have different patterns of how they consume power. For example, a refrigerator goes on and off at certain periods and draws a certain amount of power. This gives it a distinct pattern of usage from other loads, which is known its *load signature*. This can be used to distinguish loads based on their unique behaviour as shown in Figure 2.8 [35].

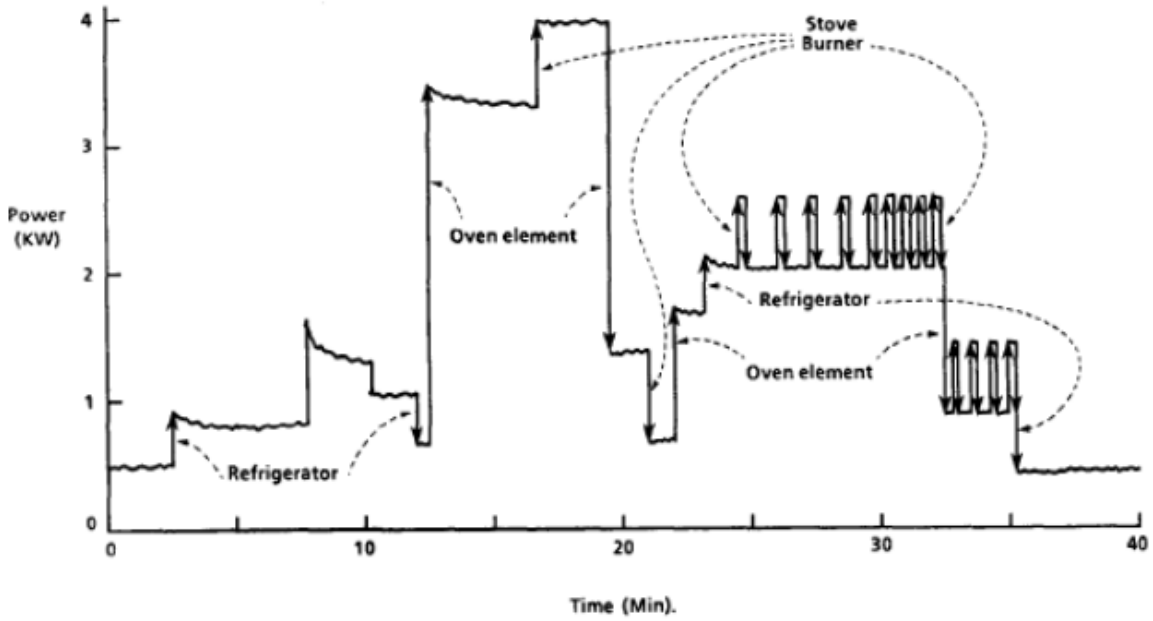


Figure 2.8 Power consumption over time showing different appliance events

NALM has undergone improvements over the years which include training the system [36], thus making it even more powerful. Positively, NALM can be used to provide customers with personalised energy saving feedback by showing them where losses are incurred. Negatively, NALM makes it possible to have serious privacy leakage that is comparable to a person peeping through your home's curtains. With smart meters collecting high-resolution data, privacy exposure becomes a practical threat [37]. The risk of privacy exposure comes in many ways, some of which are rather unexpected:

i) Burglars could monitor the power consumption of several households to identify temporarily vacant homes and time their break-ins [16]. They will also know which devices they expect to steal and by checking the load signatures, they can tell which houses have a working electrical alarm.

ii) A landlord or a stalker could estimate the number of residents in a household based on the frequency of power switches turned and the number of appliances simultaneously in use [16].

iii) A flood of targeted advertisement (e.g., informing customers about inefficient appliances) could be sent [26].
iv) Debt collectors know exactly when they should collect their dues.
v) User profiling and household classification is possible [26]. Someone analysing the data would be able to estimate the income level of home owners based on their devices and activities.
vi) An employer could discover the usage of an employee issued device (e.g. Company laptop) by someone a housemate of the employee.
vii) One could monitor the location of a resident inside the home based on the type of appliances being used [38].
viii) Health insurers could track eating, sleeping, and to some extent exercise habits by monitoring household appliance usage [18].
ix) Under the right conditions, it is even possible to identify the TV channel or movies being watched since television power consumption changes with the image being displayed [39]. In addition to snooping on people's behaviour, this also causes copyright infringement since it would bypass security measures to hide the media content through Digital Rights Management (DRM).
x) A spy could unveil hidden manufacturing processes by analyzing a company's electrical consumption. Such information might reduce a manufacturer's competitive edge. It could also expose proprietary processes.
xi) Unexpected combinations of household information with other data sources could result in even more privacy leakages. This could involve superimposing Google maps with household data and selling the data to interested parties [40].

It may be argued that some of this information could help curb crimes and can be put to good use (e.g. by police), but even if some benefit could be derived from it, how is to be keep from malicious use? Criminals, tyrannical regimes, terrorists, identity thieves, hackers, spies and overzealous marketers could all wish to exploit such data for their own

interests. This is especially problematic because the exposure is happening right within the context of our own homes, which we expect to be a shelter of privacy.

The solution to these privacy problems calls for both technological solutions (involving engineering techniques) as well as policy making and legal protection (involving "regulations setting consent requirements for the disclosure of smart meter customer information to third parties" [19]). This thesis focuses on the technological solutions.

2.4.4 Solutions to Privacy Leakage

Given the far reaching effects of privacy violation, it is little wonder that this has been an interest to many researchers. The problem of privacy has been looked into from various angles. Kalogridis et. al. [41] propose an approach known as Battery-based Load Hiding (BLH) (see Figure 2.9) which was later improved upon by Yang, Li and Qi [42]. BLH works by using an algorithm that seeks to flatten the load profile by switching between the battery and the mains in order to hide device usage information. It thus blocks NALM from working. Two shortcomings of using BLH are that power is lost since batteries are never is 100% efficient, and the high cost of batteries might keep users from applying it to their privacy needs.

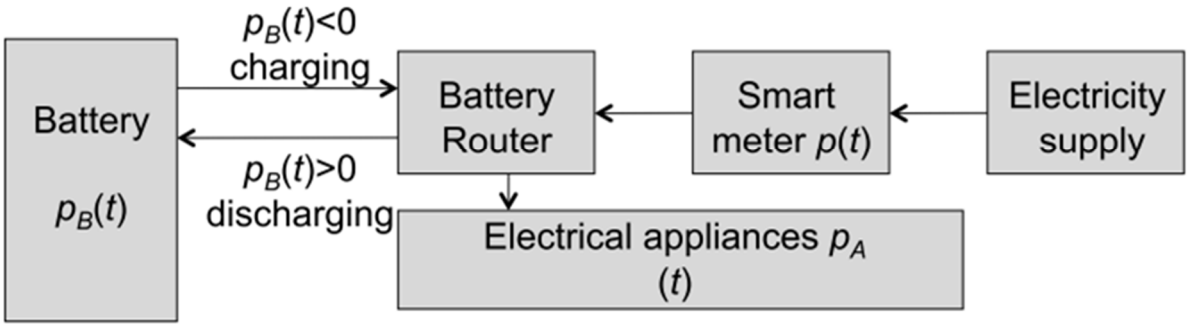


Figure 2.9 Battery-based Load Hiding Algorithm

Rial and Danezis [43] propose a scheme where a user combines meter readings with a certified tariff policy to produce a final bill. The bill is then transmitted to the provider alongside a zero-knowledge proof that shows the calculation is correct, without leaking any additional information.

Efthymiou and Kalogridis [44] propose the use of *anonymization* of smart metering data. They use a trusted third party to ensure that the frequent electrical metering data sent by smart meters is anonymous. The utility is thus able to get the information it needs for its operations, but the high frequency data is not does not need to be attributed to a particular meter. In their conclusion, they admit that the method "may not offer sufficient smart metering privacy protection [but] contributes an additional layer of security towards that direction." [44]

The use of a photovoltaic converter has also been presented by Reinhardt et. al. [26]. This modifies maximum power point tracking (MPPT) in order to generate fake load signatures. Though it does not introduce power loss as in the case of using batteries, it requires users to have solar and at the same time, it reduces its efficiency.

Homomorphic encryption is a fairly recent development in cryptography [39] that can offer privacy protection. It allows one to manipulate encrypted data without decrypting it. Both eavesdroppers and the utility company cannot access all the customer data. Only aggregate usage data from various customers is available [10]. Homomorphic encryption comes with significant computational cost. Smart meters are generally light-weight devices and some may not handle Homomorphic encryption. The smart meters also need to have a trusted component and enjoy a certain level of autonomy [40].

End-to-end encryption involves encrypting the data between a customer and a utility. A shared key between the two enables them to encrypt and decrypt the data. A strong and well implemented encryption algorithm thwarts an eavesdropper's efforts. In the context

of smart meters, computational complexity and security, need to be balanced to avoid either having a system that is very fast but insecure or one that is extremely slow but secure.

Table 2.4 below compares the above methods on a number of factors: do they introduce power loss? Is there a significant increase the computational or communication overhead? Do you need to have a trusted third party for them to work? Can the utility company access the customer's data? This comparison is helpful in choosing a method suitable to a particular setting with its own unique threats. It could further help in combining options and having multiple layers of security to curb a range of threats.

Table 2.4 Comparison of various privacy enhancing techniques

Technique	Power loss introduced	Computational/ Communication overhead introduced	Trusted third party needed?	Can utility company access data?
Battery Load Hiding (BLH)	Yes	No	No	No
Solar PV convertor	Yes	No	No	No
Anonymization	No	Yes	Yes	No
Homomorphic encryption	No	Yes	Yes	No
End-to-End encryption	No	Yes	No	Yes

We now move to a discussion on encryption, which is the method chosen in this work to aid in cyber security and privacy.

2.5 Encryption

Encryption is a cryptographic technique used to provide secure communication and information protection. It involves making data illegible to third parties, typically hackers,

but giving the intended recipient the ability to retrieve the message. A key is used to convert a message, known as the plaintext, into ciphertext (encrypted information). The recipient then uses a key to convert the ciphertext back to plaintext. The harder it is for another party to obtain the plaintext from the ciphertext, the better the encryption technique. Figure 2.10 demonstrates how encryption works [45].

Two communicating parties, usually named Alice and Bob, intend to communicate through an insecure channel. Alice could be a customer's smart meter while Bob is the utility company. An attacker (Oscar) seeks to eavesdrop on their communication or even manipulate the data. By first encrypting the data using a key only known to the two of them, Alice and Bob can prevent Oscar from obtaining the original message.

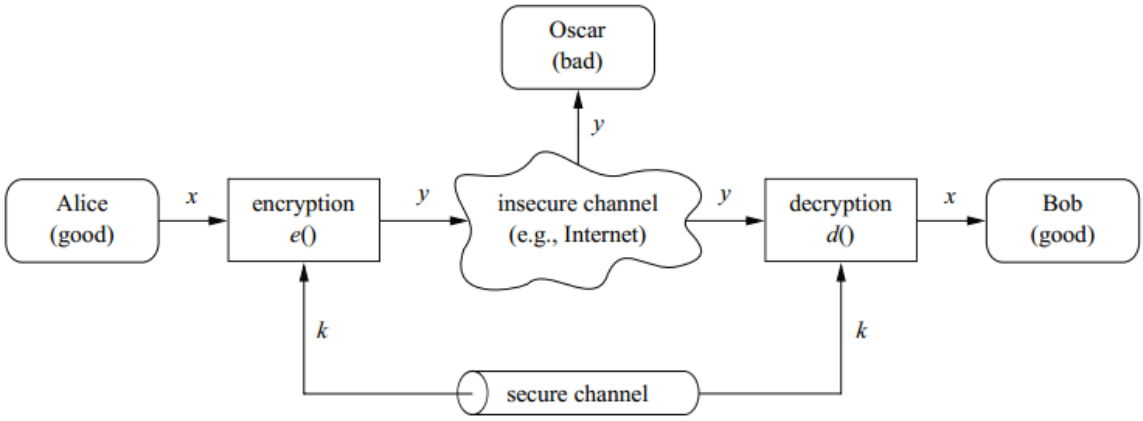


Figure 2.10 Block diagram of Symmetric Encryption

Encryption techniques can be divided into two major categories (symmetric and asymmetric encryption), depending on how the key is applied. In *symmetric key* encryption the same key is used (as seen in Figure 2.10), while *asymmetric key* encryption uses two different, but related keys (a public and a private key). One key is used for encryption while the other is used for decryption.

The length of a key is an important parameter in determining the security of an encryption algorithm. The longer the key size, typically measured in bits, the larger the key space (i.e. number of possible keys). A large key space makes it impractical to guess the key using an exhaustive trial and error search (i.e. using *brute-force*). An encryption algorithm is said to be computationally secure against a brute-force attack if it would take too long (or cost too much) to test all keys on modern computers. Table 2.5 gives estimates of how long it would take to successfully attack various key lengths [45]. As technology improves, the time needed decreases.

A large key space is necessary for security but it is not sufficient because there are other ways to break encryption other than using brute-force. These other methods are called analytical attacks and involve mathematical analysis, observing patterns in the ciphertext, exploiting weaknesses in the encryption method and other creative schemes for recovering the data. Care and keeping abreast with breakthroughs in cryptography is needed to ensure systems are still secure. Since analytical attacks cannot be measured, how is one to determine if an encryption method is secure? Principles of best practice are to be followed which include using tried and tested methods and applying suitable analysis to the context being considered.

Table 2.5 Estimated time for successful brute-force attacks on different key lengths

Key Length	Security Estimation
56-64 bits	short term: a few hours or days
112-128 bits	long term: several decades in the absence of quantum computers
256 bits	days long term: several decades, even with quantum computers that run the currently known quantum computing algorithms

An important concept in selecting an encryption technique is the *Kerckhoffs'* Principle postulated in 1883 [40][45]. The principle teaches that the security of an algorithm should only depend on the key being secret and not on the details of the algorithm being hidden. It is unrealistic to expect a widely used algorithm to stay secret for long. Thus it is better to work with algorithms that have been subjected to both professional analysis and amateur experimentation without serious deficiencies being discovered. Thus, one should not put confidence in an encryption method whose operation details have to be hidden for it to be secure. Such "security by obscurity" is to be shunned, since it will almost inevitably contain hidden weaknesses that hackers will uncover sooner or later. A tested and tried algorithm is the safer option and new algorithms should be subjected to wide scrutiny by making them public.

A desirable property of any encryption algorithm is the *Avalanche Effect* [46]. This is where a small change in either the plaintext or the key should produce a significant change in the ciphertext. In particular, a change in one bit of the key (or plaintext) should produce a change in many bits of the ciphertext. If the change were small, this fact might reveal a weakness in the cipher that would make a hacker's task easier. An encryption algorithm that does not exhibit the Avalanche Effect is generally easier to break than one which does.

2.5.1 Symmetric versus Asymmetric Ciphers

Which option is better for encryption of smart meters – symmetric or asymmetric? Asymmetric ciphers tend to need much longer keys to achieve the same level of security as symmetric ciphers. To obtain equivalent security, symmetric keys have bit lengths considerably smaller than that of private keys in public-key systems, (e.g., by a factor of 10 or more) [47]. For example, the popular asymmetric algorithm RSA requires 3072 bits to provide the same strength as 128 bit symmetric key system [45].

Asymmetric algorithms also take much longer to execute. Their complexity grows roughly with the cube of the key's bit length [45]. As an example, increasing the bit length by a factor of 3 (i.e. from 1024 to 3076) in a given RSA signature generation software results in an execution that is $3^3 = 27$ times slower. Symmetric key cryptography on the other hand requires approximately constant computational resources regardless of the key size [46]. Symmetric ciphers however require secure exchange and update of secret keys among network nodes, thereby complicating the process of key management. These differences are summarized in Table 2.6.

Table 2.6 Main differences between symmetric and asymmetric ciphers

	Symmetric ciphers	**Asymmetric ciphers**
1	Need relatively shorter keys	Need relatively longer keys
2	Complexity does not vary much based on the key size	Complexity grows roughly with the cube key size
3	Key management is complex	Key management is more flexible

Thus, symmetric encryption is a better option for smart meters, which consist of light weight embedded systems with limited storage.

2.5.2 Block versus Stream Ciphers

Symmetric encryption can further be divided into two groups: block and stream cipher. A block cipher is one in which a block of plaintext is treated as a whole and used to produce a ciphertext block of equal length. A stream cipher encrypts plaintext one bit or byte at a time.

Stream ciphers are typically faster and use far less code than block ciphers [46]. This gives it an advantageous application in smart metering. A summary of encryption classification is given in Figure 2.11.

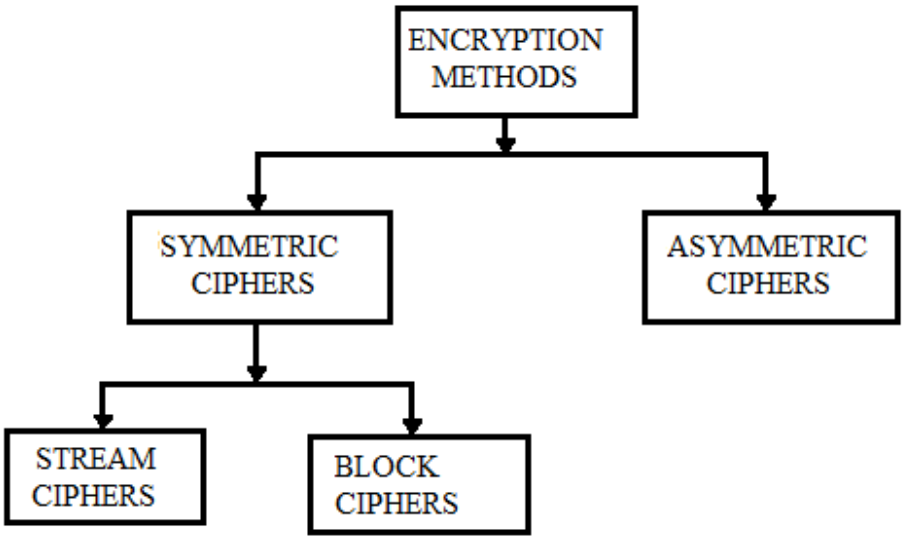


Figure 2.11 A summary of classification of Encryption Techniques

In stream ciphers, plaintext is typically encrypted by taking its bitwise exclusive-OR (XOR, symbolized as ⊕) with a stream of bytes. If the plaintext stream is 11001100 and the key is 01101100, then the ciphertext is 10100000 as shown below:

	11001100	plaintext
⊕	01101100	key stream
	10100000	ciphertext

Decryption uses the same operation:

	10100000	ciphertext
⊕	01101100	key stream
	11001100	plaintext

Figure 2.12 shows how stream ciphers operate in general [46]. The key difference comes in the type of pseudorandom byte generator.

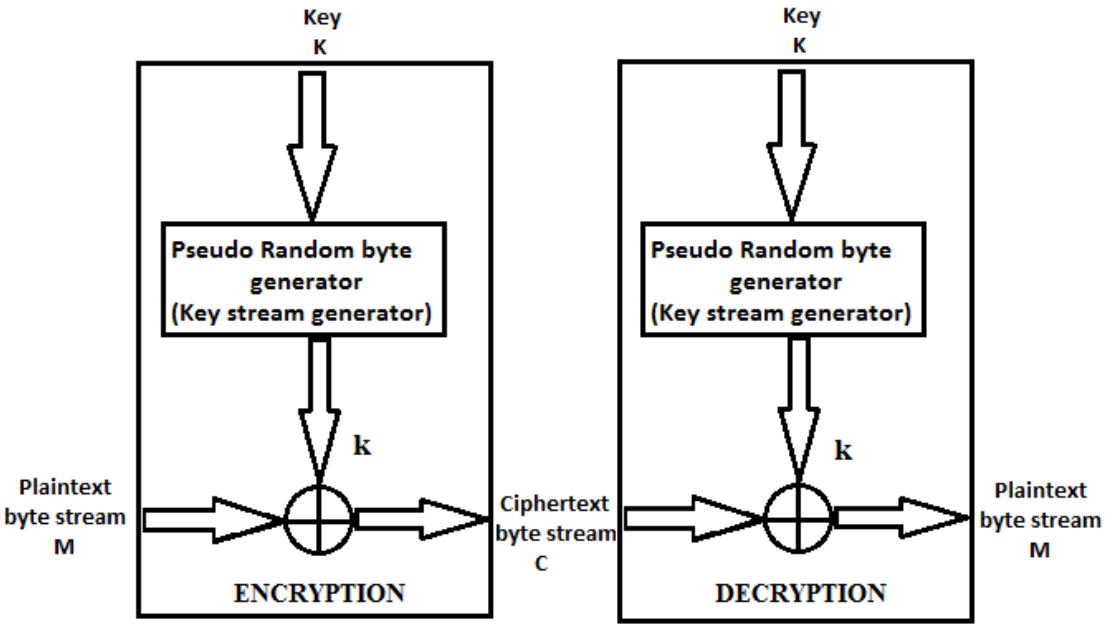


Figure 2.12 Block Diagram showing the operation of a stream cipher

2.5.3 Key Selection

With symmetric ciphers, selection of a key is a crucial task. As mentioned earlier, because of *Kerckhoffs'* Principle, one should assume that the attacker knows (or will eventually learn) the details of the encryption method used. History has shown that if the details of the encryption method are kept secret, it is just a matter of time before they are discovered and the method becomes insecure [40]. The secrecy of a system must not be based on hiding the details of the method, but on the secrecy of the key being used.

The size of a key is a crucial aspect in implementing a cryptographic solution. This is because the most straight forward way to break code is to try every possible key, an approach known as brute force. Technology keeps getting faster and cheaper each year so it is prudent to choose a key that is long enough to resist attacks, even those which do not currently seem practical yet. As it stands, a key size of 128 bits is considered secure. With such a key, the total number of possible keys is:

$$No\ of\ Keys = 2^{128} \cong 3.403 \times 10^{38}$$

Even if it were possible to test 1 billion keys every second, it would take 10^{22} years to exhaustively search through all keys and thus break the cipher [48].

Generation of keys is an important facet of securing a system. Done poorly, it would make the strongest of encryption algorithms very weak. A common way of doing this is by using a password. If this task is to be left to the customers, it introduces two major problems, especially in the application of smart meters. First, there is a high tendency for them to use words which are easy to remember and in doing so allow attackers to use *dictionary* attacks. This is where common words, usernames, special numbers, etc, are used to try guessing the key. Second, an appropriate and secure method of communicating this key to the utility would also be needed.

Leaving each user the task of coming up with their own key is undesirable. Given the number of customers a country's utility typically has (usually in the millions), a scalable solution is required in which the generation of many keys would be automated. The keys can then be stored in an encrypted form within the smart meters while the utility securely keeps its copies. A pseudorandom generator can help generate keys in bulk. This work proposes a method that uses *hash functions*, which are good at producing pseudorandom numbers.

Hash functions are algorithms which take an input of variable size and produce an output with a fixed number of bits. This output is a condensed representation of the message and is called the *message digest* (e.g. an input of a 2 MB file can produce a specific output of a 512 bit digest). Figure 2.13 from [46] explains the general concept of what a hash function does. Among the existing hash functions available, the most reliable and widely used belong to a family of algorithms developed by the National Security Agency (NSA)

known as Secure Hash Algorithm (SHA) [49]. SHA hash functions vary in terms of the size of the output produced, the largest being 512 bits and is known as SHA-512.

Hash functions operate by combining various bitwise operations such as exclusive OR, addition modulo 2^n, left/right shifts among others. These are repeated over several rounds (e.g. 80). The result is an output that is has a very high probability of changing whenever the input is altered. This is a desirable property when considering the generation of millions of unique keys for use in encryption [46].

A helpful way of examining if the keys generated are secure, keys produced from adjacent meter numbers can be compared. If a vast majority of the bits are similar, this could be an indication that the key generation method in question is insecure. A hacker would look for a way of guessing other keys if one of them was compromised.

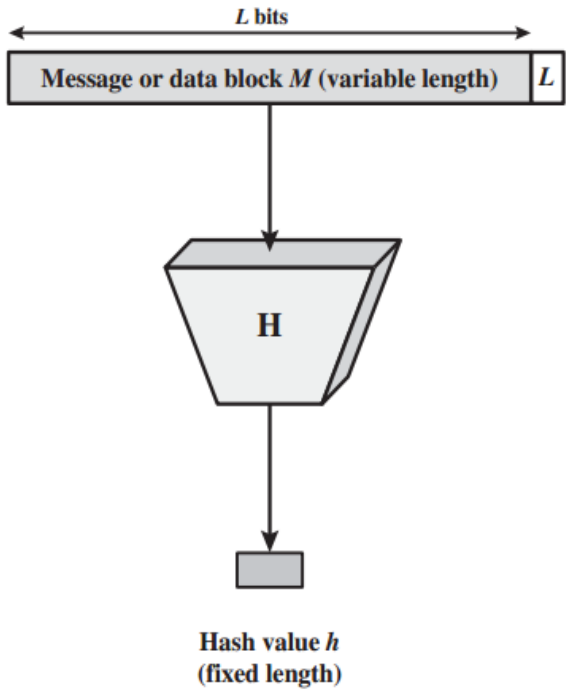


Figure 2.13 Block diagram describing a general hash function, H

The *hamming distance* is a useful quantity in comparing the keys produced. The hamming distance between two binary strings of equal length is the number of positions at which the corresponding bits are different [50]. It is denoted by d($\mathbf{v}_1$,$\mathbf{v}_2$). As an example, if $\mathbf{v}_1$ = 11011 $\mathbf{v}_2$ = 10110, then the hamming distance is d($\mathbf{v}_1$,$\mathbf{v}_2$) = 3, since there are 3 positions where the two are different (i.e. at the 2nd, 3rd and 5th bits). If the hamming distance between keys generated by adjacent meter numbers is approximately half their length, then the generation technique can be considered secure. If all the bits were different, this would merely imply a bitwise negation of the key (whereby all 0s are interchanged with 1s and vice verse). This could also provide an attacker with information from which they might identify a pattern.

2.5.4 RC4 Encryption

RC4 cipher is a very popular stream cipher. It has been used in Wired Equivalent Privacy (WEP), Wi-Fi Protected Access (WPA), Transport Layer Security (TLS) and many other applications. It is a relatively simple algorithm, able to work quickly in software. Being a stream cipher, it works by generating a pseudorandom stream of bytes used in the encryption. This stream is at the heart of its security, and has been show to have a very large period ($> 10^{100}$) before it repeats its values [46]. The details for the RC4 algorithm are given in Appendix 1.

The core of RC4 cipher is a state vector S, which contains a permutation of 256 bytes, i.e. [0, 1, 2, … , 254, 255]. It also has 2 registers, each of 1 byte in length. The number of possible states that RC4 has is at most

$$\#(N) = N^2 N! \tag{2.1}$$

Substituting N = 256, this computes to $\approx 5.62 \, x \, 10^{511}$ states. In terms of information, this can be expressed $\log_2 N^2 N!$ which computes to ≈ 1,700 bits. Attackers can attempt a *state*

recovery attack in which an output sequence is used to determine the internal state. The best state recovery attack has a complexity of 2^{241} steps [51].

RC4 thus has suitable properties for smart meters and has in fact been implemented in the Open Smart Grid Protocol (OSGP) [52]. However, it has been shown to have weaknesses in the way that the implementation is done [5][6]. These weaknesses are comparable to the same ones which lead to the breaking of Wired Equivalent Protocol (WEP) – a protocol for encrypting wirelessly transmitted packets on IEEE 802.11 networks [53][54]. An efficient encryption system is thus needed that would overcome the challenges of RC4 while maintaining its strengths.

2.5.5 Spritz Encryption

The developers of RC4 developed a "drop-in replacement" for RC4 known as Spritz [3]. It was designed to have improved security by revisiting particular design decisions and improving on them, in light of known attacks. Using Spritz is not violation of *Kerckhoffs'* principle because its details have been publicly published and its developers factored well known attacks in its development.

Like RC4, Spritz is a stream cipher that typically works byte by byte. Its operation has what is called a sponge-like construction. Roughly speaking, sponge-functions work by "absorbing" numbers and "squeezing out" results. Spritz has an element array, S, containing N values (typically 256) whose values are altered during the running of the algorithm. The bytes used in encryption and decryption are produced in a manner analogous to a squirt of liquid, hence the name Spritz. Its algorithm is given in Appendix 2. Spritz also provides additional cryptographic capabilities, (e.g. hashing, MAC (Message Authentication Code), Authenticated Encryption).

Spritz consists of 6 other registers in addition to a state vector S. The number of states Spritz has is at most

$$\#(N) = N^6 N! \tag{2.2}$$

This computes to $\approx 3.684\, x\, 10^{516}$states (or 1,716 bits). Attackers can attempt a *state recovery* attack in which an output sequence is used to determine the internal state. The best attack published for recovering the state of Spritz as of this writing requires 2^{1247} steps [55] ≈ 2.423x 10^{375}steps. Though this is an improvement from brute-force, it is still well outside the reasonable range of our best supercomputers.

2.5.6 Qualitative improvements of Spritz over RC4

Spritz cipher was developed to overcome weaknesses in RC4 and offers several benefits [3]:

1. It provides additional cryptographic features that can be useful in a wide range of applications. As an example, it provides a hash function, which can be used to update the key if necessary.
2. It has a larger state, making it theoretically harder to break than RC4.
3. It was designed to curb a number of weaknesses identified in RC4 including:
    a. Existence of weak keys in RC4 [56].
    b. Biases in the output stream that is generated [57], [58].
    c. A lower state recovery attack (complexity of 2^{241}steps compared to 2^{1247} steps in Spritz [51], [55]).
    d. RC4 has key collisions [59]. These occur when two different keys produce the same state. This results in colliding keys producing the same output stream.

2.5.7 Statistical tests for Cryptography

The task of providing a quantitative test for examining the objective strength of cryptography can be an elusive one. This is because an attacker can come up with an ingenious technique to bypass a well developed system by finding an unexpected loophole. However, the Federal Information Processing Standards (FIPS) has developed "security requirements for cryptographic modules" [60]. They give four statistical tests that can be used to examine the randomness of cryptographic output. These tests are the *monobit* test, *poker* test, *runs* test, and *long runs* test. All the tests start with a 20,000 bits sequence, which is then analyzed as follows:

a) **The monobit test:** The number of ones in the 20,000 bits is counted and the value denoted as *X*. The monobit test is passed if

$$9{,}725 < X < 10{,}275$$

b) **Poker test:** The 20,000 is arranged in 5,000 consecutive segments of 4 bit each. Each of these segments is a value ranging from 0 to 15 (i.e. 0000b to 1111b). The number of occurrences for each value is counted and denoted as *f(i)* where $0 \leq i \leq 15$. From there *X* is computed by the expression below:

$$X = {}^{16}/_{5000} * \sum_{i=0}^{15} [f(i)]^2 - 5000 \qquad (2.3)$$

With this quantity *X* computed, the test is passed if

$$2.16 < X < 46.17$$

c) **Runs test:** The occurrence of consecutive ones or zeros is called a *run.* The number of repeated ones or zeros is called the length of the run. Consider the binary string 001110. It starts with a run of zeros of length 2, followed by a run of ones of length 3 and ends with a run of zero of length 1. In a random string, such as one produced by cryptography, it is expected that the most

common run would have a length of 1 with longer runs having decreasing probabilities of occurrence. When all the runs are computed, the test is passed if both ones and zeros have their runs falling within the following ranges (the last category is for 6 or more).

Table 2.7 Range of intervals for passing the runs test

Length of run	Required interval
1	2,315 – 2,685
2	1,114 – 1,386
3	527 – 723
4	240 – 384
5	103 – 209
6+	103 - 209

d) **Long run test:** A long run occurs when there are 26 or more consecutive ones or zeros. The test passes if there are no long runs.

Passing all this tests does not guarantee that a cryptographic method is secure from every conceivable attack. Weaknesses could still be present, though fairly hidden. However, a cryptographic method that fails these tests is susceptible to attackers who would be even more motivated to uncover the weaknesses.

2.6 Summary of Literature survey

From what has been examined above, we can summarize that:

i) The Smart Grid is a very promising system, though it needs to have the communication challenges – especially cyber security and privacy – addressed for it to work effectively.

ii) Security solutions from other communication scenarios may not work in smart meters due to their low computational abilities.
iii) The light weight encryption schemes that are most plausible for smart meters could be stream ciphers.
iv) Generating keys is important for the proper working of symmetric ciphers.
v) Though the popular RC4 has been implemented in smart meters, it has been found to have significant weaknesses.
vi) Spritz is a promising candidate for encryption in smart meters, being an improvement on RC4, being relatively light weight and having good security. It is however more computationally involving and a comparison is needed to see how it fares in particular applications, compared to RC4.

CHAPTER THREE

METHODOLOGY

3.1 Problem formulation

This work aims to demonstrate the working of an encryption scheme suitable for smart meters. It thus addresses the problem of privacy and Cyber Security. The scheme should provide reasonable levels of security in spite of a meter's limited computational resources, and should do so without unduly affecting its operation. Encryption of customer usage data before transmission to the utility keeps hackers from accessing it and spying on customers.

Data from a typical household was used to demonstrate the impact of encryption and decryption. The data was obtained from a graph in literature [35] that had been collected over a 24 hour period using a one-minute time interval. Image processing was used to extract data points from the image. The technique used is described below.

Two encryption algorithms were used, namely, RC4 and Spritz. RC4 has been deployed in actual smart meters on customer premises and is thus able to run without diminishing the performance of the meters. However, the usage of RC4 was found to be insecure. It was used in this work to provide a benchmark to compare with Spritz in terms of performance. If Spritz can perform comparably well to RC4 in an environment with limited resources, it can prove to be a suitable candidate for smart meter encryption, but Spritz must also prove to be more secure than RC4. The main goal of this chapter is to test if Spritz meets the two requirements of being both relatively fast and secure.

Encryption schemes need keys that will be shared between the transmitter and receiver. In this work, keys were generated with the aid of hash functions of the meter numbers. To examine if adjacent keys created using this method are related, the hamming distance was

computed for the neighbouring keys. An observable relationship between neighbouring keys would be highly undesirable because it might provide hackers information they can use to decrypt data from neighbouring meters.

After the encryption is performed, tests were done to examine what would happen if a hacker tried to decrypt with wrong keys. Three keys were used: one which was identical to the actual one apart from one bit, another which was a bitwise negation of the actual key and a final one generated at random. If any of these chosen keys produced data that was even close to the original, it would be a sign of weakness in the cipher. It is likely that the cipher provides a hacker with extra information that they can use to tell if their guesses are approaching the correct key. This would greatly simplify the task of breaking the cipher.

The speeds for RC4 and Spritz were compared to see how their performances relate. Spritz is a more complex algorithm than RC4, and was thus expected to take longer. However, given that RC4 has been used in smart meters (under the OSGP standard), what is needed is to see how much slower Spritz would be. If the time taken is of the same order of magnitude, then it would be a suitable candidate for smart meter encryption.

Comparing the strength of RC4 and Spritz was attempted using two approaches. The first was applying statistical tests for randomness on the output produced by both. The second approach was using brute force to get the key for both ciphers. Short keys were used to make brute force attack practical, since this entails searching through all possible keys. The time taken for each was computed and a plot made to compare which of the two is harder to break.

Matlab and Octave (two related and strongly compatible programming software programs) were used in processing data, performing encryption/decryption, timing,

plotting and analysis of the results. The experiments were done on an Intel(R) Core(TM) i5-2540M CPU @ 2.60GHz. The details of the above methodology are given below.

3.2 Extraction of data for Encryption using Image Processing

The typical household data shown in Figure 3.1 was used as the source of data for testing encryption. The data was extracted using image processing with a technique described below.

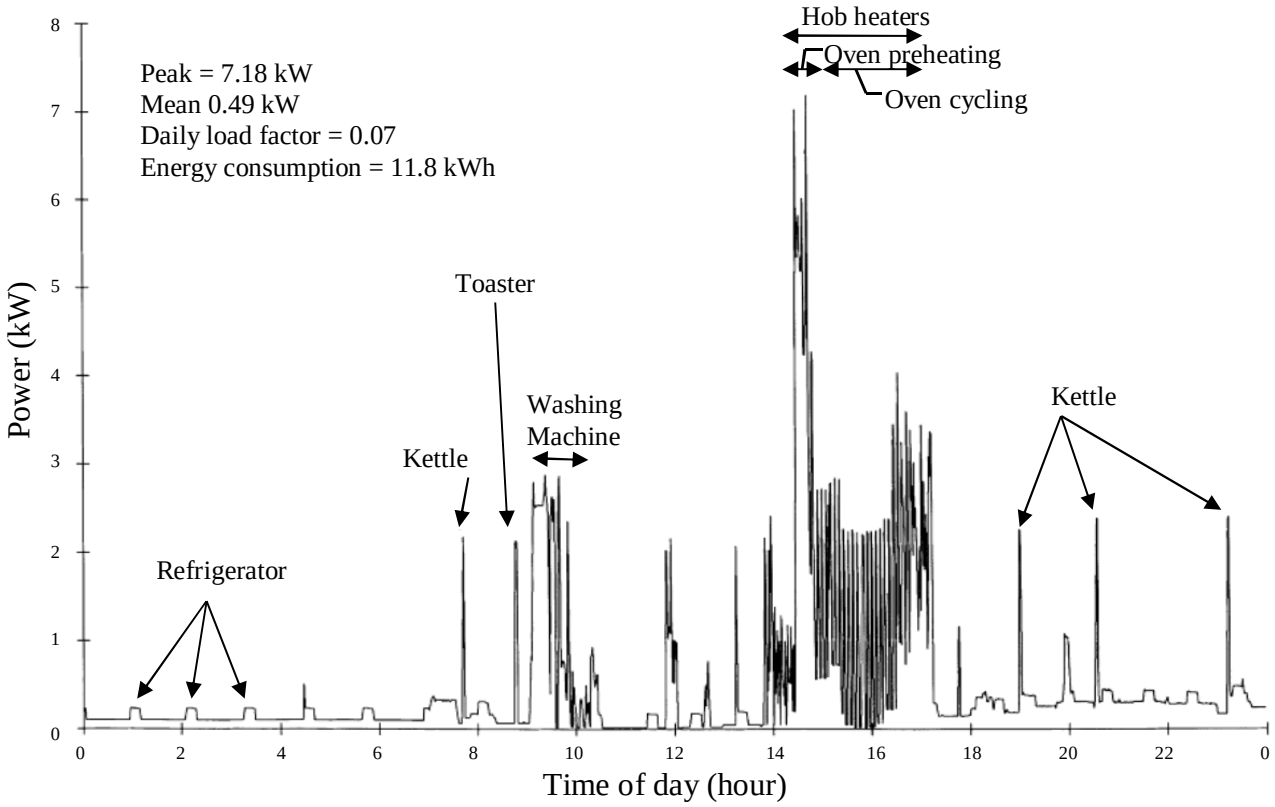


Figure 3.1 Household profile with devices identified

There were three reasons for using data from this image rather than from an existing dataset: First, data mining has already been done on this data (using NALM) showing details on electrical device usage which then expose a user's lifestyle habits and thus leak their privacy. By using this data, there is no further need to prove that privacy is being

leaked since device usage details have already been laid bare. Second, this technique, which was developed in the process of this research, has applications beyond smart meter data and can help researchers work with data only accessible in graphical format. They can then use the extracted data for testing algorithms, computing statistical data or any other kind of analysis or manipulation they may wish. Third, smart meter can offer value-added services to customers including showing them a graph with their consumption. Such a graph can be used by an attacker to mine for private information or even manipulate it in other ways. This technique provides an example of how such attacks can occur.

Conversion of the figure into actual data points was done through image processing. The developed algorithm picked values of electrical consumption from an image of a graph and stored them as numerical values for use in encryption analysis. The procedure was as follows:

1. The image of the graph was read into a two dimensional array of pixel values.
2. Using a suitable threshold value, the image was converted from colour/gray scale into a black and white binary image. Parts which are black are represented by 0, while white parts are represented by 1.
3. Pixel coordinates for the origin (X_0, Y_0) and of the top right corner of the plot area (X_{max}, Y_{max}) were located.
4. For each value of x (i.e. along the horizontal axis), edge detection was used to find where the line was located. A vertical linear search was done pixel by pixel, until the pixel value changes from white to black, then back to white. This indicates a crossing of the line and was stored as the value of y.
5. Scaling along the vertical axis was done by multiplying the obtained pixel values with the ratio of the maximum actual graph values to maximum pixel values.
6. Linear interpolation was used to scale values along the horizontal axis.

The Matlab implementation for this algorithm is shown in Appendix 3. The data obtained from this plot was used in testing the encryption algorithms.

3.3 Encryption of smart meter data

3.3.1 Key Selection and analysis

The security of symmetric ciphers is heavily dependent on the key. A key chosen in a predictable manner would compromise the security of the strongest cipher. In the case of electrical customers, millions of smart meters will need to have a key that is shared with the utility and thus a scalable solution for generating the keys is needed. It would be convenient to have the keys correspond to the meter numbers to help in managing the keys. However, the relationship between the meter number and the corresponding key should not be obvious. If it is, someone with a neighbouring meter number might find a way of decrypting the data.

The method used to generate the key *must* be kept a secret in the same way people do not disclose how they choose their passwords. Disclosure of the key generation technique would render the whole system insecure.

In this work we present the following method for use in generating the large number of keys needed. The following steps were followed:

1. The meter number was express as a string of characters
2. The message digests was computed using SHA512 (Secure Hash Algorithm – 512 bits).
3. The first 128 bits of the hash value obtained were chosen as the key (the one produced by the original meter number will henceforth be called *Key1*).

To examine the security of the above technique, the hamming distance was computed between keys produced by adjacent meter numbers. Using the meter number that follows

the one above, a key designated $Key1^+$ was produced. And using the meter number just before the original one, a key designated $Key1^-$ was produced. Since the keys generated by the above are 128 bits, it would be desirable to have about half of the bits (i.e. around 64 bits) being different.

3.3.2 RC4 Encryption/Decryption

RC4 encryption was applied to the household usage data. The steps followed were:

1. RC4, being a stream cipher, performs encryption one byte at a time. Its input thus needs to be in byte form. Each data point was converted by splitting it into a pair of bytes (i.e. a 16 bit quantity allowing a range of 0 – 65535W, which was an adequate range for the data used).
2. A key stream (which is a sequence of pseudorandom bytes) was generated using RC4 algorithm (see Appendix 1 for the details of RC4). *Key1* was used in this step.
3. XOR operation was applied between each pair of bytes from step 1 and each successive pair of bytes from step 2. This operation results in a pair of bytes serving as the ciphertext (i.e. encrypted form of the data).
4. The encrypted pairs of bytes were then merged to form a value that can be transmitted. This essentially reverses the splitting of step 1.
5. This encrypted data is then converted back to byte pairs and decrypted using the same key. This is to confirm that the data received at the utility is the same as the original once it is decrypted. Figure 3.2 is a block diagram summarizing how RC4 was applied.

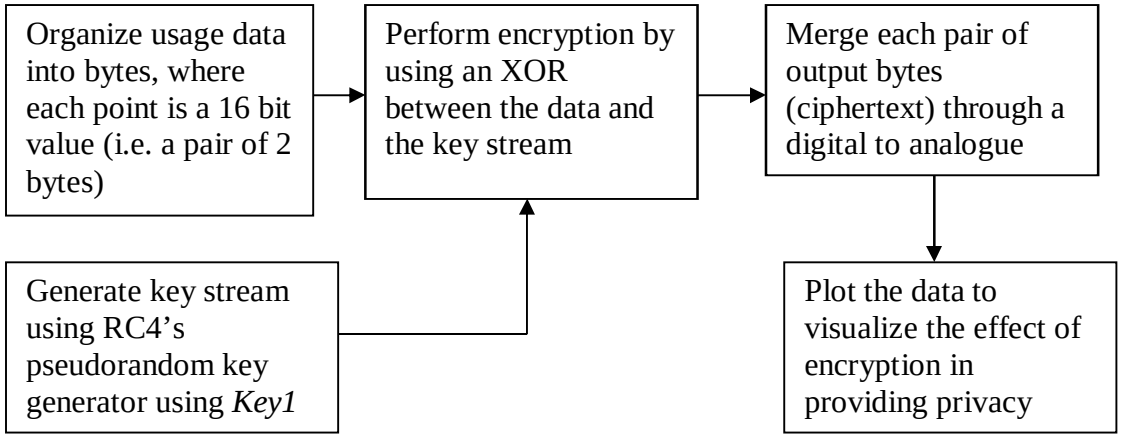


Figure 3.2 Applying RC4 to electrical usage data

3.3.3 Spritz Encryption/Decryption

A similar procedure to the one just described for RC4 was used:

1. Spritz is also a stream cipher and it encrypts data byte by byte. Each data point was thus converted by splitting it into a pair of bytes.
2. A key was fed into the "*keySetup*" function for Spritz and then encryption was done (see Appendix 2 for the details of Spritz). The same key used in RC4 (i.e. *Key1*) was used in this step, since different keys could influence the results.
3. Encrypted pairs of bytes were then merged to form a value that can be transmitted. This essentially reverses the splitting of step 1.
4. This encrypted data is then converted back to byte pairs and decrypted using the same key. This is to confirm that the data received at the utility is the same as the original once it is decrypted. Figure 3.3 is a block diagram summarizing how Spritz was applied.

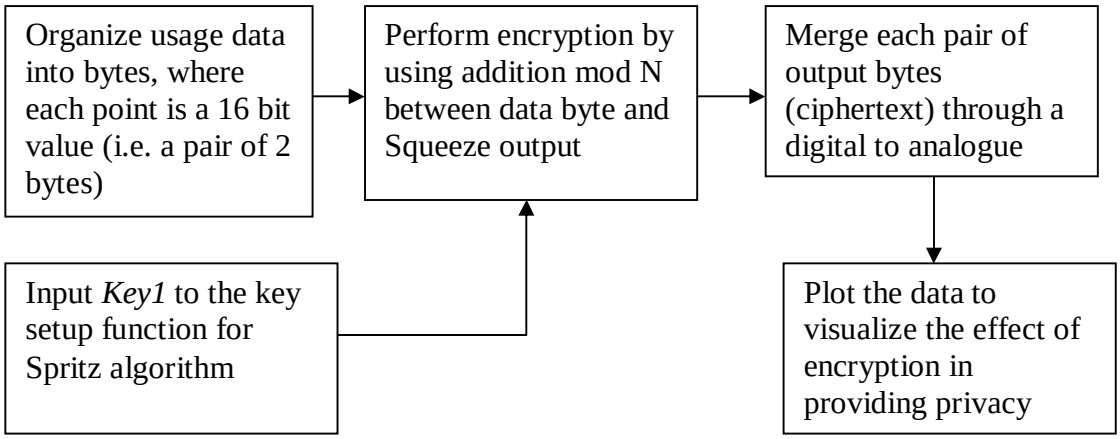


Figure 3.3 Applying Spritz to electrical usage data

3.3.4 Attempted decryption using wrong keys

One way of trying to break encryption is through brute-force. This involves attempting decryption using several possible keys and examining the results to see if the original message is recovered. If using a key close to the correct one produces results that are close to the original message, the task of an attacker would be greatly simplified. They would not have to work through a large number of keys, but they would keep improving their guess until they break the code.

To test if this vulnerability, the encrypted data was decrypted using three wrong, but conveniently chosen keys: one key was wrong on only one bit (this will be referred to as *Key2*), another was chosen as the bitwise negation of the original key (*Key3*) and the final one was generated at random (*Key4*). The results of the decryption were then plotted to see if they would produce anything close to a discernible message.

3.4 Comparing effectiveness of Spritz against RC4

3.4.1 Time taken to run RC4 and Spritz

Time taken to perform the encryption of each data point was determined using a timing function know as *tic…toc*, which gives the time taken to run code. Ten (10) runs were performed to calculate the average computation time.

3.4.2 Statistical analysis of RC4 and Spritz

Comparison of RC4 and Spritz ciphers was done through four statistical tests: the Monobit test, Poker test, Runs test and Long run test. Each test was done using the first 20,000 bits (i.e. 2,500 bytes) produced by each cipher after encrypting the electrical usage data. The 4 tests were performed as described below:

A. Monobit test – checking the number of ones in a 20,000 bits stream.
    1. The number of bits with the value "1" was counted to obtain a value *X*.
    2. The value *X*, from the total in the previous step was tested to see if it lies in the allowable limits for the Monobit test (i.e. 9725< X <10275). The test is passed if the value is within the range, and fails otherwise.
B. Poker test
    1. The 20,000 bits were stored in a 2 dimensional array of 5000 x 4 bits.
    2. The 5,000 values, each of 4 bits in length, were converted to decimal numbers ranging from 0 to 15.
    3. The number of occurrences for each value (0 to 15) was counted.
    4. *X* was computed using Equation 2.3 and the frequencies of occurrence from the previous step. *X* was tested to find if it lies in the allowable limits (i.e. 2.16<X<46.17). The test is passed if the value is within the range, and fails otherwise.
C. The runs test – examining the number of repeated 1s and 0s.

1. The length of each run (i.e. number of occurrence of identical consecutive bits) in the 20,000 bit steam was counted.
2. A 2 x 26 array called *runCount* was used to store the count of the frequency for each run length. The first row and second rows were used to store the counts for 0s and 1s respectively. The first column stored the number of times a run of 1 occurs, the second stored the number of times a run of 2, and so on until the 25^{th} column. The last column, however, was used to store the frequency for runs of 26 or more. The value 26 was chosen because the long run test (discussed below) fails if there are runs of length 26 or above.
3. The values of the array from position 6 onwards were added together (catering for the condition where 6+ repeats are examined). The final result was stored in a 2x6 array.
4. These results and compared with the ranges in Table 2.7 to see if the runs test was passed.
5. Plots were drawn to show if to see if the value of the runs test (both 0s and 1s) for each cipher was within the desired range (i.e. if it meets or fails the runs test).

D. Long run tests – are there any *runs* of length 26 or more?

1. The procedure for the runs test was used, whereby the 26^{th} column of the 2 x 26 array described above was compared to zero.
2. If the value was found to be greater than zero, then a long run was detected and the test is failed. If the value is zero, there is no long run and the test is passed.

The flowchart for the runs test and long run test is as shown in Figure 3.4

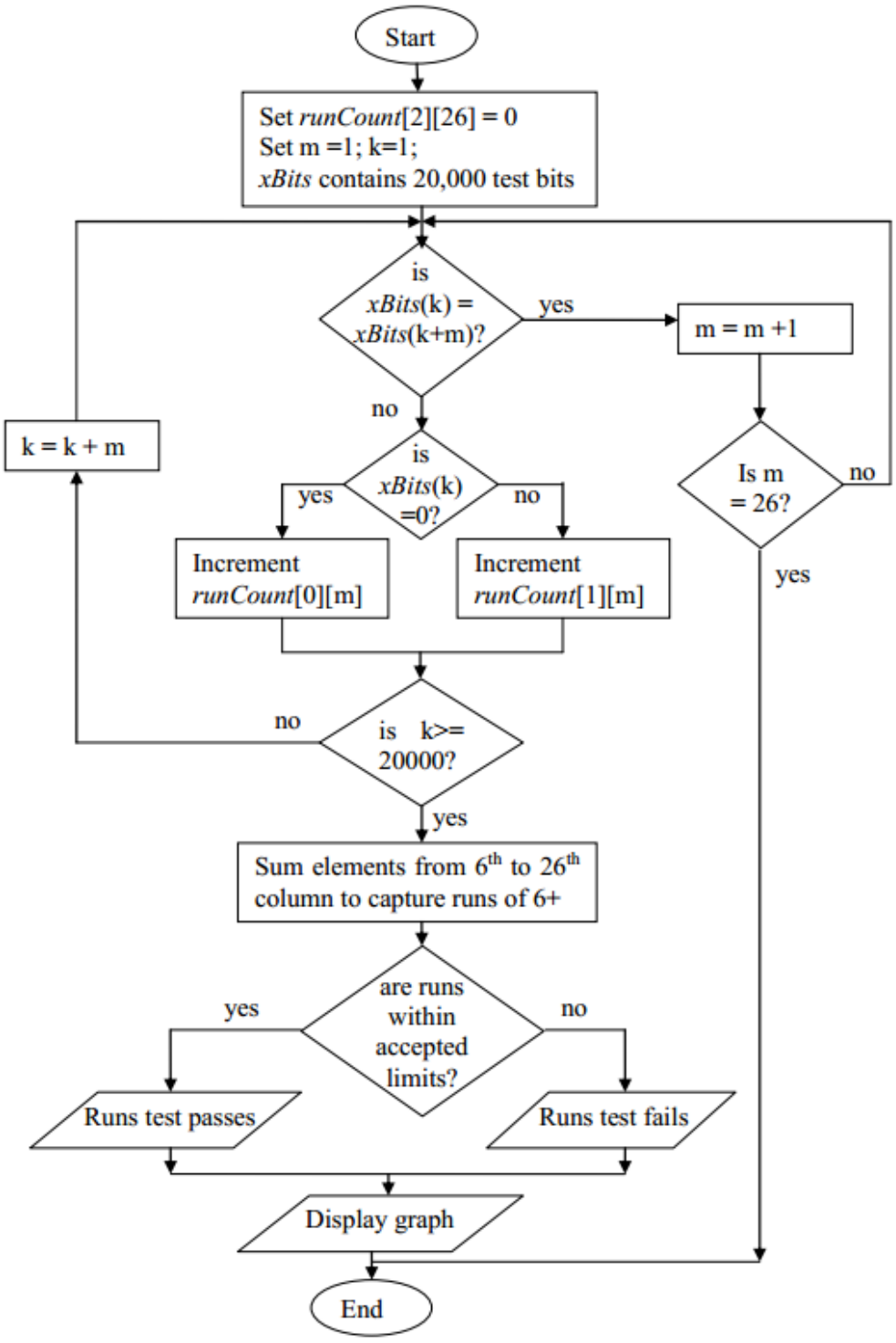


Figure 3.4 Flow chart for runs test and long run test

3.4.3 Attacks on attenuated Ciphers

An attack that works across different ciphers is brute force. All possible keys are tried until the plaintext is uncovered. In this work keys of 128 bits were proposed for the implementation of smart meters. These give 2^{128} ($\cong 3.403 \times 10^{38}$) possible options. Working through such even at a rate of 1 million keys per second would take 10^{25} years. Such a feat is outside our best computers.

In light of the above computational barrier, it would be practically impossible to compare the strength of the two ciphers when both use 128 bit keys. Instead, short keys were applied to both RC4 and Spritz and the time taken to recover the key by brute force was computed. This can be compared to testing the effect of disease causing agent (e.g. bacteria) on an animal by using its weakened version. The results can then be extrapolated from there. To compare the strength of RC4 and Spritz, smaller keys were applied to the original data. The time taken for each to be broken was then plotted. This plot would indicate which of the two ciphers is more secure from a brute force attack.

Keys were chosen ranging from 1 to 8 bits in length. They were then used to encrypt the household data using RC4 and then using Spritz. The time taken for each case was computed. A graph of time taken retrieve the key was plotted against the length of the key (in bits).

CHAPTER FOUR

RESULTS AND ANALYSIS

The following results were obtained:

4.1 Data extracted using Image Processing

The technique used for the obtaining data points from an image provided the data points as an array of values. These were then plotted to see if the shape of the original image graph was maintained. The reconstructed graph is shown in Figure 4.1.

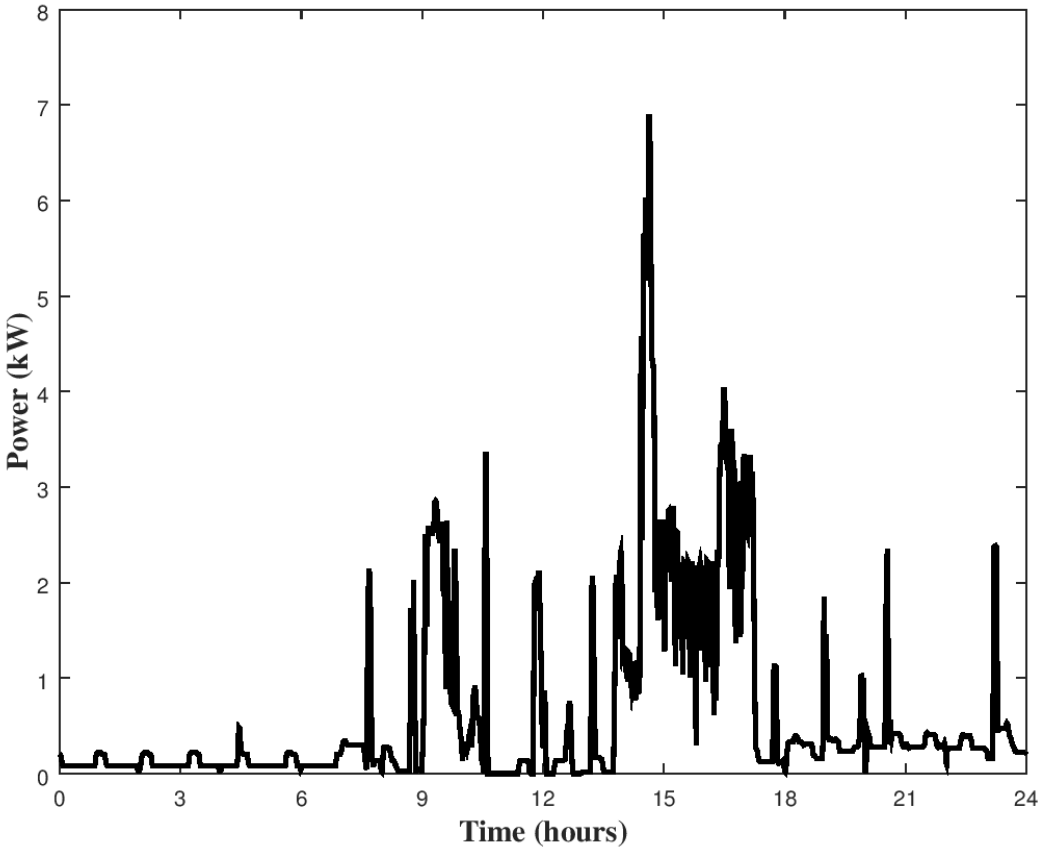


Figure 4.1 Typical household energy usage, plotted from extracting data points

The general profile was captured. The accuracy of this approach is largely dependent on the quality of the original image. An image with high noise or blurred pixels would introduce inaccuracies in the data extracted. The general shape of the original plot is maintained with the vast majority of the data points being the same in both plots. This is sufficient for the purposes of demonstrating how encryption would work with data from a household whose data can be mined using NILM.

4.2 Results for Encryption

4.2.1 Key Selection and analysis

The key was generated using a subset of the hash function for the meter number. An actual meter number was used. The first and last 3 digits of this meter number were 221…006. The result obtained by taking the first 128 bits of its hash function (using SHA512) were found to be (in hexadecimal)

$$Key1 = 0xca5c1d000455c154023829d96301c2fc \quad (4.1)$$

To check if this key is closely related to the one generated by adjacent meter numbers, a comparison was made to two other keys. Taking the meter number just after the one used, (i.e. 221…007) and the one just before (i.e. 221…005), two keys were obtained using the same technique. These were denoted as $Key1^+$ and $Key1^-$ respectively. Their values were found to be:

$$Key1^+ = 0xe1f23f98c4edc8e8b1b6ad62f1a49c19 \quad (4.2)$$

$$Key1^- = 0x58cb4141b4d158f903dec6562e962458 \quad (4.3)$$

With the above, the hamming distance, $d(\mathbf{v}_1,\mathbf{v}_2)$, was computed to see if there is a likely relationship between keys from adjacent meters.

$$d(Key1, Key1^+) = 61 \quad (4.4)$$

$$d(Key1, Key1^{-}) = 63 \quad (4.5)$$

This indicates that the difference between the bits of *Key1* and $Key1^{+}$ is ≅ 47.7% while the difference for *Key1* and $Key1^{-}$ is ≅ 49.2%. These are good results, indicating that keys generated by adjacent meter numbers are significantly different.

In addition to the above keys ($Key1^{+}$ and $Key1^{-}$), an analysis was performed on the next 100 meter numbers. Figure 4.2 shows the percentage variation in hamming distance between *Key1* and the 100 consecutive keys after it. The average value was 49.39%. This is a good sign that proposed key generation method makes it very difficult for an attacker to predict other keys, even if they have access to a key obtained from a particular meter number.

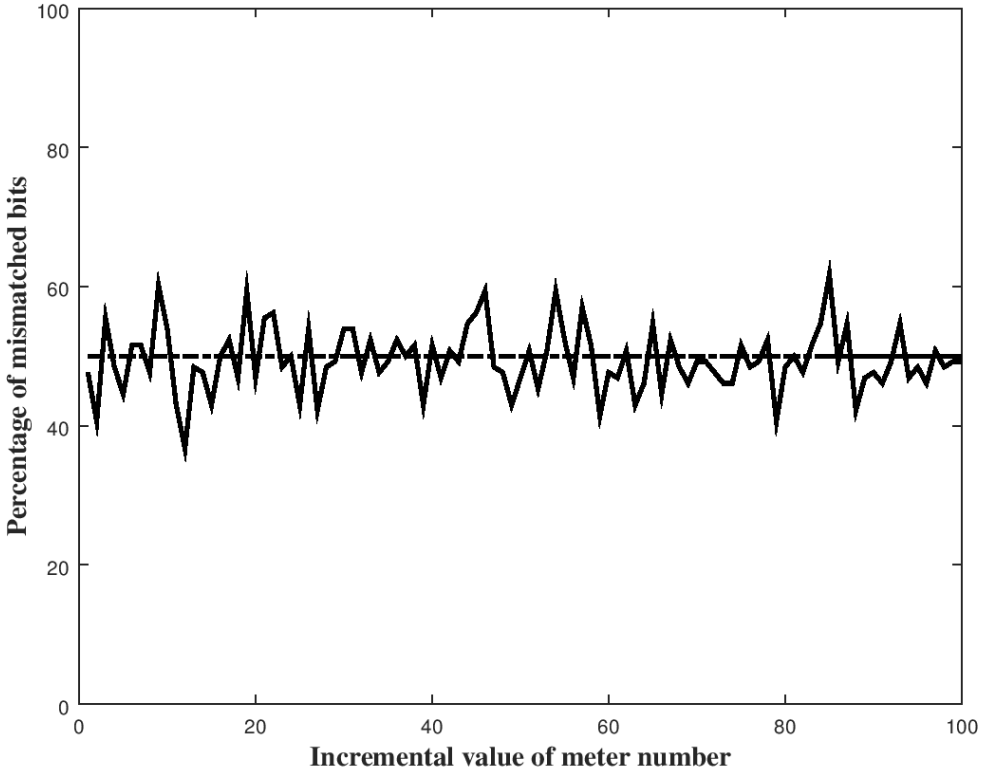


Figure 4.2 Percentage variation in hamming distance as meter number increases

4.2.2 RC4 Encryption/Decryption

When encryption was done using RC4, the plot of Figure 4.3 was obtained. Two data points are labelled: one at 6 am (600hrs) and another at 6 pm (1800hrs). These help distinguish the various random looking plots that will be encountered.

This plot shows that the data has been obscured by the encryption. Standard techniques for extracting the details of electrical usage patterns would fail.

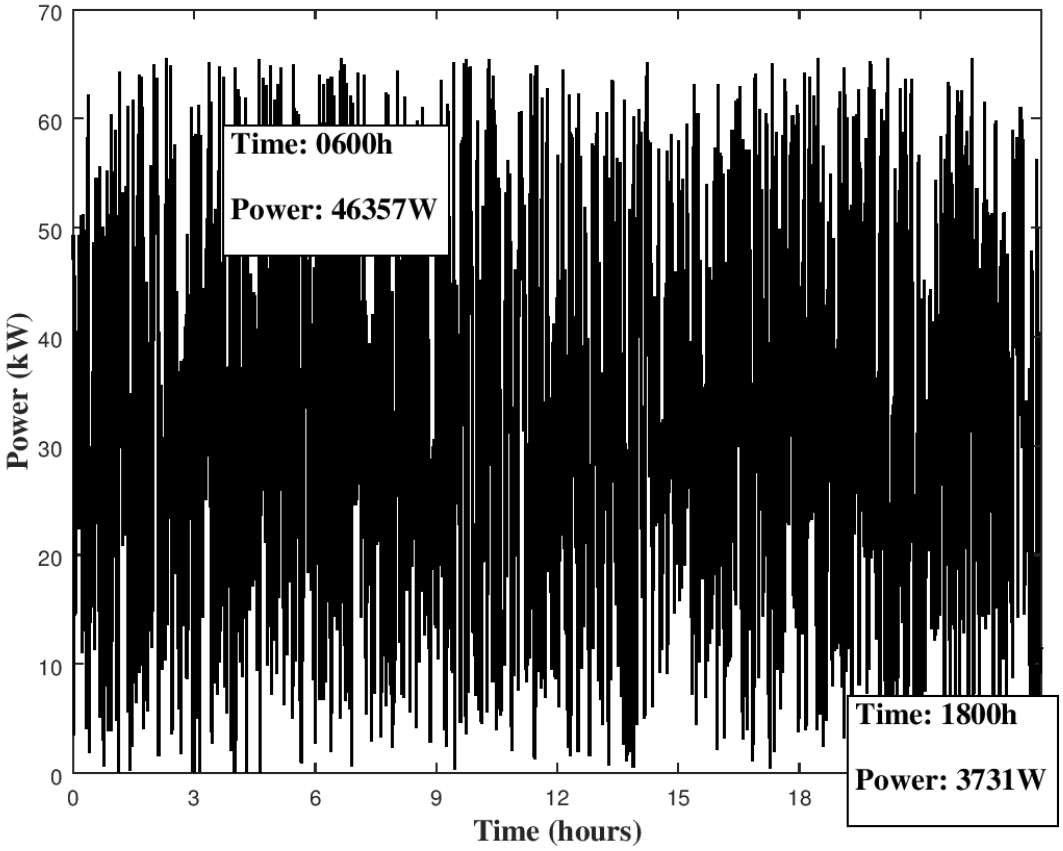


Figure 4.3 RC4 Encrypted data using *Key1*

When the data was decrypted using the same key, the original data is retrieved as shown in the plot in Figure 4.4 below. The utility company receiving the encrypted data would thus be able to find the original data on power consumption.

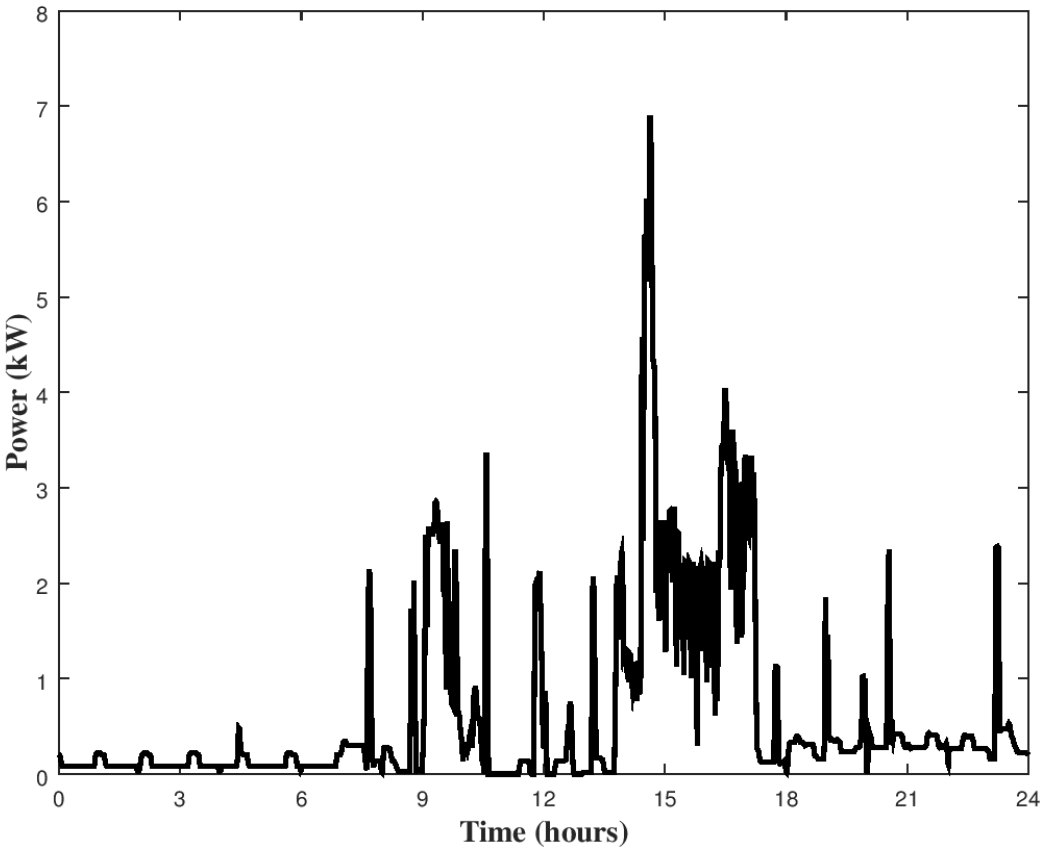


Figure 4.4 Data restored through RC4 decryption using *Key1*

4.2.3 Spritz Encryption/Decryption

When encryption was done using Spritz, and applying *Key1*, the data was also obscured as shown in Figure 4.5 An attempt to identify consumption habits using a technique such as NALM would not work as it would prior to encryption.

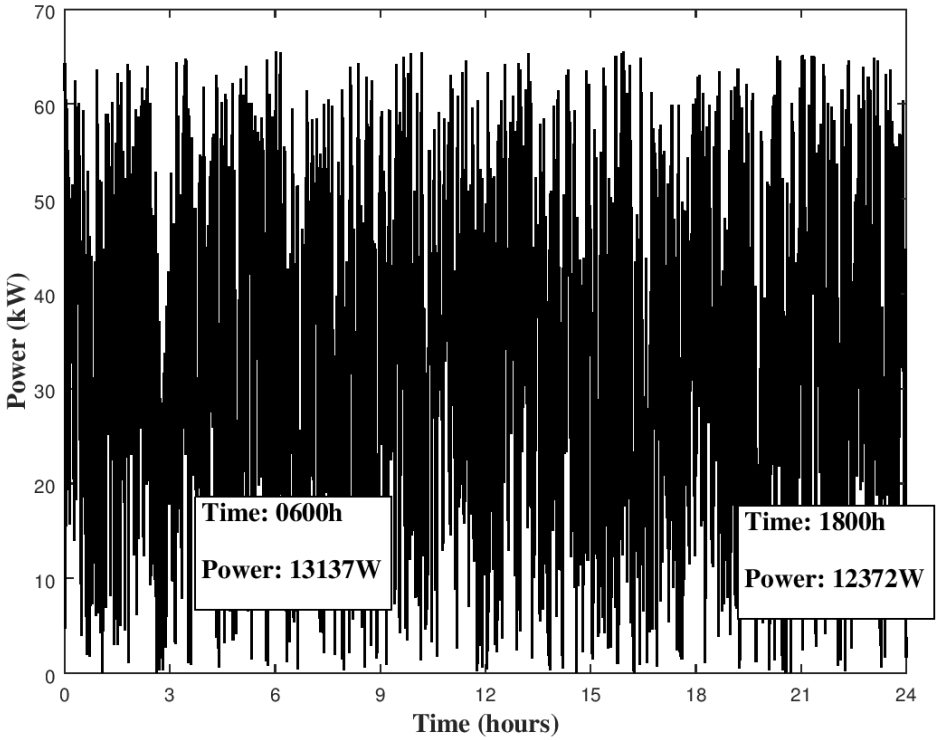


Figure 4.5 Spritz encryption using *Key1*

In both RC4 and Spritz encryption, two data points were labelled, corresponding to 6am and 6pm. These help distinguish the random figures obtained during different encryption procedures. As Table 4.1 shows, the two graphs, though both appearing random, encrypt the data differently.

Table 4.1 Test data points for RC4 and Spritz

Data source	**Time = 6am**	**Time = 6pm**
Original value (unencrypted)	38.044W	77.702W
RC4 (using *Key1*)	46.357kW	3.731 kW
Spritz (using *Key1*)	13.137kW	12.372 kW

Restored value (after decryption)	38.044W	77.702W

Decryption for Spritz using the same *Key1* also restores the original data as shown in Figure 4.6. The decrypted data was found to be identical to the original data. The process of encryption/decryption did not introduce any noise/distortion.

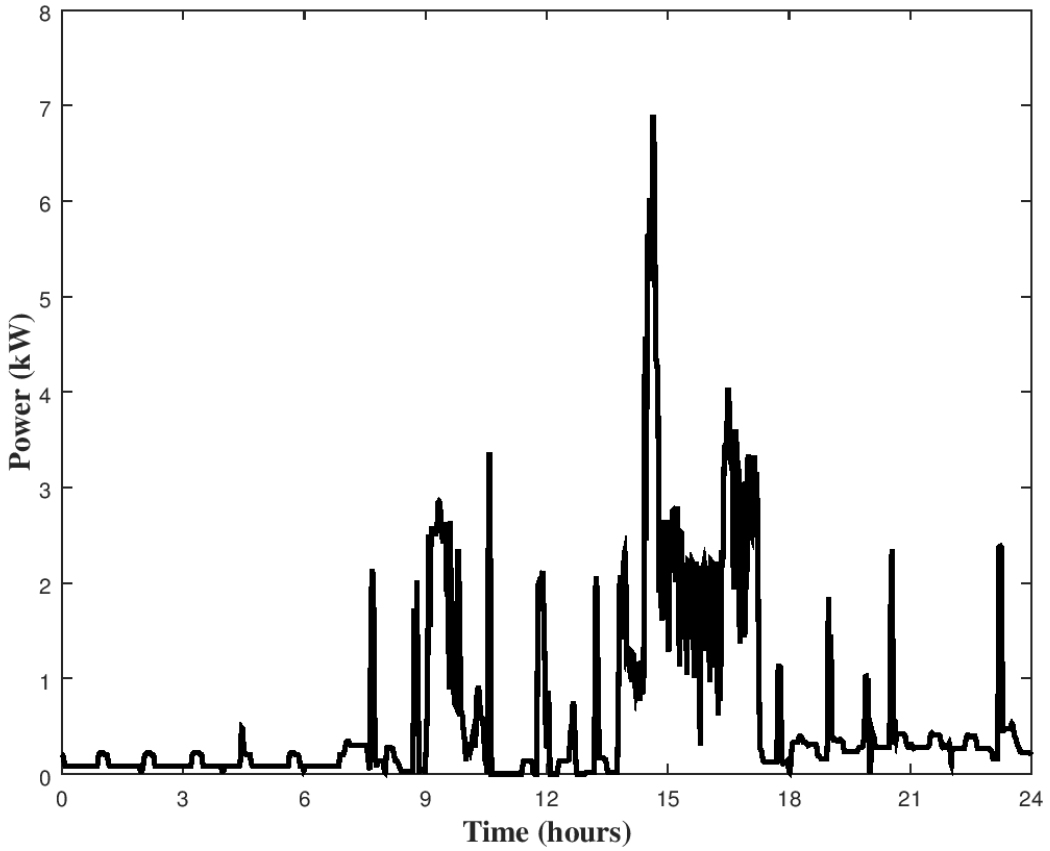


Figure 4.6 Data restored through Spritz decryption using *Key1*

4.2.4 Attempted decryption using wrong keys

Three keys were used to test what would happen if an attacker were to try guessing a key then use it to decrypt. *Key1* is the original 128 bit key used in encryption generated using the method described earlier. *Key2* was produced by changing the least significant bit of

Key1 and represents a scenario where an attacker's guess is very close to the actual key (with 127 bits correct out of 128). If the result of decryption using *Key2* was close to the original, this might reveal information to guide the attacker on whether they are getting closer.

Key3 was a key created by taking a bitwise inversion of every bit in *Key1*. Every 1 is converted to a 0 and vice versa. Its inclusion here is also to see whether it would give the attacker a hint at the correct answer. *Key4* was generated at random using a function for generating uniform distributed numbers. It represents a scenario where an attacker has made a random guess. Table 4.2 below contains the values of the correct key and the 3 incorrect ones used for these tests. The values are in hexadecimal format.

Table 4.2 Wrong keys used to attempt decryption

Key	Value in hexadecimal notation
***Key1* (correct key)**	0xCA5C1D000455C154023829D96301C2FC
***Key2* (1 wrong bit)**	0xCA5C1D000455C154023829D96301C2FD
***Key3* (Bitwise inversion)**	0x35A3E2FFFBAA3EABFDC7D6269CFE3D03
***Key4* (Random key)**	0x317A8FD94652CF9C6687C234C14DA828

The results for attempting Spritz decryption are show in Figure 4.7, Figure 4.8 and Figure 4.9 when *Key2, Key3* and *Key4* were used, respectively.

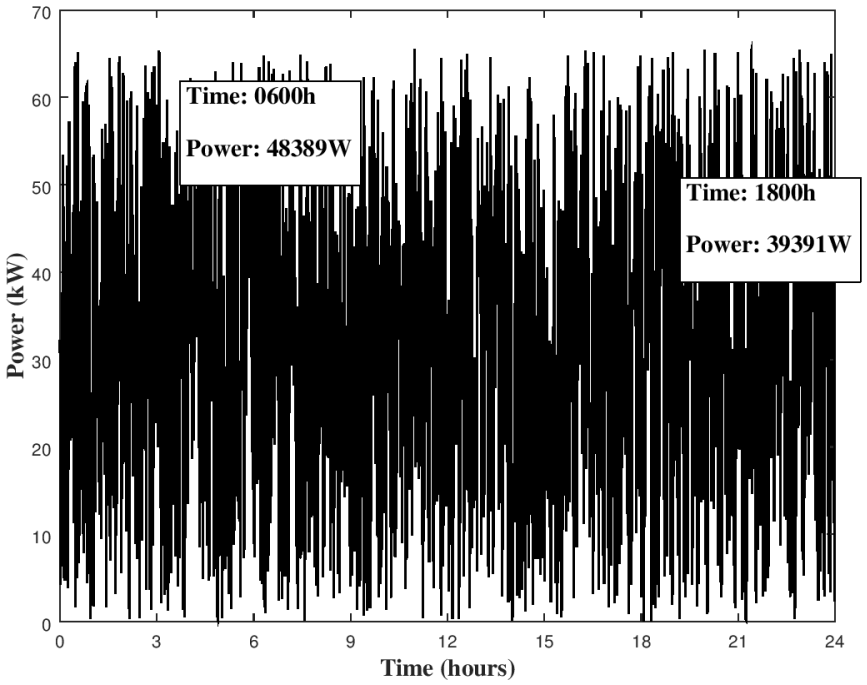


Figure 4.7 Attempted Spritz Decryption using *Key2*

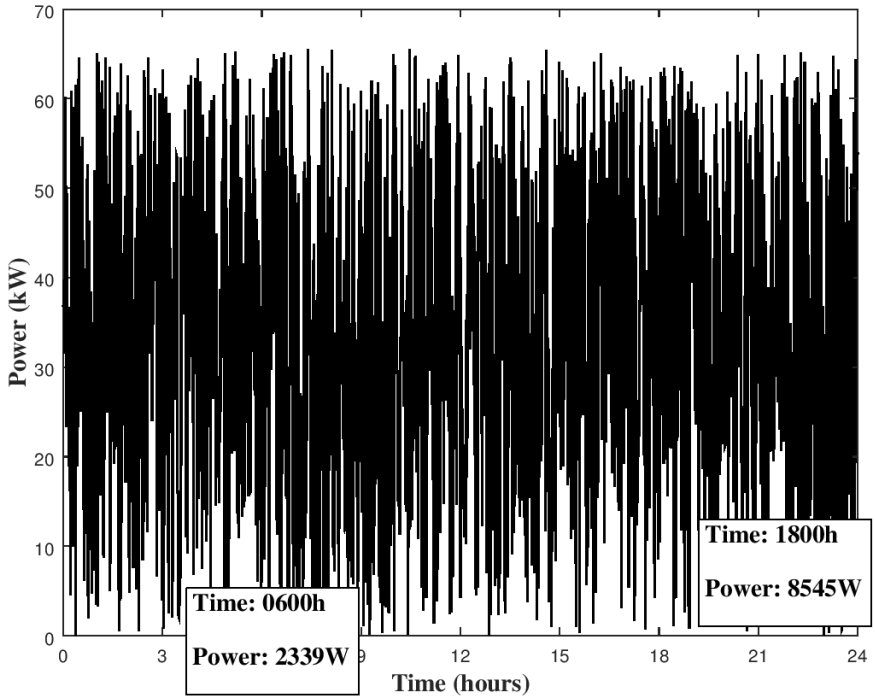


Figure 4.8 Attempted Spritz Decryption using *Key3*

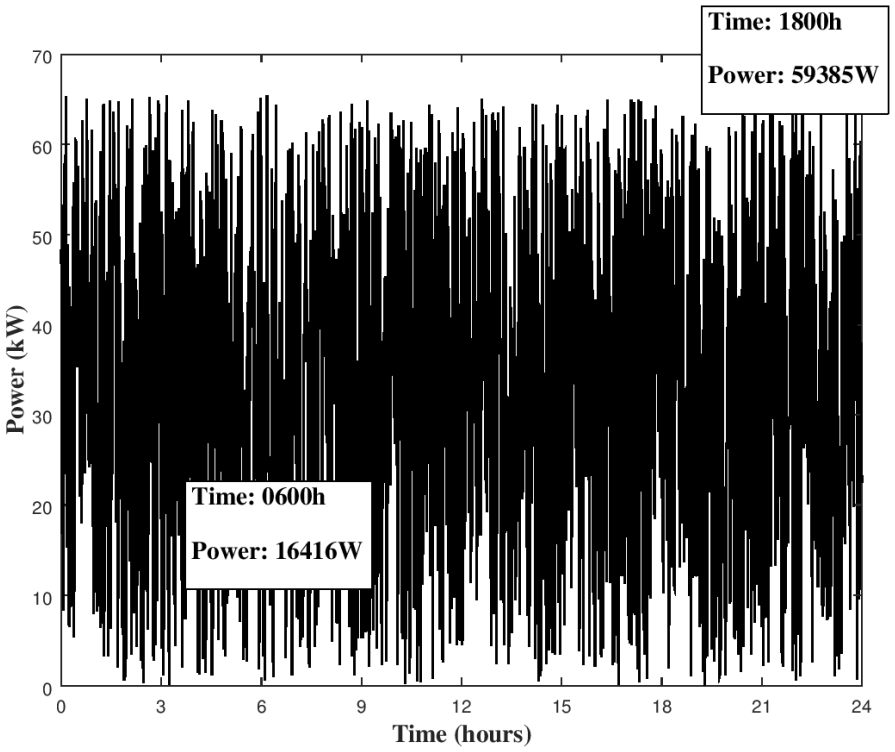


Figure 4.9 Attempted Spritz Decryption using *Key4*

What is revealed from these graphs is that wrong keys produce random looking results with no resemblance to the original data. Spritz cipher thus shows good resilience to attacks where a hacker tries to improve on their guess based on the results obtained. RC4 also exhibited the same property as shown in Figure 4.10, 4.11 and 4.12 below.

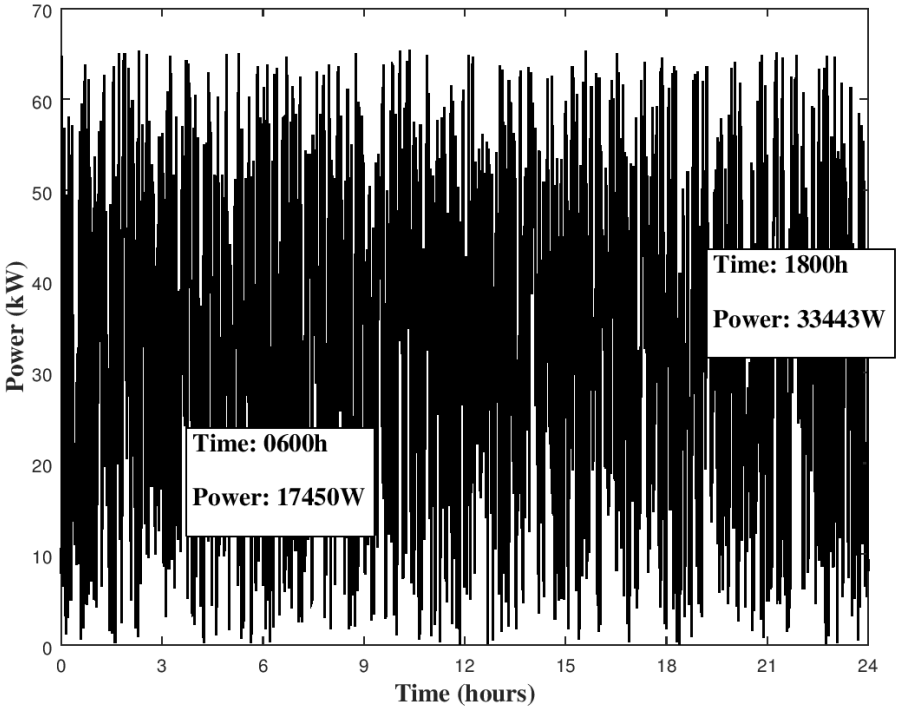


Figure 4.10 Attempted RC4 Decryption using *Key2*

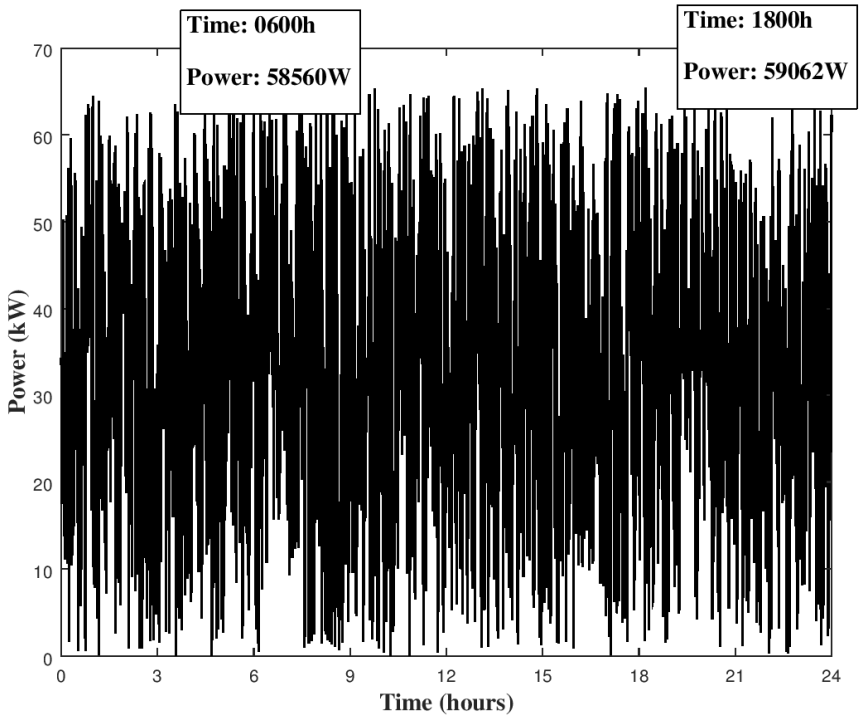


Figure 4.11 Attempted RC4 Decryption using *Key3*

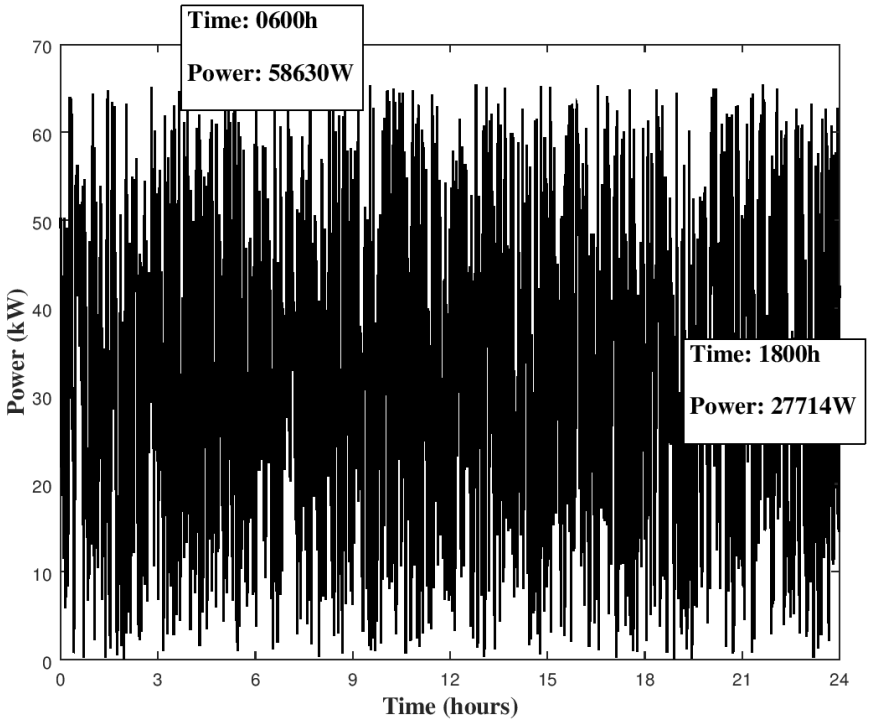


Figure 4.12 Attempted RC4 Decryption using *Key4*

4.2.5 Time Taken for RC4 and Spritz Encryption

The average time taken to encrypt one byte using RC4 encryption, done over 10 trials was found to be 0.44901 milliseconds. The 10 rounds used are shown in Table 4.3. Similarly, results for Spritz were repeated 10 times and the average found to be 0.85122 milliseconds (see Table 4.4). The ratio of these two was found to be 1.8958.

Thus, given that all factors were constant when running the tests (i.e. same key was used, code run on the same computer under the same conditions), Spritz is found to be only slightly slower than RC4. This makes it a potential candidate for use in smart meters.

The time taken for the trails was found to progressively decrease. This is probably complier optimizations occurring as the function for Spritz or RC4 is called. The values do however stabilize after about a dozen runs.

Table 4.3 Time taken for RC4 encryption on 10 repeated trials

Trail	Time (ms)	Trail	Time (ms)
1	0.71049	**6**	0.39463
2	0.57353	**7**	0.36265
3	0.56556	**8**	0.34167
4	0.48459	**9**	0.33069
5	0.42461	**10**	0.30171

Table 4.4 Time taken for Spritz encryption on 10 repeated trials

Trail	Time (ms)	Trail	Time (ms)
1	1.7452	**6**	0.62445
2	1.20127	**7**	0.59048
3	0.97932	**8**	0.59452
4	0.83937	**9**	0.59255
5	0.75241	**10**	0.59258

4.3 Results for Statistical Analysis of RC4 and Spritz

The two ciphers were compared using the 4 statistical tests described in Section 3.8. In Table 4.5, the first 50 bits (out of the 20,000 bits used in statistical testing) are given below for both RC4 and Spritz.

Table 4.5 Sample bits for testing RC4 and Spritz

Cipher	First 50 bits
RC4	11011000011001100111011110000011101011010011110100
Spritz	11010111100110000011011001111100011111110010001000

The samples above arguably appear random to a human observer, but the following results provide a more objective test for randomness based on statistical properties:

4.3.1 Monobit test

1. RC4 scored a value of X = 9,986 which lies in the range 9,725 < X < 10,275 and thus passes the test.
2. Spritz scored a value of X = 10,068 which lies in the range 9,725 < X < 10,275 and thus passes the test.

4.3.2 Poker test

1. RC4 scored a value of X = 8.2176 which lies in the range 2.16 < X < 46.17 and thus passes the test.
2. Spritz scored a value of X = 10.502 which lies in the range 2.16 < X < 46.17 and thus passes the test.

4.3.3 Runs test

1. The runs test results obtained for RC4 are given in Table 4.6 below

Table 4.6 Results for RC4 runs test

Length of run	Run for 1s	Run for 0s
1	2481	2481
2	1272	1273
3	608	608
4	335	335
5	136	137
6+	165	166

The graphs below show where the values of the runs are compared to the given limits. From Figure 4.13 and Figure 4.14, RC4 is seen to pass the runs test, since all values are within acceptable limits.

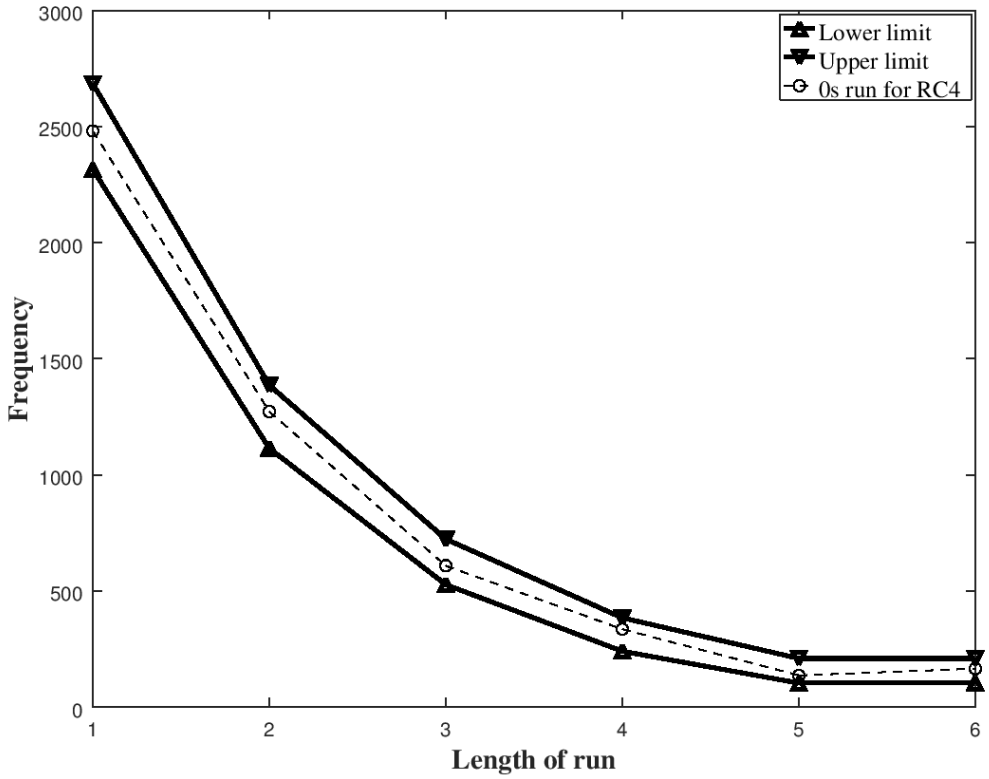


Figure 4.13 Results from 0s runs for RC4

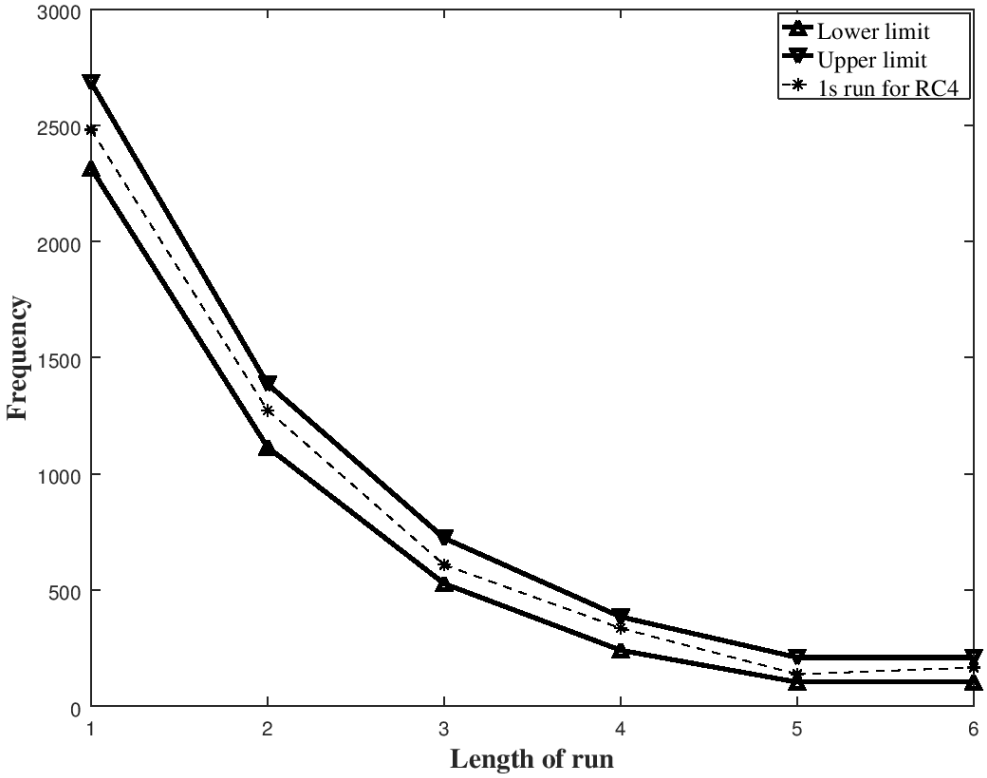


Figure 4.14 Results from 1s runs for RC4

2. The runs test results for Spritz are tabulated in Table 4.7 below:

Table 4.7 Results for Spritz runs test

Length of run	Run for 1s	Run for 0s
1	2563	2564
2	1197	1196
3	582	580
4	310	308
5	157	158
6+	165	164

The plots below help in ascertaining if the values of the Spritz runs are within acceptable limits.

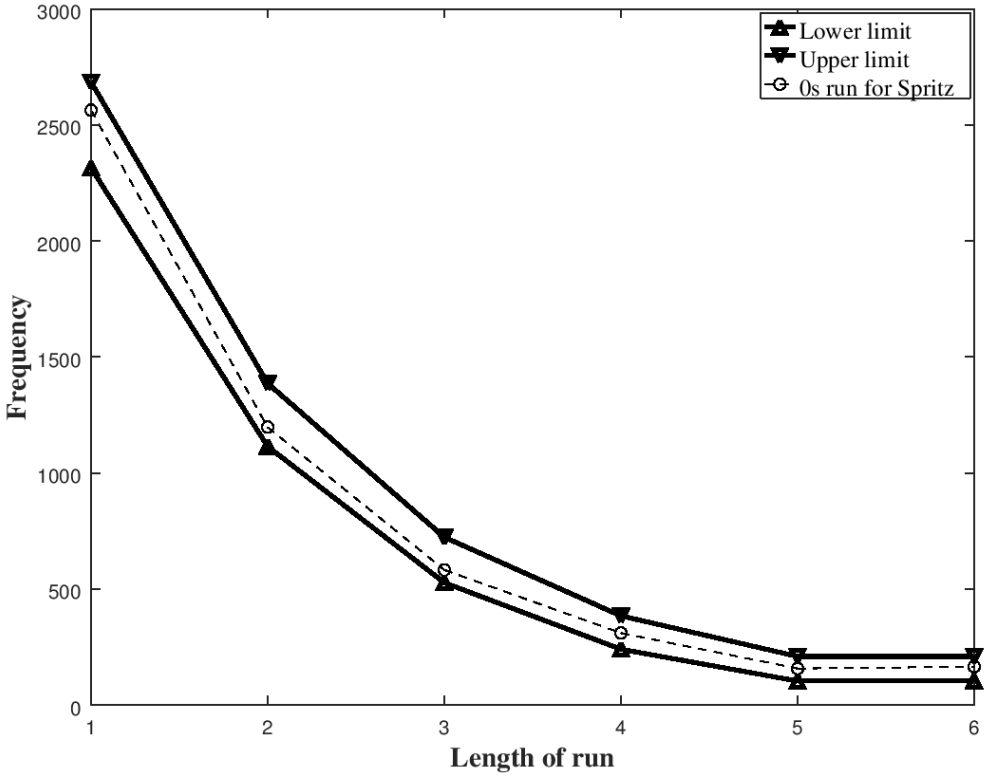


Figure 4.15 Results from 0s runs for Spritz

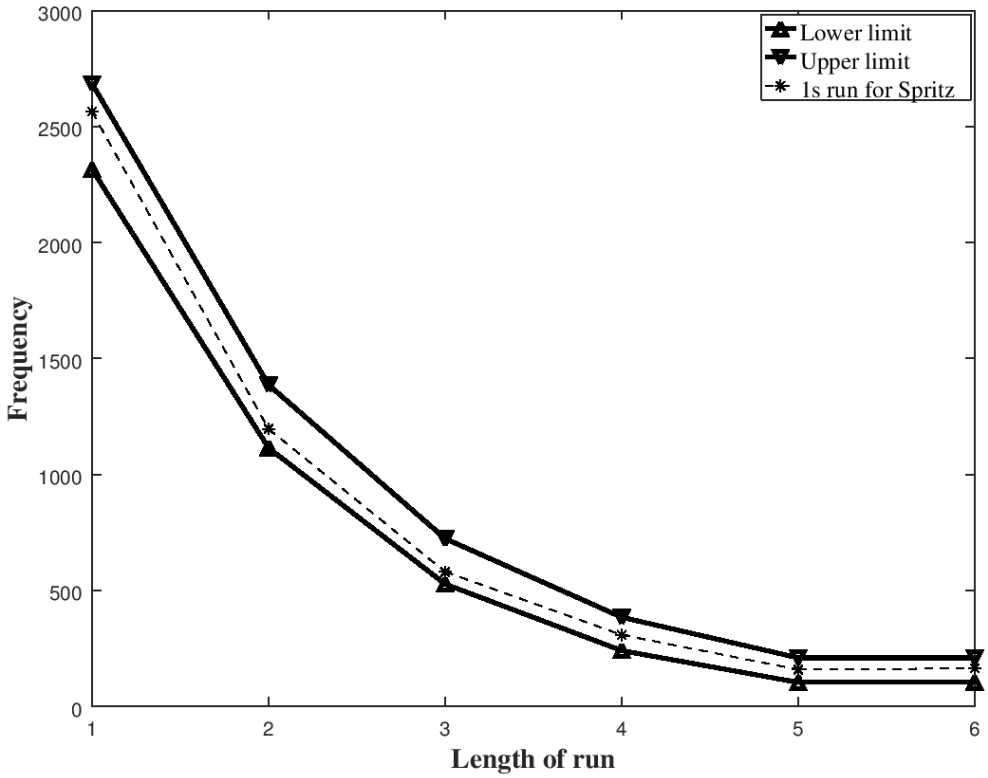


Figure 4.16 Results from 1s runs for Spritz

As Figure 4.15 and Figure 4.16 illustrate, Spritz also passes the runs tests.

4.3.4 Long run test

1. RC4 passed the long run test, having no runs of 26 bits or more.
2. Spritz also passed the long run test.

4.3.5 Conclusion from Statistical Test results

From the four statistical tests above, both ciphers pass the criteria. Since these tests are based on samples, it is not possible to conclude which cipher is better. All that can be said

at this point is that both meet the bare minimum requirement for randomness. The following test will help uncover which of the two ciphers is better.

4.4 Attacks on attenuated Ciphers

Keys were chosen ranging from 1 to 8 bits in length. They were then used to encrypt the household data using RC4 and then using Spritz. The time taken for each case was computed. A graph of time taken to retrieve the key was plotted against the length of the key (in bits).

As can be seen from **Figure 4.17**, the time taken to retrieve the key is lower in RC4 than it is in Spritz. For the case of an 8 bit key, the time needed for RC4 was 52.336 seconds while the time for Spritz 167.655 seconds. This results in a ratio of

$$\frac{167.655}{52.336} = \mathbf{3.203}$$

This means that an attacker whose goal is to break RC4 has a much easier task than one who is attacking Spritz. This test reveals Spritz to be a stronger algorithm than RC4.

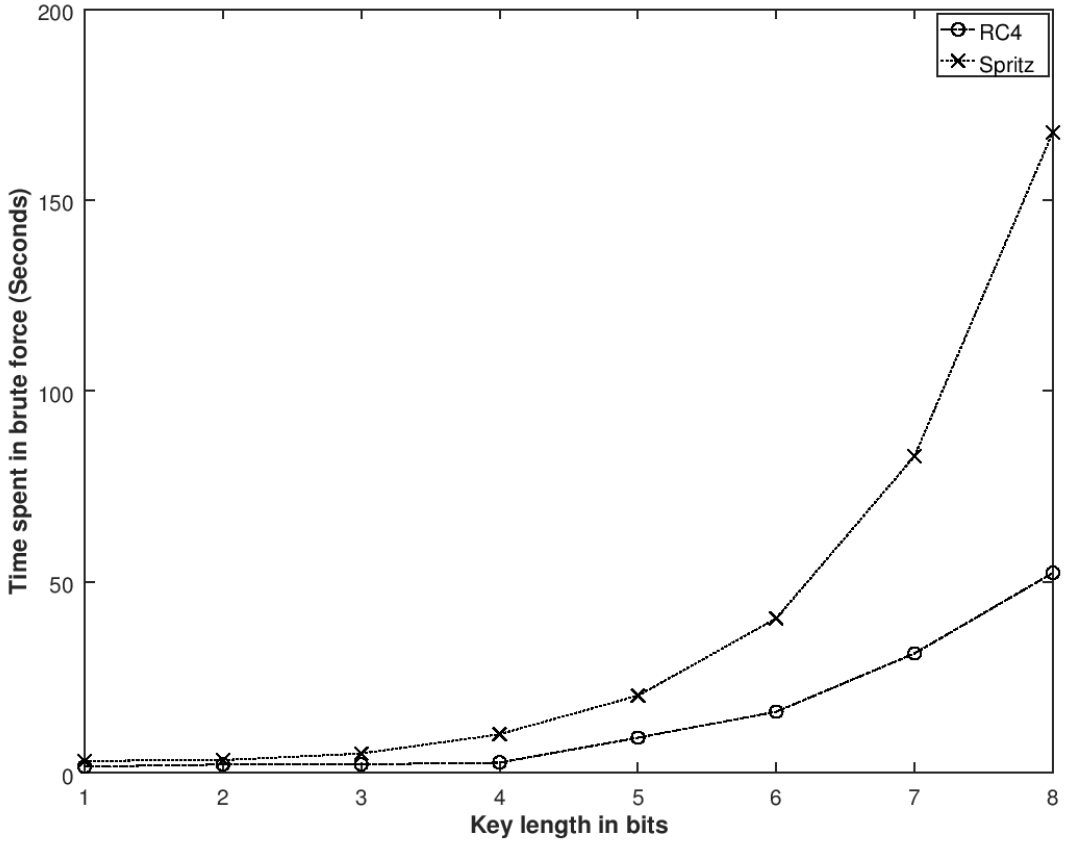


Figure 4.17 Comparison of time taken to break RC4 and Spritz using brute force

As the key length becomes longer, the time needed to break both will increase. However, Spritz will still require a longer time to break. Thus an attacker with resources that barely manage to break RC4 will still not be able to break Spritz.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

It would be a very costly oversight to ignore user privacy in the deployment of Smart Grid. As our world becomes more interconnected and cyber threats increase in both complexity and frequency, people are more aware of the impact of privacy leakage. Smart Grid adoption, with all its potential benefits, would be hindered if privacy concerns are not addressed.

This thesis recommends the use of Spritz encryption to address the problem of privacy in smart meters. It provides an analysis of how it would perform when applied to electrical usage data in the context of Smart Grid. From the results obtained, encryption is seen to obscure the details of household data from a human observer. Data mining algorithms such as NILM, which extract electrical device usage information, would fail since previous patterns are no longer visible. Statistical tests further reveal that the obscured graphs have good random properties. Details of usage patterns are hidden and thus the privacy of a user is maintained.

The popular RC4 cipher was used to benchmark the performance of Spitz. The two ciphers were compared in terms of speed performance and security. The performance on encryption was found to be in the same order of magnitude (0.44901 ms for RC4 and 0.85122 ms for Spritz). Given that RC4 has been tested and deployed in an actual smart meter (though with significant security weaknesses), it is reasonable to assume that Spritz would be able to work well given the difference in this case is only 0.4022 ms for each encryption.

Spritz was found to offer better security than RC4 both by the tests done on weakened keys and by its design. Time taken to break an 8 bit key was 3.203 times longer for Spritz than RC4, making it harder for an attacker to successfully break the system. By design, Spritz does not share some of RC4's major shortcomings such as existence of weak keys and long term and short term biases. Spritz would thus be a better option for smart meter security.

By presenting Spritz as a viable option for providing smart meter security, this work enables algorithm agility. Having multiple options of suitable encryption schemes for smart meters is crucial for two main reasons. First, security researchers are constantly looking for weaknesses in algorithms and history shows that it is usually a matter of time before a scheme is found to be insecure – or at significantly weaker. These research results are eventually converted to tools accessible by many people. Secondly, technological advances lead to increased computational speeds providing attackers have faster machines to work with. What is considered unbreakable can within a few years be within the reach of attackers with extra computational power. Therefore, having multiple tested and viable encryption schemes makes replacement easier to implement when the current schemes are broken. It is better to have tested several options and have them ready for deployment, than to get into panic the day a researcher publishes a paper, detailing how they broke smart meter encryption.

5.2 Thesis Contributions

This thesis recommends the use of Spritz encryption for use in smart meters. This has not been reported in published literature before. The use of Spritz is supported by analysis of its performance in terms of speed and security. This work also presents the use of smaller (attenuated) keys to compare the strengths of two ciphers.

A key generation method is also proposed which would help with the production of a large number of cryptographic keys. The method produced keys from neighbouring meters which were distinct, with around half the bits being different (an average of 49.39% over 100 keys). This was a desirable result, indicating a great unlikelihood of being able to compute the key of a neighbouring meter. This would prevent a cascade of failure if an attacker happens to obtain one key. It is important to for anyone wishing to generate keys for practical use to modify this method since is also accessible to attackers. The use of hamming distance to assess neighbouring keys would be a helpful analysis for examining the modified method.

This thesis also presents an approach for extracting data points from an image of a graph. This method can be applied in obtaining data from secondary sources in which details of the drawn plots are absent. It would save researchers the time and hassle needed to pick out data points manually (e.g. by using a ruler and a pencil to get adequate samples). Greater accuracy can be obtained at a much faster rate while, providing more accurate results than manual extraction. Once data has been extracted, it can be used in various ways such as testing algorithms, examining a hypothesis, manipulating the data, presenting it in other formats, finding statistical quantities such as mean and so on.

5.3 Recommendations

This work focused on solving the problem of confidentiality, through the use of encryption. There are however other security features needed in further protecting a system, such as providing integrity and authentication. These would ensure that the data has not been altered and is from the correct sender. Spritz algorithm has other (optional) features such as hash functions and Message Authentication Codes (MACs) which can enhance the security of the system. Spritz hash function can be used to update the key when needed. Research on performance with the introduction of these features would be needed to assess their suitability in smart meters.

Further work can be done on matters related to the cryptographic keys. The key is a crucial part of how secure a cipher is in practical applications. More techniques of key generation can be developed. Aspects of how the key can be shared and possibly updated (i.e. key freshness) can be examined.

Though the solution presented in this work thwarts the efforts of an eavesdropper trying to snoop on the data as it is being transmitted, it does not protect users' privacy from utility companies. The establishment of clear legal policies of handling this confidential customer data would be needed to protect customers' interests. Such policies would deal with questions of who owns the data, proper consent before it is used, restricting resale of data, and so on.

Finally, an implementation of this algorithm can be done a microchip installed in a smart meter. Such a chip would have slower processing speed than a computer, but its implementation would involve fewer overheads than the simulation used in this work. An investigation of its performance on a chip would provide the final verdict on its effectiveness in smart meters.

REFERENCES

[1] X. Fang, S. Misra, G. Xue, and D. Yang, "Smart Grid – The New and Improved Power Grid: A Survey," *IEEE Commun. Surv. Tutorials*, vol. Preprint, pp. 1–37, 2011.

[2] W. Wang and Z. Lu, "Cyber security in the Smart Grid: Survey and challenges," *Comput. Networks*, vol. 57, no. 5, pp. 1344–1371, 2013.

[3] R. L. Rivest and J. C. N. Schuldt, "Spritz — a spongy RC4-like stream cipher and hash function," *IACR Cryptol. ePrint Arch.*, 2014.

[4] W. Kleiminger, C. Beckel, T. Staake, and S. Santini, "Occupancy Detection from Electricity Consumption Data," *Proc. 5th ACM Work. Embed. Syst. Energy-Efficient Build.*, pp. 1–8, 2013.

[5] K. Kursawe and C. Peters, "Structural weaknesses in the open smart grid protocol," *Proc. - 10th Int. Conf. Availability, Reliab. Secur. ARES 2015*, pp. 1–10, 2015.

[6] P. Jovanovic and S. Neves, "Practical Cryptanalysis of the Open Smart Grid Protocol Dumb Crypto in Smart Grids Smart Grids," *Int. Work. Fast Softw. Encryption*, pp. 297–316, 2015.

[7] F. R. Yu, P. Zhang, W. Xiao, and P. Choudhury, "Communication Systems for Grid Integration of Renewable Energy Resources," *IEEE Netw.*, vol. 25, no. 5, pp. 0–21, 2010.

[8] Z. Fan, G. Kalogridis, C. Efthymiou, M. Sooriyabandara, and J. McGeehan, "The New Frontier of Communications Research : Smart Grid and Smart Metering," *Proc. 1st Int. Conf. Energy-Efficient Comput. Netw.*, pp. 115–118, 2010.

[9] M. Kezunovic, J. D. McCalley, and T. J. Overbye, "Smart grids and beyond: Achieving the full potential of electricity systems," *Proc. IEEE*, vol. 100, pp. 1329–1341, 2012.

[10] M. Harvey, D. Long, and K. Reinhard, “Visualizing NISTIR 7628, Guidelines for smart grid cyber security,” in *Power and Energy Conference at Illinois (PECI)*, 2014, pp. 1–8.

[11] B. D. S. Callaway and I. A. Hiskens, “Achieving Controllability of Electric Loads,” *Proc. IEEE*, vol. 99, no. 1, pp. 184–199, 2011.

[12] G. Strbac, “Demand-side management : benefits and challenges,” *Energy Policy*, vol. 36, no. 12, pp. 4419–4426, 2008.

[13] Z. Fan, “Distributed Demand Response and User Adaptation in Smart Grids,” *Int. Symp. Integr. Netw. Manag.*, pp. 726–729, 2011.

[14] I. S. Bayram, G. Michailidis, M. Devetsikiotis, F. Granelli, and S. Bhattacharya, “Smart Vehicles in the Smart Grid : Challenges , Trends , and Application to the Design of Charging Stations,” *Control Optim. Methods Electr. Smart Grids*, pp. 133–145, 2012.

[15] W. Wang, Y. Xu, and M. Khanna, “A survey on the communication architectures in smart grid,” *Comput. Networks*, vol. 55, no. 15, pp. 3604–3629, 2011.

[16] I. Rouf, H. Mustafa, M. Xu, W. Xu, R. Miller, and M. Gruteser, “Neighborhood watch: Security and privacy analysis of automatic meter reading systems,” *Proc. 2012 ACM Conf. Comput. Commun. Secur.*, pp. 462–473, 2012.

[17] S. Galli, A. Scaglione, and Z. Wang, “For the Grid and Through the Grid : The Role of Power Line Communications in the Smart Grid,” *Proc. IEEE*, vol. 99, no. 6, pp. 998–1027, 2011.

[18] A. Molina-Markham, P. Shenoy, K. Fu, E. Cecchet, and D. Irwin, “Private memoirs of a smart meter,” *Proc. 2nd ACM Work. Embed. Sens. Syst. Energy-Efficiency Build.*, pp. 61–66, 2010.

[19] E. L. Quinn, “Smart metering and privacy: Existing laws and competing policies,” *Soc. Sci. Res. Netw.*, 2009.

[20] G. W. Hart, “Nonintrusive Appliance Load Monitoring,” *Proc. IEEE*, pp. 1871–72, 1992.

[21] Z. Fan, P. Kulkarni, S. Gormus, C. Efthymiou, G. Kalogridis, M. Sooriyabandara, Z. Zhu, S. Lambotharan, and W. H. Chin, "Smart Grid Communications : Overview of Research Activities," *IEEE Commun. Surv. Tutorials*, vol. 15, pp. 21–38, 2013.

[22] The Smart Grid Interoperability Panel, "Guidelines for Smart Grid Cyber Security: Vol. 1, Smart Grid Cyber Security Strategy, Architecture, and High-Level Requirements," 2010.

[23] S. S. S. R. Depuru, L. Wang, and V. Devabhaktuni, "Smart meters for power grid: Challenges, issues, advantages and status," *Renew. Sustain. Energy Rev.*, vol. 15, no. 6, pp. 2736–2742, 2011.

[24] M. R. Asghar, G. Dan, D. Miorandi, and I. Chlamtac, "Smart Meter Data Privacy: A Survey," *IEEE Commun. Surv. Tutorials*, pp. 1–1, 2017.

[25] D. Koo, Y. Shin, and J. Hur, "Privacy-Preserving Aggregation and Authentication of Multi-Source Smart Meters in a Smart Grid System," *Appl. Sci.*, vol. 7, no. 10, p. 1007, 2017.

[26] A. Reinhardt, G. Konstantinou, D. Egarter, and D. Christin, "Worried About Privacy? Let Your PV Converter Cover Your Electricity Consumption Fingerprints," *2015 IEEE Int. Conf. Smart Grid Commun.*, 2015.

[27] A. Humayed, J. Lin, F. Li, and B. Luo, "Cyber-Physical Systems Security—A Survey," *IEEE Internet Things J.*, vol. 4, no. 6, pp. 1802–1831, 2017.

[28] M. La Polla, F. Martinelli, and D. Sgandurra, "A Survey on Security for Mobile Devices," *IEEE Commun. Surv. Tutorials*, 2013.

[29] Y. Liu, S. Hu, and T. Y. Ho, "Vulnerability assessment and defense technology for smart home cybersecurity considering pricing cyberattacks," in *IEEE/ACM International Conference on Computer-Aided Design, Digest of Technical Papers, ICCAD*, 2015, vol. 2015-Janua, no. January, pp. 183–190.

[30] E. Ancillotti, R. Bruno, and M. Conti, “The role of communication systems in smart grids: Architectures, technical solutions and research challenges,” *Computer Communications*, vol. 36, no. 17–18. pp. 1665–1697, 2013.

[31] K. Tazi, F. Abdi, and M. F. Abbou, “Review on cyber-physical security of the smart grid: Attacks and defense mechanisms,” in *Proceedings of 2015 International Renewable and Sustainable Energy Conference (IRSEC)*, 2015, pp. 1–6.

[32] M. Badra and S. Zeadally, “An improved privacy solution for the smart grid,” *Int. J. Netw. Secur.*, vol. 18, no. 3, pp. 529–537, 2016.

[33] K. Jain, “Security Based on Network Topology Against the Wiretapping Attack,” *IEEE Wirel. Commun.*, no. February, pp. 68–71, 2004.

[34] C. V Wright, S. E. Coull, F. Monrose, and C. Hill, “Traffic Morphing : An Efficient Defense Against Statistical Traffic Analysis,” in *Proceedings of ISOC Network and Distributed System Security Symposium (NDSS)*, 2009.

[35] E. L. Quinn, *Privacy and the New Energy Infrastructure*, no. 09. 2009.

[36] O. Parson, S. Ghosh, M. Weal, and A. Rogers, “Non-Intrusive Load Monitoring Using Prior Models of General Appliance Types,” in *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012, pp. 356–362.

[37] G. Eibl and D. Engel, “Influence of data granularity on smart meter privacy,” *IEEE Trans. Smart Grid*, vol. 6, no. 2, pp. 930–939, 2015.

[38] Lisovich, M. A., D. K. Mullingan, and S. B. Wicker, “Inferring Personal Information from Demand-Response Systems,” *IEEE Secur. Priv.*, vol. 8, no. 1, 2010.

[39] U. Greveler, P. Glösekötterz, B. Justusy, and D. Loehr, “Multimedia content identification through smart meter power usage profiles,” in *Proceedings of the International Conference on Information and Knowledge Engineering (IKE)*, 2012.

[40] H. Abelson, K. Ledeen, and H. Lewis, *Blown to Bits: your life, liberty, and happiness after the digital explosion*. Addison-Wesley Professional, 2008.

[41] G. Kalogridis, C. Efthymiou, S. Z. Denic, T. a. Lewis, and R. Cepeda, "Privacy for smart meters: Towards Undetectable Appliance Load Signatures," *First IEEE Int. Conf. Smart Grid Commun.*, pp. 232–237, 2010.

[42] W. Yang, N. Li, Y. Qi, W. Qardaji, S. McLaughlin, and P. McDaniel, "Minimizing private data disclosures in the smart grid," *Proc. 2012 ACM Conf. Comput. Commun. Secur.*, pp. 415–427, 2012.

[43] A. Rial and G. Danezis, "Privacy-preserving smart metering," *Proc. 10th Annu. ACM Work. Priv. Electron. Soc.*, pp. 49–60, 2011.

[44] C. Efthymiou and G. Kalogridis, "Smart Grid Privacy via Anonymization of Smart Metering Data," *2010 First IEEE Int. Conf. Smart Grid Commun.*, pp. 238–243, 2010.

[45] C. Paar and J. Pelzl, *Understanding Cryptography: a textbook for students and practitioners*. Springer Science & Business Media, 2009.

[46] W. Stallings, *Cryptography and Network Security Principles and Practice*, 5th ed. Prentice Hall, 2011.

[47] A. Menezes, P. van Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*, vol. 19964964. CRC Press, 1996.

[48] B. Schneier, *Applied cryptography: Protocols, algorithm, and source code in C*. John Wiley & Sons, Inc, 1996.

[49] F. PUB., "Secure Hash Standard," 2012.

[50] W. Stallings, *Data and computer communications*, 8th ed. Prentice Hall, 2007.

[51] A. Maximov and D. Khovratovich, "New State Recovery Attack on RC4," *Proc. CRYPTO 2008*, vol. 5157, pp. 297–316, 2008.

[52] Protocol, "ETSI GS OSG 001: Open Smart Grid Protocol (OSGP)," vol. v1. 2012.

[53] E. Tews, R.-P. Weinmann, and A. Pyshkin, "Breaking 104 bit WEP in less than 60 seconds," *Inf. Secur. Appl.*, pp. 188–202, 2007.

[54] S. Vaudenay and M. Vuagnoux, “Passive – Only Key Recovery Attacks on RC4,” *Int. Work. Sel. areas Cryptogr.*, pp. 344–359, 2007.

[55] S. Banik and T. Isobe, “Cryptanalysis of the Full Spritz Stream Cipher,” *Int. Conf. Fast Softw. Encryption*, pp. 63–77, 2016.

[56] A. Roos, “A class of weak keys in the RC4 stream cipher,” 1995.

[57] S. Sen Gupta, S. Maitra, G. Paul, and S. Sarkar, “(Non-)random sequences from (Non-)random permutations - Analysis of RC4 stream cipher,” *J. Cryptol.*, vol. 27, no. 1, pp. 67–108, 2014.

[58] R. Bricout, S. Murphy, K. G. Paterson, and T. van der Merwe, “Analysing and exploiting the Mantin biases in RC4,” *Des. Codes, Cryptogr.*, vol. 86, no. 4, pp. 743–770, 2018.

[59] A. Jana and G. Paul, “Revisiting RC4 key collision: Faster search algorithm and new 22-byte colliding key pairs,” *Cryptogr. Commun.*, vol. 10, no. 3, pp. 479–508, 2018.

[60] K. H. Brown, “Security requirements for cryptographic modules.” Fed. Inf. Process. Stand. Publ, pp. 1–53, 1994.

APPENDICES

Appendix 1: RC4 Algorithm and code

The RC4 stream cipher is a popular and widely studied encryption algorithm. It has been adopted for use in TLS and other standards. Its pseudocode is shown below [3]:

KSA(K)

```
1  for i = 0 to N −1
2      S[i] = i
3  j = 0
4  for i = 0 to N −1
5      j = j + S[i] + K[i mod K.length]
6      Swap(S[i],S[j])
7  i = j = 0
```

PRG()

```
1 i = i+1
2 j = j +S[i]
3 Swap(S[i],S[j])
4 z = S[S[i]+S[j]]
5 return z
```

Where KSA(K) is the Key Setup Algorithm and PRG() is the pseudorandom generator.

The KSA works as follows. A variable length key, *K*, is used to initialize a state vector *S*, which contains a permutation of the numbers between 0 and 255 in its elements S[0], S[1],…,S[255].

For encryption and decryption, a byte *z* is selected from S in a systematic fashion. The values in S are permuted each time z is chosen. Encryption is then done by finding the XOR of z and the byte to be encrypted (known as the plain text).

The Matlab code used for RC4 is given below:

```
function [C KS perc] = RC4(K,P,b)
```

```
%Performs RC4 encryption

% Carries out RC4 encryption on P using key K.
% The number of bits used is 2^b. Standard value = 2^8 =
256.
if(nargin == 2) %Assume typical RC4 (i.e. 256 bits)
    b = 8;
end
% Initialization of S and T (or KSA)
s_size = 2^b;
S = 0:s_size-1; % Initialize S 0,1,2,...,N-1
T = S;  %The temporary vector

%Truncate the key if it is longer than T
KEYLEN = length(K);
if(KEYLEN>length(T))
    T = K(1:s_size);
else % If not, repeat the key to fill T
    for i = 0:s_size-1
        T(i+1) = K(mod(i,KEYLEN)+1);
    end
end
% Initial permutation of S
j=0;
for i = 0:s_size-1
    j = mod( (j + S(i+1) + T(i+1)),s_size);
    %Swap S(i+1) with S(j+1)
    temp=S(i+1);
    S(i+1)=S(j+1);
```

```
    S(j+1)=temp;
end
% Stream Generation (or KSA)
i = 0; j = 0;
counter = length(P); %Needed key stream is as long as
message.
Kholder = uint8(zeros(1,counter));%Stores the key stream.

m=1;
while m<=counter
    i = mod((i + 1), s_size);
    j = mod((j + S(i+1)), s_size);
    %Swap S(i+1) with S(j+1)
    temp=S(i+1);
    S(i+1)=S(j+1);
    S(j+1)=temp;
    t = mod((S(i+1) + S(j+1)), s_size);
    Kholder(m) = S(t+1);
    m = m+1;
end

%% XOR of keystream and plaintext
C = bitxor(P,Kholder);
```

Appendix 2: Spritz Algorithm and code

Spritz is a recent steam cipher. Its pseudocode is as show below [3] :

```
INITIALIZESTATE(N)
1 i = j = k = z = a = 0
2 w = 1
3 for v = 0 to N −1
4       S[v] = v
ABSORB(I)
1 for v = 0 to I.length −1
2       AbsorbByte(I[v])

ABSORBBYTE(b)
1 AbsorbNibble(low(b))
2 AbsorbNibble(high(b))

ABSORBNIBBLE(x)
1 if a = ⌊N/2⌋
2 Shuffle()
3 Swap(S[a],S[⌊N/2⌋+x])
4 a = a+1

ABSORBSTOP()
1 if a = ⌊N/2⌋
2       Shuffle()
3 a = a+1

SHUFFLE()
1 Whip(2N)
2 Crush()
3 Whip(2N)
4 Crush()
5 Whip(2N)
6 a = 0

WHIP(r)
1 for v = 0 to r −1
2       Update()
3 do w = w +1
4 until gcd(w,N) = 1

CRUSH ()
1 for v = 0 to ⌊N/2⌋ −1
2       if S[v] > S[N −1−v]
3               Swap(S[v], S[N −1−v])
SQUEEZE(r)
1 if a > 0
2       Shuffle()
3 P = Array.New(r)
4 for v = 0 to r −1
5       P[v] = Drip()
6 return P

DRIP()
1 if a > 0
2       Shuffle()
3 Update()
4 return Output()

UPDATE()
1 i = i+w
2 j = k +S[j +S[i]]
3 k = i+k +S[j]
4 Swap(S[i],S[j])

OUTPUT()
1 z = S[j +S[i+S[z +k]]]
2 return z
```

Nomenclature:

All values in Spritz are modulo-N (in $Z_N = \{0, 1, \ldots, N-1\}$). The default value of N is 256, so Spritz is byte-oriented.

The symbol "+" always means addition modulo N, and "−" always means subtraction modulo N.

The symbols $\lceil x \rceil$ and $\lfloor x \rfloor$ denote the ceiling and floor functions (the least integer not less than x, and the greatest integer not more than x, respectively).

Only two relevant functions have not been defined in the pseudocode above, and they are: *low(b) = b mod D; high(b) =* $\lfloor b/D \rfloor$*;*

These are used to obtain the high nibble (first 4 bits) and the low nibble (last 4 bits) from a given byte, b. If a byte b can be represented as a pair (x, y) of nibbles, then we can write $b = D \cdot x + y$.

Where x is the high-order nibble and y is the low-order nibble. D is based on the size of N and is given by $D = \lceil \sqrt{N} \rceil$

For the default N = 256 we have D = 16; each eight-bit byte contains two four-bit (hexadecimal) nibbles, which can be easily computed using shifting and masking.

Encryption and Decryption

Spritz works as a sponge function and so the pseudocode for encryption and decryption are as shown below:

```
ENCRYPT(K,M)
1 KeySetup(K)
2 C = M +Squeeze(M.length)
```

3 return C

DECRYPT(K,C)
1 KeySetup(K)
2 M = C −Squeeze(M.length)
3 return M

KEYSETUP(K)
1 InitializeState()
2 Absorb(K)

The Matlab implementation is as follows:

```
function C = spritz(K,M,mode)
%Performs Spritz encryption & decryption

global i j k z a w S
N = 256; %Default value
K = double(K); M = double(M);
if(nargin==2) %Default mode is encryption
    mode = 'encrypt';
end
if(~strcmpi(mode,'encrypt') && ~strcmpi(mode,'decrypt'))
    error('Choose correct mode: "encrypt" or "decrypt"')
end
if(strcmpi(mode,'encrypt')) % Return encrypted value
    C = encryption(K,M);
end
if(strcmpi(mode,'decrypt')) % Return decrypted value
    C = decrypt(K,M);
end
    function C = encryption(K,M)
        keySetup(K);
```

```
    C = mod(M + squeeze(length(M)),N);
end
function M = decrypt(K,C)
    keySetup(K);
    M = mod(C - squeeze(length(C)),N);
end
function keySetup(K)
    initializeState(N)
    absorb(K)
end
function initializeState(N)
    global i j k z a w S
    i=0;j=0;k=0;z=0;a=0;
    w=1;
    S = zeros(1,N);
    for v=0:N-1
        S(v+1) = v;
    end
end
function absorb(I)
    for v = 1: length(I)
        absorbByte(I(v));
    end
end
function absorbByte(b)
    if (N == 256)
        D = 16;
    else
        disp(['N = ',num2str(N)])
```

```
        D = ceil(sqrt(N));
    end
    absorbNibble(mod(b,D));
    absorbNibble(floor(b/D));
end
function absorbNibble(x)
    global a S
    if (a == floor(N/2))
        shuffle();
    end
    temp = S(a+1);
    Nhalf = floor(N/2); %dummy variable
    S(a+1) = S(mod(Nhalf + x,N)+1); %addition is mod n
    S(mod(Nhalf + x,N)+1) = temp;
    a = mod(a+1,N);
end
function shuffle()
    global a
    whip(2*N);
    crush();
    whip(2*N);
    crush();
    whip(2*N);
    a=0;
end
 function whip(r)
    global w
    for v = 0:r-1
        update();
```

```
    end
    w = w +2; % Since N is a power of 2
    end
function crush()
    global S
    for v = 0: floor(N/2)-1
        if (S(v+1)>S(N-1-v+1))
            temp = S(v+1);
            S(v+1) = S(N-1-v+1);
            S(N-1-v+1) = temp;
        end
    end
end
function P = squeeze(r)
    global a
    if(a>0)
        shuffle();
    end
    P = zeros(1,r);
    for v = 0:r-1
        P(v+1) = drip();
    end
    return
end
function x = drip()
    global a
    if(a>0)
        shuffle();
    end
```

```
        update();
        x = output(z);
        return
    end
    function update()
        global i j k w S
        i = mod(i + w, N);
        j = mod(k + S(mod(j+S(mod(i,N)+1),N)+1),N);
        k = mod(i + k + S(mod(j,N)+1),N);
        temp = S(i+1);
        S(i+1) = S(j+1);
        S(j+1) = temp;
    end
    function z = output(z)
          global i j k z S
        z = S(mod(j +S(mod(i+S(mod(z +k,N)+1),N)+1),N)+1);
    end
end
```

Appendix 3: Code for extracting data points from an Image graph

Below is the Matlab code used to extract the data and to plot it:

```
I = imread(filepath,'bmp'); %filepath points to the image
level = 0.9; %Chosen through trial and error
BW = im2bw(I,level);
imshow(BW)

%From the image, the corners are picked from the image
X0 = 41;    % This was the location of the origin
Y0 = 529;   % Matlab counts vertical pixels downwards
Xmax = 848; % [X0 -> Xmax] represents 24 hours
Ymax = 20;  % Ymax-Y0: represents 8000W
% NB: Graph lies between X0-Xmax and Ymax-Y0

yGen = zeros(1,(Xmax-X0+1)); % Stores vertical values
black = false; % True if black value is encountered
white = false; % True if white value is found again

for x = 1:Xmax-X0+1
    black = false;
    white = false;
    found = false;
    for y = Y0:-1:Ymax
        if(BW(y,x+X0-1)==0)
            black = true;
        end
        if(BW(y,x+X0-1)==1)
```

```
            white = true;
            if black == true %i.e. if it is white again
                yGen(x) = Y0 - y - 1; %Height in pixels
                found = true;
                break
            end
        end
    end
end
% Scale yGen along the y-axis
% This will convert the values from pixels to Watts.
yGen = yGen*8000/(Y0-Ymax); % Original image has 8kW on y-
axis

figure,plot(linspace(0,24,numel(yGen)),yGen)
```

Appendix 4: Miscellaneous Code

```
%% Generation of keys from a given meter number:
keyBuffer = hash("sha512",meterNumber);
keylength = 128;
x = keyBuffer(1:keylength/4);
%Divide by 4 because hex digit has 4 bits
y = zeros(1,numel(x)/2,'uint8');
for m = 1:numel(y)
    y(m) = hex2dec(x(2*m-1:2*m));
end
key1 = pairHex2dec (y);

%% Code for computing the Hamming Distance
function N = getHamming(A,B)
  %Computes the Hamming Distance (i.e. number of different
bits)
  D = bitxor(A,B);
  D_bin = dec2bin(D);
  N = numel(find(D_bin=='1'))
  end

%% Code for computing the Monobit test
function y = monobitTest(Xtest)
     Xbinary = get20000bits(Xtest(1:20000/8));
     y =  numel(Xbinary(find(Xbinary=="1")))
     if(y>9725 && y <10275)
       disp("Monobit Test passed")
     else
```

```
        disp("Monobit Test failed")
    end
end

%% Code for computing the Poker Test
function y = pokerTest(Xtest)
    Xbinary = get20000bits(Xtest(1:20000/8));
    X4val = reshape(Xbinary,20000/4,4);
    X4dec = zeros(1,numel(X4val)/4);
    for k = 1:numel(X4dec)
      X4dec(k) = bin2dec(X4val(k,:));
    end
    XCount = zeros(1,16);
    for k = 1:numel(X4dec)
      %Count occurence of each number
      %NB: No of 0s are stored in XCount(1)
      XCount(X4dec(k)+1) = XCount(X4dec(k)+1)+1;
    end
    Xsum = 0;
    for k = 1:numel(XCount)
      Xsum = Xsum + (XCount(k).^2);
    end
    y = (16/5000)*Xsum - 5000;
    if(y>2.15 && y <46.17)
      disp("Poker Test passed")
    else
      disp("Poker Test failed")
    end
end
```

```
%% Code for computing the Runs and Long run Test
function y  = runTest(Xtest)
     Xbin = get20000bits(Xtest);
     runCount = zeros(2,26); %Detect long run for >=26
     k = 1;
     while (k<numel(Xbin))
       %Keep moving as long as numbers are the same
       for m = 1:26; %Number of runs
             if((k+m)>numel(Xbin) || m == 26)
           disp(["Long Run found! k = ", num2str(k),
               "m = ", num2str(m)])
           break;
         endif
         if (Xbin(k)!= Xbin(k+m))
           break
         elseif(Xbin(k)== Xbin(k+m))
           continue;
         endif
       endfor
       %Increment for correct bits (0 or 1)
       if(Xbin(k)=='0')
         %Increment value for 0
         runCount(1,m) = runCount(1,m)+1;
       else
         %Increment value for 1
         runCount(2,m) = runCount(1,m)+1;
       endif
       k = k + m; % Jump m steps. If no match, m == 1
     endwhile
```

```
    y = runCount(:,1:6);
    %Combine values for 6+
    y(1,6) = sum(runCount(1,6:end));
    y(2,6) = sum(runCount(2,6:end));
end
%% Code snippet for Brute force of RC4 and Spritz with
% small keys
for k=1:8
  rCipher(:,k) = RC4(testKey(k),Dbytes);
  sCipher(:,k) = spritz(testKey(k),Dbytes);
end

% Brute force RC4 (key 1-8 bits) and record times
timeRC4 = zeros(1,8);
for n = 1:8
  t = tic;
  for m = 0:2^n-1
    mRC4guess = RC4(m,rCipher(:,n)'); %Decrypt for all m
    if(isValid(mRC4guess))
      disp(["Key found!! Value = ", num2str(m)])
      break
    endif
  endfor
  timeRC4(n) = toc(t);
endfor

% Brute force Spritz (key 1-8 bits) and record times
timeSpritz = zeros(1,8);
```

```
for n = 1:8
  t = tic;
  for m = 0:2^n-1
    mSpritzGuess = spritz(m,rCipher(:,n)','decrypt');
     %Decrypt for all values of m
    if(isValid(mRC4guess))
      disp(["Key found!! Value = ", num2str(m)])
      break
    endif
  endfor
  timeSpritz(n) = toc(t);
endfor
% Plot to compare brute force timing
figure, plot(1:8,timeRC4,'k-.o',1:8,timeSpritz,'k:x')
legend ('RC4','Spritz')
xlabel('Key length in bits','FontWeight','bold')
ylabel('Time spent in brute force (Seconds)',
'FontWeight','bold')
```

Appendix 5: Conferences and Publications from this work

Conference proceedings

1. Lincoln Kamau, Philip Kibet, Christopher M Muriithi, "*Opportunities and challenges for smart grid communications*," 2013, Proceedings of the Sustainable Research and Innovation (SRI) Conference
2. Lincoln Kamau, Philip Kibet, Christopher M Muriithi, "*The use of RC4 Encryption for Smart Meters*," 2014, Proceedings of the Sustainable Research and Innovation (SRI) Conference, p. 58-62
3. Lincoln Kamau, Philip Kibet, Christopher Maina, Robert Macharia, "*A method for Extracting Data Points from an Image of a Plotted Graph*," 2018, VizAfrica 2018 Visualization Symposium (in press)
4. Lincoln Kamau, Philip Kibet, Christopher M Muriithi, "*Leakage of customer privacy in electrical smart meters and solutions to the problem*," 2018, The 13th JKUAT Scientific and Technology Conference (in press)

Journal Publications

1. KAMAU, Lincoln; LANGAT, Kibet; MURIITHI, Christopher. "*The use of RC4 Encryption to Provide Privacy for Smart Meters*." Journal of Sustainable Research in Engineering, [S.l.], v. 4, n. 2, p. 69-75, June 2018. ISSN 2409-1243. Available at: http://sri.jkuat.ac.ke/ojs/index.php/sri/article/view/664
2. Lincoln Kamau Kiarie, Philip Kibet Langat, and Christopher Maina Muriithi, "*Application of Spritz Encryption in Smart Meters to Protect Consumer Data,*" Journal of Computer Networks and Communications, vol. 2019, Article ID 5910528, 10 pages, 2019.
   Available at: https://doi.org/10.1155/2019/5910528