UNIVERSITY OF SOUTHAMPTON

Faculty of Physical Sciences and Engineering

Electronics and Computer Science

Cyber-Physical Systems

An Information Security Model for an Internet of Things-Enabled Smart Grid in the Saudi Energy Sector

by

Abeer Siraj Akkad

Supervisors: Dr Abdolbaghi Rezazadeh, Dr Gary Wills, and Dr Son Hoang

Thesis for the degree of Doctor of Philosophy

PhD in Computer Science

April 2024

UNIVERSITY OF SOUTHAMPTON

Faculty of Physical Sciences and Engineering

Electronics and Computer Science

An Information Security Model for an Internet of Things-Enabled Smart Grid in the Saudi Energy Sector

ABSTRACT

The evolution of an Internet of Things-enabled Smart grid affords better automation, communication, monitoring, and control of electricity consumption. It is now essential to supply and transmit the data required, to achieve better sensing, more accurate control, wider information communication and sharing, and more rational decision-making. However, the rapid growth in connected entities, accompanied by an increased demand for electricity, has resulted in several challenges to be addressed. One of these is protecting energy information exchange proactively before an incident occurs. It is argued that Smart Grid systems were designed without any regard for security, which is considered a serious omission, especially for data security, energy information exchange, and the privacy of both consumers and utility companies.

This research is motivated by the gap identified in the requirements and controls for maintaining cybersecurity in the bi-directional data flow within the IoT-enabled Smart Grid. Through literature and industry standards, the initial stages of the research explore and identify the challenges and security requirements. Threat modelling analysis identified nine internet-based threats, proposing an initial information security model. This initial model is validated using expert reviews, resulting in a reference model that includes seven security requirements and 45 relevant security controls.

To demonstrate the usefulness of this reference model as a foundation for further research, a segment of the reference model is elaborated using Event-B formal modelling. This approach assists in incorporating additional details during refinements and confirming the consistency of those details. The formal modelling process begins by formulating the functional requirements in a consistent model and then augmenting it with security controls. The effectiveness of these security controls is validated and verified using formal modelling tools.

The contribution of this research, therefore, is the unique approach to developing a framework for an IoT-enabled Smart Grid (SG) by utilising threat analysis and expert reviews in combination with formal methods. As the field of security continues to evolve, this generic framework and formal template can be reused as a foundation for further analysis of other components or access points, and to implement new security controls. The resulting model enables field experts, security practitioners, and engineers to verify any changes made, ensuring they do not compromise the security of information flow within the IoT-enabled Smart Grid during the initial design stages of the system life cycle.

Keywords: IoT-enabled Smart Grid, Cybersecurity, Internet of Things, System Information Security, Threats modelling, STRIDE.

Table of Contents

List of Tables

List of Figures

Declaration of Authorship

I, Abeer Siraj Akkad, declare that this thesis entitled as follows, and the work presented in the thesis, are both my own and have been generated by me as the result of my own original research:

An Information Security Model for an Internet of Things-enabled Smart Grid in the Saudi Energy Sector

I confirm that:

1. This work was done wholly or mainly while in candidature for a research degree at this University;
2. where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
3. Where I have consulted the published work of others, this is always clearly attributed;
4. Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
5. I have acknowledged all main sources of help;
6. Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
7. Parts of this work have been published as:

- Akkad, A., Wills, G. and Rezazadeh, A. (2022) 'An information security model for an IoT-enabled Smart Grid in the Saudi energy sector', *Computers and Electrical Engineering*, 105(November 2022), pp. 157–165. doi:10.5220/0011042200003194.
- Akkad, A., Wills, G. and Rezazadeh, A. (2022) 'An information security model for an IoT-enabled Smart Grid', in *Proceedings of the 7th International Conference on Internet of Things, Big Data and Securit- IoTBDS*. ISBN 978-989-758-564-7; ISSN 2184-4976. SCITEPRESS - Science and Technology Publications, pp. 157–165. doi:10.5220/0011042200003194. (April 2022)
- Akkad, A., Wills, G. and Rezazadeh, A. (2022) 'An information security model for an IoT-enabled Smart Grid in the Saudi energy sector', in *Proceedings of the 10th* Saudi Arabia Smart Grid Conference 2022. IEEE Xplore digital library. DOI: 10.1109/SASG57022.2022.10200572 (December 2022)
- Akkad, A., Wills, G. and Rezazadeh, A. (2022) 'An IoT-enabled Smart Grid: Definitions, characteristics, challenges, and future directions', in *Proceedings of the 3rd Summer School on Cyber-Physical Systems and Internet-of Things,* vol. III. Lech Jozwiak, Radovan Stojanovic, N.V. (ed.) Montenegro, Budva: MECOnet, 2022 and MANT, pp. 1171–1179. doi:10.5281/zenodo.6698644. Available at: https://www.researchgate.net/publication/361486185_Proceedings_of_the_3rd_Summer_School_on_Cyber-Physical_Systems_and_Internet-of-Things_vol_III_2022. (June 2022)

Signed:...

Date:..

Published Work

Lists of peer-reviewed publications in support of this research:

1. A journal paper in Elsevier Computers and Electrical Engineering Journal:

   Akkad, A., Wills, G. and Rezazadeh, A. (2022) 'An information security model for an IoT-enabled Smart Grid in the Saudi energy sector', *Computers and Electrical Engineering*, 105(November 2022), pp. 157–165. doi:10.5220/0011042200003194.

2. A conference paper in Proceedings of the 7th International Conference on Internet of Things, Big Data and Security – IoTBDS:

   Akkad, A., Wills, G. and Rezazadeh, A. (2022) 'An information security model for an IoT-enabled Smart Grid', in *Proceedings of the 7th International Conference on Internet of Things, Big Data and Securit- IoTBDS*. ISBN 978-989-758-564-7; ISSN 2184-4976. SCITEPRESS - Science and Technology Publications, pp. 157–165. doi:10.5220/0011042200003194. (April 2022)

3. A conference paper, in Proceedings of the 10th Saudi Arabia Smart Grid Conference 2022 on IEEE Xplore digital library:

   Akkad, A., Wills, G. and Rezazadeh, A. (2022) 'An information security model for an IoT-enabled Smart Grid in the Saudi energy sector', in *Proceedings of the 10th* Saudi Arabia Smart Grid Conference 2022. IEEE Xplore digital library. DOI: 10.1109/SASG57022.2022.10200572 (December 2022)

4. A conference paper in Proceedings of the 3rd Summer School on Cyber-Physical Systems and Internet-of-Things, Vol. 3. with two associated conferences: 11th Mediterranean Conference on Embedded Computing (MECO2022), and 10th International Conference on CPS and IoT (CPS&IoT2022):

   Akkad, A., Wills, G. and Rezazadeh, A. (2022) 'An IoT-enabled Smart Grid: Definitions, characteristics, challenges, and future directions', in *Proceedings of the 3rd Summer School on Cyber-Physical Systems and Internet-of Things, Vol. III.* Lech Jozwiak, Radovan Stojanovic, N.V. (ed.) Montenegro, Budva: MECOnet, 2022 and MANT, pp. 1171–1179. doi:10.5281/zenodo.6698644. Available at: https://www.researchgate.net/publication/361486185_Proceedings_of_the_3rd_Summer_School_on_Cyber-Physical_Systems_and_Internet-of-Things_vol_III_2022. (June 2022).

Acknowledgements

First and foremost, I am grateful to Allah Almighty, the most Gracious, the most Merciful, for granting me the wisdom, strength, energy, and good health to learn, work, and complete this thesis.

I am sincerely grateful to the many individuals and institutions who provided me with their guidance, support, and care. I acknowledge my great gratitude to my country, the Kingdom of Saudi Arabia, which has given me the opportunity to complete my PhD in a country that I had always dreamed of visiting. Thanks are also extended to the University of Southampton for its facilities and services.

I owe a debt of gratitude to my supervisory team, Dr Gary Wills and Dr Abdolbaghi Rezazadeh, for their continuous support and imparting of their knowledge and expertise. They have been integral to all stages of this project, and I thank them for their time, constructive criticism, continuous encouragement, and insight, in providing suggestions and feedback in all stages of this journey.

I would thank Dr Son Hoang, who recently joined the supervisory team on the retirement of Dr Gary Wills in 2022. I also would like to convey my appreciation to Dr Asieh Salehi, who recently joined the supervisory team, for her kind support in modelling during the challenging time of my PhD journey. I wish also to thank both Dr Leonardo Aniello and Dr Mike Wald for their comments and interventions in the confirmation viva which helped me to improve this study.

My thanks are also extended to my lab colleagues, Mr Mehmet Yagmahan and Mr Fahad Alotaibi for, for their help and support during the formal modelling stage of the PhD. My thanks and appreciation are also extended to those academic and field experts who reviewed the research instruments. Many thanks must be given to all those volunteers who participated in the pilot study for the time that they devoted to making this research possible.

I should like to express my deepest thanks and sincere gratitude to my lovely family, who were behind every step of the way, especially my husband, Eng. Mahmoud Haneif. Indeed, words of thanks are not enough to express what I would like to say. He gave me the power to overcome my studying difficulties. He gave me enough support to remove my tension and laugh. He gave up so much of his time remotely and physically to take care of our children while I worked on this thesis. I acknowledge that this research project would not have been possible without him being there beside me in happy and difficult times. To my lovely daughters Remas, Rowen, and yet-to-be-born Helen Haneif. I know I defaulted on some important aspects with them, but they should know that I love them all more than they can imagine. Thanks for being patient, for understanding when I amended family priorities, and for being in my life.

I express my endless and sincere appreciation to my parents, Teacher Howyda and my father Captain Siraj, who both provide love, support, belief, and prayers. Their prayers and supplications are always the key to open doors to all my achievements. My sincere thanks go to my sisters, MS Ghadeer, Mrs Areej, and Ms Lujain, and my brothers Eng Amr and Eng Badr, my father-in-law, my husband's family, and all my relatives who have given me their prayers and encouragement.

I would like to express my thanks to all of my PhD friends, whether at Southampton ECS School, CPS research group, other departments, or other UK universities. I appreciate their assistance with information exchange and providing honest advice: Mr Ahmed Zahrani, Mr Samuel Isuwa, Mr Ahmed Alenezi, Mrs Ameerah, Ms Hind, Mrs Ohoud, Mrs Shefaa, Ms Sara, and Mrs Asmaa. I will never forget the people who have stood by my side and given me love and encouragement; please forgive me for my errors and omissions.

Finally, my humble thanks go to myself, for believing in my abilities! I endured much pain and learned, in order to attain my goals.

Abbreviations

AMI	Advanced Metering Infrastructure
CEN/CENELEC/ETSI	EU Joint Working Group on Standards for Smart Grids
CIA	Security principles Confidentiality, Integrity, and Availability
CISA	US Cybersecurity & Infrastructure Security Agency
CPS	Cyber-Physical System
DCS	Distributed Control System
DLMS	Device Language Message Specification
DMZ	Demilitarised Zone
ECRA	Saudi Electricity and Co-generation Regulatory Authority
HTTPS	Hypertext Transfer Protocol Secure
ICT	Information Communication Technology
IETF	The Internet Engineering Task Force is the US open internet standards body that develops the technical standards of the Internet Protocol Suite (TCP/IP)
IoT	Internet of Things
IoTSF	IoT Security Foundation
KSA	Kingdom of Saudi Arabia
LDC	Load Distribution Centre
MAC	Message Authentication Code
NERC CIP	Critical Infrastructure Protocol
NIST	US National Institute of Standards and Technology

OT	Operational Technology network
PLC	Programmable Logic Controller
PTI	US Public Technology Institute
RMU	Ring Main Unit
RTU	Remote Terminal Unit
SCADA	Supervisory Control and Data Acquisition, a control system
SG	Smart Grid
SQL	Structured Query Language
SSL/TLS	Secure Sockets Layer/Transport Layer Security
TCP/IP	Transmission Control Protocol/Internet Protocol
URL	Uniform Resource Locator
US Code	The Code of Laws of the United States of America is the official compilation and codification of the general and permanent federal statutes of the United States
US DoD	US Department of Defense, provides military guidance for use by the armed forces in preparing their appropriate plans
I/O	Input/Output
DER	Distributed Energy Resources
SEC`	Saudi Electricity Company
HMI	Human Machine Interface

Definitions

Data diode A hardware device, often called a “unidirectional security gateway”. It is placed between two networks with different levels of security to control the flow of information. A **data diode** is a cybersecurity solution that makes sure that information can travel in only one direction.

ICT systems Including SCADA, ICT elements control, and managing the physical entities.

IoT-enabled smart grid Part of the IoT framework, according to Bekara (2014).

Physical assets The physical entities of the Smart Grid (transformers, circuit breakers, smart meters, cables, etc.).

Smart Grid According to IEEE, the Smart Grid (SG) has come to describe a next-generation electrical power system typified by increased use of communications and information technology in the generation, delivery, and consumption of electrical energy.

STRIDE A well-known technique in performing architectural threat analysis and identifying the threats to a system.

Net Metering An electricity billing mechanism that allows consumers who generate their electricity to use that electricity at any time.

Chapter 1 **Introduction**

The Smart Grid (SG) can be regarded as an extensive Cyber-Physical System (CPS) (Dagle, 2012). It is considered to be a critical infrastructure in all communities worldwide. Globally, the energy market is believed to be the most important asset in allowing a country to expand its economy (Bedi *et al.*, 2018). Moreover, as cities want to assure sustainable green energy as a step toward their transformation into smart cities, implementing an IoT-enabled SG is considered the best way to achieve this goal. Thus, the SG is one of the largest applications of IoT (Reka and Dragicevic, 2018; Al-Turjman and Abujubbeh, 2019). The McKinsey Global Institute predicted that the IoT will have the significant economic contribution of $3.9 to $11.1 trillion per year by 2025 (Manyika *et al.*, 2015). This influence will be felt in many areas and applications, including homes, factories, retail environments, offices, worksites, human health, outside environments, cities, and vehicles (Dalipi and Yayilgan, 2016).

The conventional power grid uses an analogue and electromechanical infrastructure in which electricity is transmitted from a centralised utility or power plant to the consumer through long-distance and high-voltage lines. The power is delivered to the neighbourhood by a distribution system consisting of transformers, distribution substations, and power lines. In this unidirectional model, there is no feedback from the consumer (Al Khuffash, 2018), so utility companies depend on meter readings by engineers to ensure that the balance of supply and demand is met in an effective manner. Meter readings provide insufficient information on the grid's condition and consumption, with no real-time energy information (Al Khuffash, 2018). Consequently, consumers must be consumption-conscious. Besides real-time challenges, there are significant issues of exponential growth and changes in demand (Mckinsey & Company, 2024), outdated grid architecture, latency, variations in load, many power outages (U.S. Department of energy, 2023), and increased carbon emissions (Al Khuffash, 2018; Mckinsey & Company, 2024). New infrastructure is needed that may overcome these challenges, and the evolution of the IoT-enabled SG could handle these drawbacks associated with the conventional grid.

The electric utility sector is currently developing an IoT-enabled SG. This is viewed as the largest-ever installation of an IoT, with thousands of smart objects such as smart meters, smart appliances, and other sensors (Reka and Dragicevic, 2018). This huge number of connected devices, besides the increasing demand for electric energy, results in significant challenges for the IoT-enabled SG. Although the SG can address the drawbacks of the traditional power system, it involves issues of security, Big Data processing, cost, centralisation, scalability, interoperability, heterogeneity, and latency.

This research discusses the present challenges to an IoT-enabled electricity Smart Grid, focusing on securing the information flow that is essential for better automation, sensing, controlling, communicating, and timely decision-making (U.S. Department of Energy, 2018). The current research proposes a comprehensive model for securing the information system of the IoT-enabled SG. The IoT-enabled Smart Grid Information Security Model is confirmed by conducting a survey among experts in this field. Then, the research demonstrates the usefulness of this initial model for further detailed analysis of the interaction between access points in the IoT-enabled Smart Grid.

1.1 Research Motivation and Research Problem Statement

The electricity grid is a highly critical infrastructure: any damage to it can bring a whole city to a standstill. For example, a power cut affects other technical infrastructure and shuts down all servers and systems in banks, healthcare, and universities. For instance, Klinger et al. (2014) conducted a comprehensive study on the effects of power outages on hospitals in which 28% of the surveyed hospitals reported critical system failures during power outages, including disruptions to electronic health records, laboratory services, and radiology departments. Another evidence in the banking sector, NASA.gov (2008) analysed the impact of power outages on financial systems in their report of system failure case studies. They found that during the 2003 Northeast blackout, there was a significant disruption to payment systems and electronic fund transfers. The cost of financial losses related to the outage was estimated at $4 to $10 billion. The electricity grid is currently encountering exponential growth in demand and variety in loads (Al Khuffash, 2018), so there is a pressing need to be smarter about managing and monitoring consumption effectively. The SG integrates the power grid with ICT (Dalipi and Yayilgan, 2016) in a huge and crucial application of an IoT framework (Bekara, 2014). SG and IoT need to be integrated (Kaur and Kalra, 2016) to bring about sustainable green energy (Reka and Dragicevic, 2018; Al-Turjman and Abujubbeh, 2019). The integration of IoT with smart grids has emerged as a key enabler for efficient energy management, real-time monitoring, and enhanced grid reliability in smart citie (Alotaibi et al., 2020a; Cavalieri et al., 2022). An IoT-enabled SG can be considered one pillar of the smart city (Kimani *et al.*, 2019), connecting billions of smart devices around the grid, through which data can be exchanged (Kumar et al., 2020).

An IoT-enabled SG consists of physical assets such as smart meters, actuators, circuit breakers, cables, transformers, and sensors, so cyber-attacks against the SG may compromise these physical assets (Bekara, 2014) and, in consequence, cause serious financial damage to the economy. This significant financial damage is supported by several real-world examples and case studies such as Ukraine power grid attack and Stuxnet attack shown in Table 2-4. Securing such critical infrastructure is considered an absolute necessity as will be shown in the security attack incidents worldwide in Table 2-4 in section 2.9.2. Although numerous studies have been conducted in the field of security of the SG, further work is still required to overcome its security challenges.

Related works are reviewed in section 2.9.3 to highlight the differences and therefore the actual contribution of this work. Despite the progress that IoT has achieved with the SG, the literature reveals that many obstacles are still to be overcome (Bekara, 2014; Risteska Stojkoska and Trivodaliev, 2017; Reka and Dragicevic, 2018; Shakerighadi *et al.*, 2018; Al-Turjman and Abujubbeh, 2019; Ghasempour, 2019; Kimani *et al.*, 2019; Mugunthan and Vijayakumar, 2019), so it is important to develop a model with the appropriate structured approach to address issues that might impede the secure flow of information across the energy sector.

For example, the Kingdom of Saudi Arabia (KSA) is currently undertaking huge investment in the SG (SASG, 2023), in light of the highly variable demand for and consumption of electricity, as reported by ECRA (Marcelo et al., 2013). This research aligns the current shift in Saudi Arabia towards smart meters to the surrounding security concerns (SASG, 2023). In line with the Saudi National Transformation Program and Saudi Vision 2030, this research supports the goals announced for Saudi Vision 2030's digital transformation and energy sustainability (KSA, 2017). This covers not only the delivery of energy from utilities to consumers but also its integration with ICT to reduce costs, save energy, and increase reliability. In addition, this research follows the NEOM project (the futuristic mega smart city in KSA). It is reported that NEOM will be at the forefront of renewable energy (NEOM, 2024).

From the above, it can be seen that this research aims to support the electricity industry, enabling it to exchange information securely among a massive number of IoT devices around the whole SG in a systematic way. The main focus of this research is the cybersecurity of an IoT-enabled SG. It highlights the intersection between three active research fields: IoT; SG; and security. A comprehensive security model for IoT-SG is not available, it is believed that no previous study has addressed all the main potential access points of an IoT-enabled Smart Grid as a whole. These are vulnerable to internet-based threats. No study has investigated all the main relevant security controls that could mitigate against such internet-based threats using a structured modelling approach, either generally or specifically for KSA's electricity sector. Moreover, it is believed that no previous security model using formal methods has been developed to formulate IoT-enabled Smart Grid components such as SCADA/stations, instead being designed usually without any security considerations. This research, by contrast, is designed around such components and considers security.

The research conjecture is that the security controls identified here are sufficient to address the security threats to IoT-enabled SG.

The main focus of this research is information cybersecurity in the IoT-enabled SG. This research will be beneficial to system designers, information security practitioners, and stakeholders, as they consider the key requirements and challenges, identify the security threats and vulnerabilities, and maintain the required measures through the initial stages of the development of the IoT-enabled SG system life cycle.

1.2 Research Question

To meet the research motivation, the following research question needs to be answered.

RQ: What is an appropriate model for a secure information system for an IoT-enabled Smart Grid in the Saudi energy sector?

This question has three sub-questions:

a. Question 1: What are the main challenges of the IoT-enabled Smart Grid?
b. Question 2: What is the security model for the information flow around the IoT-enabled SG?
c. Question 3: How to verify the security model for the IoT-enabled SG?

1.3 Aims and Objectives

The research questions have four objectives:

Research Questions	Aims	Objectives
a. Question 1: What are the main challenges of the IoT-enabled Smart Grid?	1. To investigate the main challenges of the IoT-enabled Smart Grid.	a. To survey the main challenges that the electric energy sector faces in securing the bi-directional flow of information.
b. Question 2: What is the security model for the information flow around the IoT-enabled SG?	2. To develop a security model for the information flow around the IoT-enabled SG.	b. To review the security requirements that IoT-enabled SG needs to fulfil. c. To analyse the main IoT cybersecurity threats against information security in IoT-enabled Smart Grids in order to identify the security controls that mitigate and detect potential IoT cybersecurity threats in the electric energy sector. d. To confirm the developed Information Security Model for an IoT-enabled Smart Grid, including access points, security requirements, threats, and controls.

c. Question 3: How to verify the security model for the IoT-enabled SG?	3. To validate and verify the security model for the IoT-enabled SG.	e. 1. To demonstrate the usefulness of the proposed model to conduct a more detailed and tangible analysis. 2. To verify and validate the effectiveness of the mitigation security controls to address the identified security concerns.

1.4 Research Scope

This research focuses on the cybersecurity of the information flow, using technical security controls against internet-based threats in the IoT-enabled Smart Grids, as seen in Figure 1-1. This research is not concerned with physical security, power flow, performance, and financial considerations around security controls, non-internet-based threats such as Phishing by phone or voice calls, human-based or non-technical security controls such as (staff training, policies, and regulations).

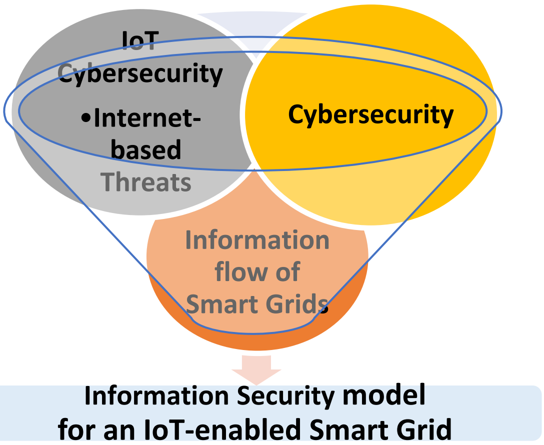

IoT
Cybersecurity
•Internet-based
Threats
Cybersecurity
Information flow of Smart Grids
Information Security model
for an IoT-enabled Smart Grid

Figure 1-1: Research scope

1.5 Thesis Structure

Chapter 2 presents a literature review that discusses the concept of the SG, its definitions, components, and characteristics, as well as the conceptual and architecture models, and defines the concepts in this field. The role of IoT in the SG is explained. Then, the challenges of implementing an IoT-enabled SG are discussed, and the IoT cybersecurity dimensions are specifically addressed, followed by a comprehensive exploration of related work. Moreover, the security concerns and measures are presented. Finally, an overview of formal methodologies and event-B is provided.

The security requirements are investigated and mapped to the potential internet-based threats in **Chapter 3**, which are then analysed using threat modelling. Security controls and measures are identified to mitigate the cyber threats discussed. Finally, the IoT-enabled Smart Grid Information Security Model is developed.

Chapter 4 outlines the methodology used to approach this research. In **Chapter 5**, the IoT-enabled Smart Grid Information Security Model is reviewed by experts, and the findings are analysed and discussed. **Chapter 6** validates and verifies the confirmed model using formal methods, formalising the functional requirements and basic aspect of availability to develop the Functional Formal Model. Then, **Chapter 7** augments the formal model by adding the security layer, concluding the conclusions of this research so far in **Chapter 8**, with an overview of future work.

Chapter 2 **Literature Review**

This chapter provides an overview of the state of the art in this field. It considers the background and available solutions and identifies the limitations of particular areas. It highlights the intersection between three active research fields: IoT; SG; and information security. It is believed that no previous study has addressed sufficiently the challenges of IoT-enabled SG. The research objective is to develop an appropriate model for a secure information system for an IoT-enabled SG in the Saudi energy sector.

2.1 **SG Definition**

The many definitions of SG vary between organisations and studies, as shown in Table 2-1, and there is no agreement; however, the common concept is that SG revolves around an information communication infrastructure. For instance, in the definition by the largest standardisation authority, IEEE, the SG describes a new age of electricity that features the use of ICT in the generation, delivery, and consumption of electricity and the electric system (IEEE, 2018). Likewise, the viewpoint of the Ontario Independent Electricity System Operator (IESO), the leader in SG, is that it involves the use of ICT in optimising all power system operations for the benefit of the consumer and the environment (Singer, 2009).

Both definitions focus on the SG component, which is specifically the communication infrastructure, whereas others focus on the outcomes that benefit from SG. For instance, the Energy Independence and Security Act of 2007 (EISA, 2007) produced the first official definition of SG (US Public Law, 2007; Al Khuffash, 2018), as given in a report to the US Congress: "*The modernization of the Nation's electricity transmission and distribution system to maintain a reliable and secure electricity infrastructure that can meet future demand growth and to achieve a set of requirements that together characterize a SG*" (US Public Law, 2007; U.S. Department of Energy, 2018). This definition describes SG from the perspective of its benefits. Within the IEEE and EISA definitions, SG domains are prominent, including electricity generation, transmission, distribution, and consumption (US Public Law, 2007; IEEE, 2018).

In the context of information technologies, other definitions focused on how information could be transferred through the SG. The bi-directional flow has given rise to the term "prosumers" in SG (Dalipi and Yayilgan, 2016), meaning customers who generate energy for the grid, as stressed by the European Union's viewpoint as well as the UK Institution of Engineering and Technology (IET, 2013), also shown in Table 2-1. The IET's definition of SG is based on that of the ETP (IET, 2013). From an environmental perspective, both Singer (2009) and the Electric Power Research Institute (2005) mention green energy and the environmental impact of SG in their definitions as the most important advantages of SG due to their contribution to a reduction in the CO_2 footprint (EPRI, 2005; Singer, 2009).

From the above, the SG can be defined as the integration of ICT into the existing electrical network, consisting of renewable sources and involving its multiple domains (generation, transmission, distribution, and consumption) in the efficient automation and real-time demand management of a reliable, sustainable, bi-directional, and economic green electrical energy.

Table 2-1: Summary of Smart Grid definitions

Organisation	**Definition**
IEEE	Smart Grid describes a new age of electricity that features the use of Communications and Information Technology (CIT) in the generation, delivery, and consumption of the electrical system. (IEEE, 2018)
DOE/EISA (US Dept of Energy)	The modernization of the Nation's electricity transmission and distribution system to maintain a reliable and secure electricity infrastructure that can meet future demand growth and to achieve a set of requirements that together characterize a Smart Grid. (U.S. Department of Energy, 2018)
IESO (Independent Electricity System Operator)	Smart Grid is the employment of ICT in optimizing all power system operations for the benefit of the consumer and the environment. (Singer, 2009)
ETP (European Union)	Smart Grid is developed by the European Technology Platform, and it means the smart integration of all operations from the connected producer, consumers, and prosumers to supply sustainable, and secure power energy. (ETP, 2006)
EPRI (Electric Power Research Institute)	A Smart Grid is one that incorporates information and communications technology into every aspect of electricity generation, delivery, and consumption in order to minimize environmental impact, enhance markets, improve reliability and service, and reduce costs and improve efficiency. (EPRI, 2005)

There are two flows in the Smart Grid:

Electricity flow is the classic flow in a conventional electrical grid from generating stations to consumers, while in SG this flow is bi-directional (Bekara, 2014).

Information flow is a bi-directional flow between utilities and all components of the SG, including smart meters, sensors, actuators, smart appliances, and electric vehicles. Consequently, this flow is a real-time Big Data flow, owing to the increase in the number of connected devices on the SG (Bekara, 2014).

2.2 Why SG?

Decarbonisation has become a goal worldwide for all countries to address climate change and limit global warming by reducing CO_2 emissions (Colak, 2016; Reka and Dragicevic, 2018). Over time, the exponential growth of demand and the variety of loads have become a burden on the electricity grid (Al Khuffash, 2018). For instance, electric vehicles require charging, and the United Kingdom is investing significantly in supporting the growing number of electric vehicle infrastructures (GOV.UK, 2024). Consequently, there is a strong probability that the grid will be overwhelmed by increasing demand for electricity. Then, costs will rise due to operational expense and latency (Al Khuffash, 2018). Thus, Colak (2016) emphasised that the rapid growth in demand for electricity, and the variety of loads, must be managed and planned efficiently to secure cost containment or reduction.

In addition, transmission and distribution lines experience both losses and unauthorised consumption (Colak, 2016; Al Khuffash, 2018), so inevitably the need has arisen to be smarter with the electricity grid to manage and monitor consumption effectively and to ensure power availability. The electricity supply could be managed efficiently by increased standardisation of the information system between utilities and consumers in the SG (DeBlasio and Tom, 2008; Sortomme *et al.*, 2011; Colak, 2016). In an SG, more monitoring and control could regulate power generation to respond to demand. By contrast, in a conventional power grid, traditional meter readings provide insufficient information on grid conditions and consumption, with no real-time energy information (Al Khuffash, 2018). Consequently, consumers have no data on their usage, which, in turn, leads to rising costs. In addition, the centralised architecture of the conventional grid may represent a burden on its productivity. Therefore, the SG is considered to have the potential to solve the drawbacks of the old infrastructure on a conventional grid (Ghasempour, 2016).

Both Ghasempour (2019) and Al Khuffash (2018) argue that SG is essential to enable and integrate all the renewable energy sources in the system, such as solar, hydro, and wind. The SG can both handle the variability and counterbalance the constant fluctuations in wind and solar.

From the above, the reasons why cities need an SG can be summarised as: the SG ensures controllable automation; the integration of renewables; sustainable green energy solutions; and real-time awareness (IET, 2013; Colak, 2016; Kaur and Kalra, 2016). As a result, the development of the SG is looked on as the infrastructure to overcome the challenges of rising carbon emissions, rapid growth in demand, overloading, latency, transmission losses and outage, real-time information inadequacy, a centralised old architecture, and integrating the multiple forms of green energy.

2.3 What Makes the Grid Smart?

It is argued that digital technology is what makes the grid smart (U.S. Department of Energy, 2018). To achieve this, information technology systems have to be deployed to supply the data required for better sensing, precise control, wider information communication and sharing, powerful computing, and better decision-making (U.S. Department of Energy, 2018).

2.4 IoT-Enabled SG Characteristics

The following 10 points characterise the IoT-enabled SG. This step contributes to developing a proper model by assuring the functionality that needs to be accomplished in each characteristic. The characteristics issued by NIST and commonly used in the sector (NIST, 2014; U.S. Department of Energy, 2018) are as follows:

1. To increase the usage of ICT to enhance the reliability and efficiency of the power grid
2. To optimise the operations and resources of the grid, with full cybersecurity
3. To integrate distributed resources and generation, such as those of renewable resources
4. To incorporate the demand response, demand-side resources, and energy-efficiency resources
5. To deploy smart technologies such as real-time, automated, interactive technologies that improve the physical operation of appliances and consumer devices for metering, communicating, and reporting grid status
6. To integrate smart appliances and consumer devices
7. To integrate advanced electricity storage and peak-sharing technologies, including plug-in electric and hybrid electric vehicles and thermal-storage air conditioning
8. To provide timely information and control services to consumers
9. To standardise the communication and interoperability of appliances and devices connected to the power grid with the grid’s infrastructure
10. To reduce unnecessary obstacles to the adoption of SG technologies, practices, and services.

2.5 SG Architecture

The European Commission developed the Smart Grids Architecture Model (SGAM) (CEN/CENELEC/ETSI, 2012). This consists of the following five logical layers:

a. **Business**: Provides the business view on information exchanged in SGs. This layer supports business experts in economic decision-making for new businesses, as well as regulating the market models

b. **Function:** Provides functions and services. This layer outlines the use cases without actors or components

c. **Information:** Describes the information exchanged among functions, services, and components. It contains information objects and data models

d. **Communication:** Represents the protocols and mechanisms to exchange information between components

e. **Component**: Represents the physical distribution of all components in the SG, including physical assets and ICT technologies.

Based on the description above, the SGAM model has a lack of system actors and components, whereas NIST provides details of smart grid components and focuses on the interactions between different domains and actors within the smart grid. This is vital to model the system properly. In addition, SGAM is promoting interoperability and economic viability which is beyond the scope of current research. Therefore, The SGAM model will not be used in modelling in this research.

2.6 SG Conceptual Model

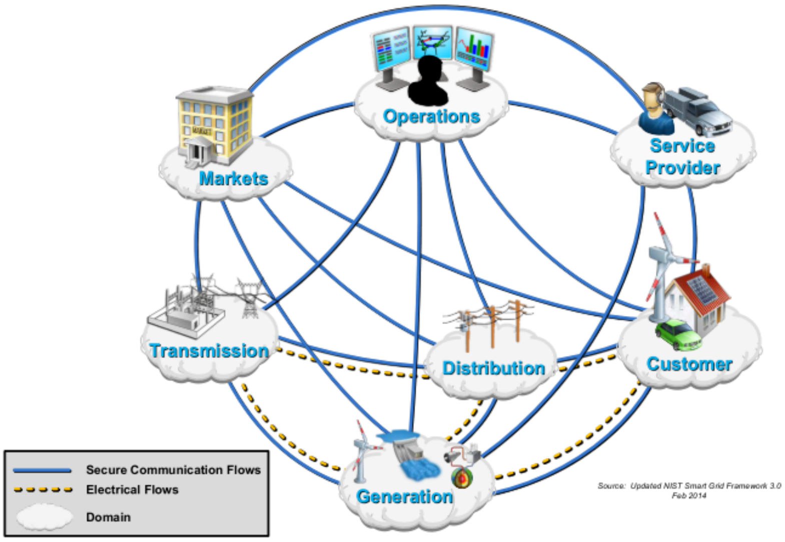


Figure 2-1: NIST conceptual reference model for SG

Various designs have been proposed for the SG architecture; however, the conceptual reference model by NIST (US National Institute of Standards and Technology) is commonly referred to in the sector. According to this model, Figure 2-1 (NIST, 2014), SG contains the seven domains and actors shown in Table 2-2.

Table 2-2: Description of domains, and the actors in each (NIST, 2014)

Domain	Description	Actor
Bulk and non-bulk generation	The process of creating and extracting bulk quantities of electricity from other sources and types of energy, such as chemical, geothermal, nuclear, and renewable energy including solar rays,	Generators

Domain	Description	Actor
	wind, and hydro flow. It may also store the energy for any distribution needed later.	
Transmission	The systems that pass and link the electricity flow from the generation systems to the distribution systems over long distances. In order to minimise electricity losses, these systems are designed to operate at extremely high voltages.	Carriers
Distribution	Systems that consist of the electrical networks supplying the electricity flow to and from the customers and transmission systems.	Distributors
Customer	A customer is the end-user of electricity, classified as one of three types: residential; commercial; and industrial. Customers have the ability to generate and store electricity.	Consumers
Markets	In the deregulated energy industry there are two markets: the energy market; and the transmission market. The energy market provides a competitive marketplace for energy and other energy products (e.g. ancillary services), whereas the transmission market is a competitive marketplace for transmission rights to carry electricity from one place to another. Power system operations involve the management of electricity flow and ensuring that the electricity is delivered in a reliable, safe, and economic manner. Power system operations can be divided into bulk transmission, distribution, and field devices.	Operators and participants in electricity markets
Service providers	The organisations that perform services to electrical customers and utilities. They boost the processes of power system generators, transmitters, distributors, and customers in business, ranging from traditional utility services (such as billing and customer account management) to enhanced customer services (such as energy consumption management and home energy generation).	Providers
Foundational support systems	The systems that execute the non-industrial processes that support energy industrial processes, for example information technology (IT), cyber and physical security, architecture solutions for IT support systems, and cost-benefit analysis.	Systems and applications

In summary, while SGAM offers a detailed, multi-dimensional approach to smart grid architecture, the NIST Conceptual Model provides a high-level, strategic framework widely recognized for its authority, comprehensive scope, and emphasis on standardization and security.

Although the NIST conceptual model is commonly referred to in the sector, it lacks details in terms of cybersecurity and information flow, especially in the IoT infrastructure. The NIST model contributes to the concept of the SG architecture only, while this research fills in the lack of the NIST conceptual model to develop a case study that is useful for the related sectors to use.

2.7 IoT and IoT Devices

This section presents the background of the Internet of things (IoT) and IoT devices. To address the research goals it discusses the main IoT modules of IoT devices. IEEE defines IoT as the integration, via the internet of things that are equipped with sensors. The ITU Telecommunication Standardisation Sector (ITU-T) considers the IoT system as an infrastructure for an information system that connects physical and virtual entities. Cisco (2013) gave a definition for IoT as 'the Internet of Everything', with the ability to gather people, data, and things to construct a network capable of exchanging information (Cisco, 2013). Hewlett-Packard states that IoT is a system in which every object is connected to the internet. Shakerighadi et al. (2018) define IoT as infrastructure, including sensors, communication systems, information systems, and objects connected to the internet, which are essentially standardised.

According to Rathke and Sassone (2010), an IoT device consists of five main modules, as shown in Figure 2-2: (1) a sensing module; (2) a processing module; (3) an actuation module; (4) a communication module; and (5) an energy module. These are supported by storage and applications.

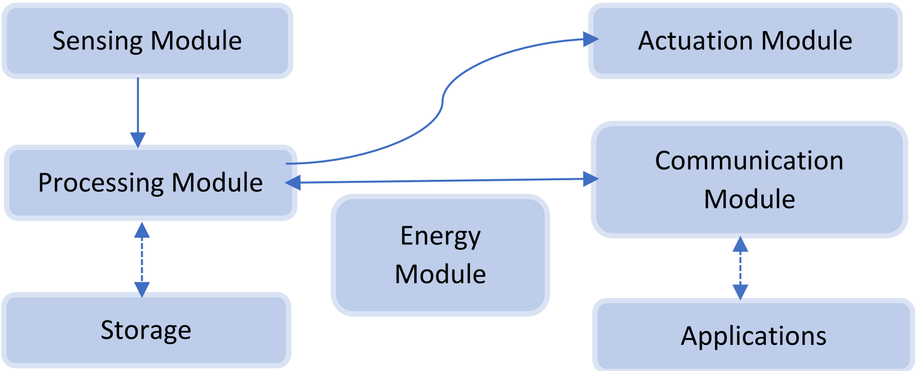


Figure 2-2: Main modules of an IoT device (Rathke and Sassone, 2010)

2.8 IoT and SG

In this section the role of IoT in the SG is explained to establish whether the IoT is advantageous to SG or not. The SG consists of thousands of connected devices, and these require tracking for monitoring purposes, connectivity, and automation (Al-Ali and Aburukba, 2015; Saleem *et al.*, 2019). The IoT is an enabling technology that brings internet connectivity to the SG (Al-Ali and Aburukba, 2015; Saleem *et al.*, 2019; Yang, 2019).

In the context of IoT, in SG every device is connected to the internet. To facilitate communicating information and receiving control commands via the internet protocols, each must have a unique IP address (Al-Ali and Aburukba, 2015; Saleem *et al.*, 2019). Under the IP addressing schemas, IoT can offer **monitoring and control capabilities** for SG, as discussed by Kaur and Kalra (2016). This monitoring aspect can cover the generation plan, distribution, storage, and consumption to achieve efficiency management, demand management, renewable energy needed measurement, and CO_2 emissions administration. Therefore, IoT devices contribute to the reduction of wasted energy and the accurate estimation of required energy.

Further, those devices exchange data in bi-directional flow via the SG communication layer, using communication protocols such as Wi-Fi, Zigbee, WiMax, LTE, and GPRS. IoT standardises communication, reducing the number of these protocols relating to the SG components (Al-Ali and Aburukba, 2015). Both Saleem et al. (2019) and Al-Ali and Aburukba (2015) emphasise that IoT technologies enable SG to **communicate** across all its multiple subsystems of generation, transmission, distribution, and consumption. Al-Ali and Aburukba (2015) state that each device can exchange data and commands from the control centres and utilities.

Both Kaur and Kalra (2016) and Al-Ali and Aburukba (2015) suggest that all objects in an SG can be represented as IoT devices distributed throughout the residential network, substations, and utilities. For instance, these devices could be:

- Smart home appliances with electric vehicle charging
- Substation devices (smart meters, actuators, circuit breakers, transformers, switches, routers, concentrators, voltage regulators, capacitors, or cameras)
- Renewable energy sources
- Utility and control data centres (servers or testing devices).

The conventional power grid relies on SCADA systems, which are built with a centralised architecture (Yang, 2019). Utilising IoT in such systems increases their scalability, efficiency, and availability; however, there are serious risks. Similar to devices, on the control centre side SCADA has its own IP address. Classical SCADA systems are renowned for having no proper security controls, in part because they were never designed to be open to the internet. With the integration of complex new architectures such as IoT, therefore, in deploying SCADA systems on the internet an issue is security (Sajid *et al.*, 2016). This is especially true since several types of malware have recently targeted SCADA systems owing to this lack of built-in security (Pour *et al.*, 2017; Kimani *et al.*, 2019). Centralisation and security issues are discussed in the Challenges section.

Mugunthan and Vijayakumar (2019) support the claim that IoT technologies afford SG with the cloud, 5G, mobile wireless networks, application programming interfaces (APIs), machine learning, AI, predictive analytics, and Big Data management.

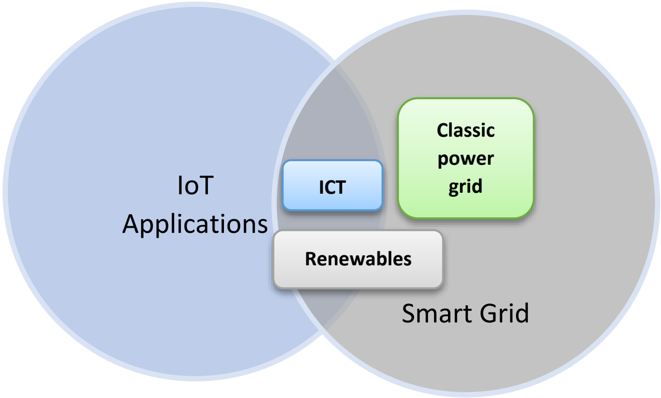


Figure 2-3: SG and IoT

In short, SG is considered to be one of the biggest applications of IoT (Bekara, 2014; Reka and Dragicevic, 2018; Al-Turjman and Abujubbeh, 2019), as shown in Figure 2-3. In other words, IoT technology is subsumed under the wider umbrella of SG. However, many studies consider IoT as a technology separate from the SG. From the cyber-physical systems point of view, this research considers IoT as part of the SG itself, enabling all those features that are discussed here.

2.9 SG and Security

SG affords many opportunities, but it also presents security challenges. To get the most out of an SG it is essential to develop a highly secure information system. As described above, it is argued that automation systems such as SCADA were designed with no regard for security (Aloul, 2012). Moreover, Modbus, which exchanges SCADA information to control industrial processes, was not intended for critical security environments such as SGs (Aloul, 2012). Thus, securing the information system in an SG must be assigned the highest priority, since power assets represent critical national infrastructure that may attract terrorists and state actors. Any damage, such as security attacks on the power grid, could cause chaos across whole cities. Electric Power Research Institute (ERPI) reports that one of the main concerns in SG implementation worldwide is security.

Security is important to prevent an attacker from modifying the consumption data. This compromises the security of smart meters and misleads estimations, and an incorrect consumption estimation could lead to large financial losses (Mahmood *et al.*, 2016). Security is also important to protect the privacy of consumers and the utility. It has been pointed out that it is possible to monitor a user’s activity through their electricity consumption patterns, revealing the times when the user is present or absent from the house (Kimani *et al.*, 2019), by collecting information from their smart meter (Mahmood *et* *al.*, 2016). In this context, information security is defined by three concepts: **availability**; **integrity**; and **confidentiality** (NIST, 2014). Table 2-3 presents the CIA's definition (NIST, 2014; US Code, 2016). These concepts are crucial to the topic of security analysis as a whole.

Table 2-3: Security concepts (CIA)

Security Concept	Definition
Availability	Ensuring timely and reliable access to and use of information (US Code, 2016)
Integrity	Guarding against improper information modification or destruction, including ensuring information non-repudiation and authenticity (US Code, 2016)
Confidentiality	Preserving authorised restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information (US Code, 2016)

Security challenges of IoT-enabled SG can arise for many reasons. First, the electricity components in the IoT-enabled SG communicate using the IP-based communication network, exchanging sensitive and private data between both consumers and utility companies. Such networks are susceptible to many types of cyber security threats, such as man-in-the-middle, denial of service, eavesdropping, and replay attacks, as shown in Chapter 3. Second, IoT-enabled SG consists of various components that communicate with one another, which requires interaction among these technologies. The interconnectivity capabilities of IoT technologies and their computational constraints expose the electricity components to cyberattacks (Stellios *et al.*, 2018). Accordingly, this communication introduces security vulnerable access points in SG (Mahmood *et al.*, 2016). Third, SG uses wireless sensor networks to connect smart meters, for example. It has been argued that wireless networks are insecure (He *et al.*, 2013). Fourth, by allowing unauthorised access to SG, the bi-directional information flow may expose SG to many threats. Fifth, using IoT in SG may cause it to inherit IoT's security issues. For monitoring and control purposes with IoT devices, SG should use the internet (Ghasempour, 2019).

There are several security concerns over IoT technologies that stem from their exposure to the internet, which is harmfully vulnerable. This exposure can allow an attacker to tamper with the data. Besides, the ever-increasing number of IoT devices used in the SG makes it more vulnerable to attack (Kimani *et al.*, 2019). Security concerns driven by IoT are discussed in greater depth in the next subsection.

2.9.1 **Cyber Kill Chain**

Cyber Kill Chain (CKC) is based on a model used by the military to describe the phases in which an attacker operates to threaten a targeted system (Spitzner, 2019). The US military lists the phases as Find, Fix, Track, Target, Engage, and Assess (F2T2EA) (US DoD Joint Publication, 2013). Lockheed Martin later extended this for use outside the military (Hutchins *et al.*, 2011). According to Lockheed Martin, a CKC is “a systematic process to target and engage an adversary to create desired effects”. It argues that traditional defences, such as antivirus, firewall, and intrusion detection systems, are insufficient to counter sophisticated threats by a professionally skilled attacker (Hutchins *et al.*, 2011). Some researchers have therefore linked CKC to an Advanced Persistent Threat (APT), one of the threats to SG. The idea behind the Kill Chain is to understand the phases that an attacker may execute, thereby constructing a better defence at each phase (SANS, 2019).

2.9.2 **Cyber Threat Actors**

Building comprehensive knowledge of the attacker is considered a significant step in avoiding and mitigating security threats. One technique to build this knowledge is the identification of Cyber Threat Actors (CTA). Many organisations define these; for example, NIST published the 800-82 Guidelines to SCADA and Industrial Control System Security (Stouffer *et al.*, 2015). Besides the US agencies defining the Threat Actors, also the European Network and Information Security Agency (ENISA) (Marinos and Lourenço, 2018), the Cybersecurity & Infrastructure Security Agency (CISA, 2005), and SANS (Irwin, 2019) have done do. Sailio et al. (2020) discussed the various views and classifications of these international organisations, defining a Threat Actor as “an entity that is responsible for an incident that impacts or has the potential to impact an organisation’s security”. They go on to describe the various classes (CISA, 2005; Sailio *et al.*, 2020).

a. **Cybercriminals:** intend to steal sensitive data, money, and personal information for financial gain. They target data-rich sectors
b. **Nation-state actors:** sponsored by nations. They target government sectors to commit espionage and obtain intellectual property, sensitive national information, and money, to advance their nation’s cause
c. **Ideologues (hacktivist and terrorist):** both have an ideological motivation to commit their attack. Hacktivists violate security to advance their cause, such as by revealing confidential matters and deactivating the services of sectors that they consider are undesirable. Terrorists aim to cause terror. The terrorist actor is used by nation-states, so it is argued that the aforementioned Stuxnet incident (Table 2-4) should be classified as a terrorist attack as well as a nation-state actor attack
d. **Thrill-seekers and script kiddies:** these hack the systems to prove themselves, learn, and perform experiments. While script kiddies are incapable of designing hacking tools, due to their low skill level, they can use tools developed by other attackers
e. **Insiders:** these work within the sector to sell system access to another actor or to cause problems for the sector on behalf of a dissatisfied employee

f. **Competitors:** economic rivals who target the National Information Infrastructure (NII), with a strong ability to undertake industrial espionage to obtain blueprints, for example, confidential recipes. They usually target companies

g. **Partners:** vendors, clients, auditors, and suppliers who abuse trust by exploiting it through a social engineering attack.

2.9.3 **Cyber Threat Vector**

According to NIST, a cyber threat attack vector is a specific path or method that can be exploited to compromise the IT system through such vulnerability (Cichonski et al., 2012). The threat vector is the scenario that an attacker follows to gain access to a device, such as exploiting a browser vulnerability for malware installation.

2.9.4 **Challenges to IoT-Enabled SG**

Before implementing the IoT-enabled SG, it is vital to research the potential challenges and risks identified at the design stage. Khan et al. (2017) stressed the need to model SG at the system design stage for appropriate and effective security control mechanisms. Although IoT-enabled SG could address the drawbacks of the traditional power system, it involves security, cost, centralisation, scalability, interoperability, heterogeneity and complexity, and Big Data processing, as in Figure 2-4.

This section discusses the potential security risks to the implementation of IoT-enabled SG, in particular to the Saudi energy sector. This research is concerned with specifically the SG and, as explained previously in sections 2.2 and 2.3, the smartness and intelligence part means enabling IoT features to the SG. Thus, SG is considered one of the biggest applications of IoT. This section presents a subset of the challenges that filter the challenges to IoT infrastructure, namely those that apply to the SG. The challenges are inherent in the use of IoT.

As a result of the growth in the number of objects connected in the SG, **Big Data processing** becomes an issue (Sagiroglu *et al.*, 2017). Advanced Metering Infrastructure (AMI) in the SG produces Big Data that needs to be handled, stored, and analysed efficiently (Shakerighadi *et al.*, 2018). Smart metering and ICT deployment lead to generating big energy data in terms of volume, velocity, and variety (Hu and Vasilakos, 2016). These data can be exploited to obtain insight, make decisions, predict future consumption patterns, and the required distribution of power supplies (Shakerighadi *et al.*, 2018; Ghasempour, 2019). With sophisticated data analytics, superior monitoring and control can be achieved by the SG. In this context, Big Data could consume huge amounts of energy and other resources when information is collected, transferred, and handled by IoT devices. The SG should thus be designed to deal with the collection of Big Data (Ghasempour, 2019).

The internet is used in SGs for monitoring and control purposes, exposing to attack the information from sensors, smart meters, and other smart devices. Any tampering with the data collected in and from smart meters may cause serious financial loss. Thus, a SG's exposure to the internet could make it vulnerable, rendering its **security** another challenge (Bekara, 2014; Arasteh *et al.*, 2016; Risteska Stojkoska and Trivodaliev, 2017; Ghasempour, 2019). Ghasempour (2019) and Mahmood et al. (2016) hold the view that the implementation of SG should consider the **constrained nature of IoT devices** in computation power and storage capabilities. In turn, proper security algorithms are required to meet the limited ability of IoT devices so that they are capable of running them (Ghasempour, 2019).

The SG is considered to be one of the biggest applications of IoT, thus **security** is the greatest challenge that it faces. It is inherent in the use of IoT devices, as there are many security concerns around IoT technologies. As a cyber-physical system, it is argued that security represents a serious challenge for IoT-enabled SGs. All studies are similarly concerned about the SG's security (Arasteh *et al.*, 2016; Risteska Stojkoska and Trivodaliev, 2017; Bedi *et al.*, 2018; Reka and Dragicevic, 2018; Shakerighadi *et al.*, 2018; Ganguly *et al.*, 2019; Kimani *et al.*, 2019). As argued by Ghasempour (2019), an attacker could extract private information about prosumers and their consumption. Data values in smart meters could be manipulated. **Trust management and social factors** between parties such as consumers, substations, and utility companies could be violated through IoT devices, so the Smart Grid's confidentiality, integrity, and reliability could be damaged. As a consequence, the CIA triad of security may be compromised. The mobile nature of IoT devices in SG, such as electric vehicles, and connection stability are major issues in SG security (Bekara, 2014; Shakerighadi *et al.*, 2018; Mugunthan and Vijayakumar, 2019). Specifically, this involves the identification of IoT devices over the internet, as this allows identity spoofing attacks to hack the SG.

Shakerighadi et al. (2018) suggest that supplying sensors with energy may pose a challenge in terms of **cost**. The situation is complicated by the bi-directional flow of data, capacity and bandwidth limitations, smart meter constraints, and the long-distance transmission of data. Costs can be assigned to smart devices, software, staff training, regulations, and managing customer acceptance of smart meters (Bekara, 2014; Shakerighadi *et al.*, 2018; Connor and Fitch-Roy, 2019). From a performance point of view, the SG collects Big Data on a real-time basis from a variety of devices, presenting a significant challenge since real-time analysis is computationally expensive and, as discussed by Shakerighadi et al. (2018) and Bekara (2014), this needs to be taken into consideration. One explanation is that electricity varies in current, voltage, and frequency; consequently, power fluctuations may cause a power overload or shortage. Similarly, the **centralised** architecture of the SG poses a challenge to performance, causing a single point of failure in a power system (Atlam and Wills, 2019). If the node processing the information is attacked and thus unavailable, the whole power system becomes unavailable; however, a decentralised architecture may support power distribution and enhance the system's bandwidth (Al Khuffash, 2018).

Increases in connected smart devices, with their constrained nature, lead to another challenge: that of **scalability,** which causes a bottleneck in SG. Multiple requests may not be processed synchronously, thus increased communication latency could occur and a noticeable delay in serving consumers could be experienced (Mahmood *et al.*, 2016). Since the SG is used to connect many cities across a country, there is a need for a scalable system (Bekara, 2014). Scalability is the ability of an SG to expand incrementally, aiming to meet the prospective future rapid growth in electricity demand and to assure clustering and load balancing techniques (Bekara, 2014; Al-Turjman and Abujubbeh, 2019). Al-Turjman and Abujubbeh (2019) consider that scalability plays an essential role in enhancing the power grid's reliability and quality since it affects the availability of this vital asset. Thus, scalability affects security. Furthermore, an SG consists of devices from various vendors, applications, services, protocols, and communication stacks, introducing challenges of **heterogeneity and complexity** (Bekara, 2014; Arasteh *et al.*, 2016; Bedi *et al.*, 2018). The SG comprises subsystems of power systems, control systems, and communication systems. Furthermore, integrating these subsystems involves issues of information management that need to be considered, since the SG is a system of systems (Shakerighadi *et al.*, 2018).

From a communication point of view, the challenge of **interoperability** relates to heterogeneity. SG exchanges information among many IoT devices and gateways of varying specifications. By combining the views of Bekara (2014), Risteska Stojkoska and Trivodaliev (2017), Shakerighadi et al. (2018), and Ghasempour (2019), it can be seen that interoperability may be attributed to the heterogeneity of protocols and communication stack, as discussed above. There are many separate standards for IoT devices and there is no unified standardisation efforts in the SG, causing interoperability issues for IoT devices (Ghasempour, 2019). For example, legacy systems cannot communicate with IP-based systems in the SG due to the lack of support for some protocols such as TCP/IP (Bekara, 2014; Risteska Stojkoska and Trivodaliev, 2017; Shakerighadi *et al.*, 2018).

From the above, it can be concluded that the security of the SG system is affected by issues of heterogeneity, complexity, interoperability, and constrained devices. Furthermore, it is argued that centralisation may affect the availability of the system. Therefore, it is significant to remember each of these challenges alongside the others in SG implementation: for example, to maintain security it is important to consider correlated aspects.

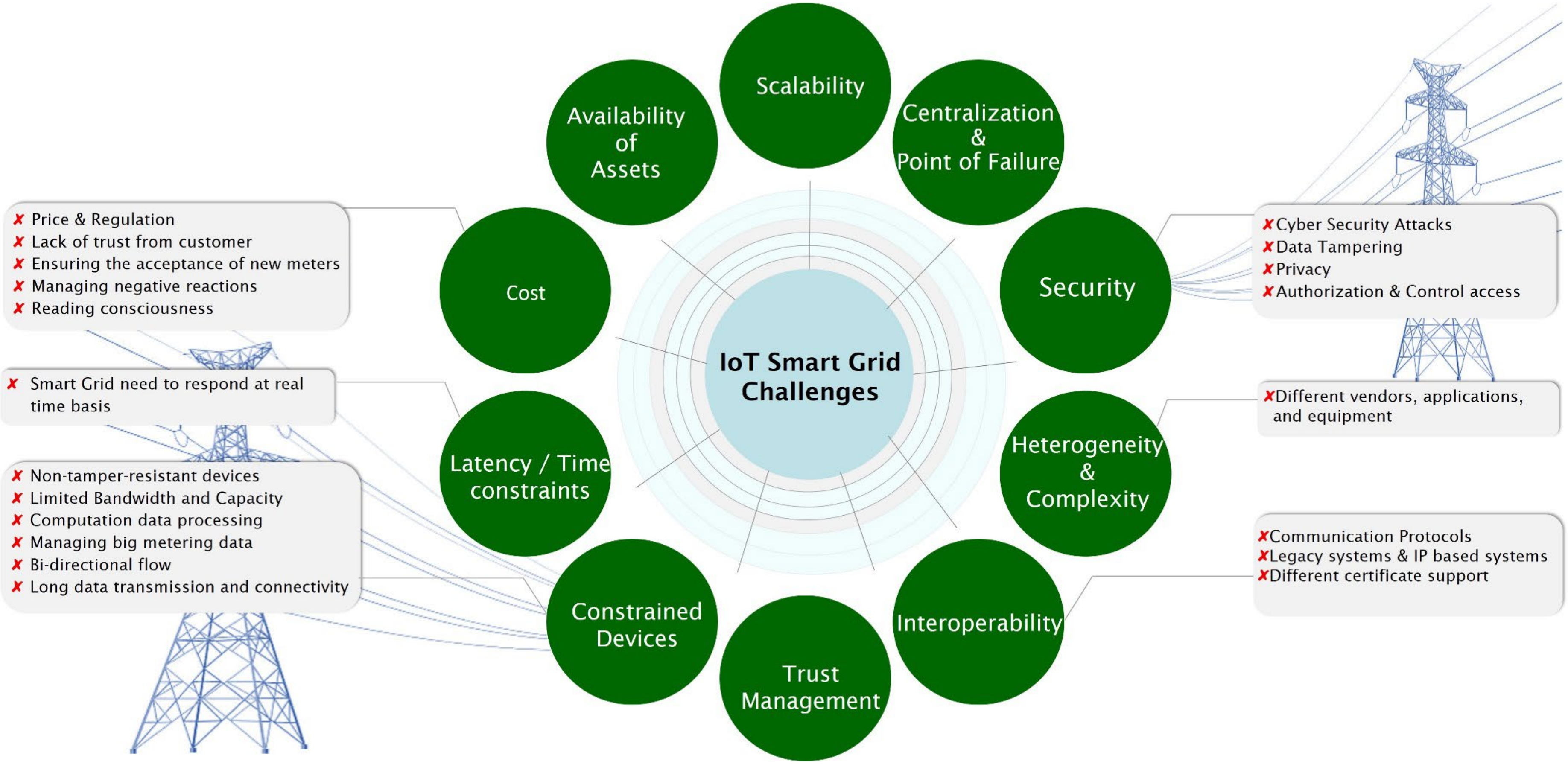

Scalability
Availability of Assets
Centralization & Point of Failure
✗ Price & Regulation
✗ Lack of trust from customer
✗ Ensuring the acceptance of new meters
✗ Managing negative reactions
✗ Reading consciousness
Cost
IoT Smart Grid Challenges
Security
✗Cyber Security Attacks
✗Data Tampering
✗Privacy
✗Authorization & Control access
✗ Smart Grid need to respond at real time basis
Latency / Time constraints
Heterogeneity & Complexity
✗Different vendors, applications, and equipment
✗ Non-tamper-resistant devices
✗ Limited Bandwidth and Capacity
✗ Computation data processing
✗ Managing big metering data
✗ Bi-directional flow
✗ Long data transmission and connectivity
Constrained Devices
Trust Management
Interoperability
✗Communication Protocols
✗Legacy systems & IP based systems
✗Different certificate support

Figure 2-4: IoT-enabled SG challenges

2.9.5 **Security Attack Incidents**

To highlight the risks posed by security attacks to such critical infrastructure, global examples of security incidents are reviewed in Table 2-4 (Kimani *et al.*, 2019; Tufail *et al.*, 2021). These incidents indicate the extent to which security is crucial to the SG. Further analysis of the security aspects is undertaken in Chapter 3.

Table 2-4 presents several types of malware, such as Stuxnet, BlackEnergy, and Industroyer, which target SCADA systems and eavesdrop on the energy and petrochemical sectors. These are designed to target the infrastructure, clearly showing how power systems, gas plants, water plants, and transportation systems are possible targets. All these incidents show how households' or utilities' IoT devices could be hacked to steal or eavesdrop on their information. Stuxnet attack (Iran), Power grid attack (Ukraine), Electricity grid cyber-attack (UK), Petrochemical plant cyber-attack (Saudi) are considered Advanced Persistent Threat (APT) cases due to their sophisticated nature, long-term persistence, and the high level of resources and planning involved.

Table 2-4: Review of security attack incidents

Attack	Location	Year	Details
Tram attack	Łódź, Poland	2008	A tram system hack in which many passengers were harmed. This hack was the first cyber-attack to injure people
Power company attack	Texas, USA	2009	A fired employee compromised the Texas Power Company network, using his credentials to disable power forecasting systems
Stuxnet attack	Iran	2009	A malicious computer worm targeted the SCADA system that controls the automation of electromechanical processes in the Iranian nuclear facility. It is considered a cyber-attack that harmed physical assets. It is alleged that the worm was developed by the US government, destroying the uranium enrichment
Water distribution system attack	Houston, USA	2011	An attack targeting Houston's water distribution system
Bowman Avenue Dam attack	New York, USA	2013	The dam was hacked when the attackers gained control of the floodgates to change the water-flow settings and the number of chemicals used in water treatment, which led to a serious disaster
Power grid attack	Ukraine	2015	Attackers used BlackEnergy or Industroyer malware to breach the power grid by targeting the SCADA system. Subsequently, there was a huge and prolonged electricity failure
Distributed Denial of Service attack	USA	2016	The attackers launched the DDOS to the internet service provider (Dyn), using the Mirai botnet system to shut down the USA's internet. This works by scanning the web for the most insecure IoT devices that keep using factory credentials to sign in. Then, the attackers force these large numbers of IoT devices to send requests to Dyn servers, causing high traffic to overload the system and leading to system failure. Thus, many websites managed by Dyn, such as Netflix, Twitter, Reddit, Spotify, and SoundCloud, became unavailable for a day.
Light-rail system ransomware attack	San Francisco, USA	2016	The attackers executed the ransomware attack on the city's light-rail system. Another example is the attack on the digital teddy bears company's database, which revealed private messages between parents and children, including personal information such as usernames and phone numbers

Attack	Location	Year	Details
Kemuri Water Company hack	USA	2016	The attackers breached the control system of the Kemuri Water Company to manipulate the valves controlling the level of chemicals used in water treatment
Smart building attack	Lappeenranta, Finland	2016	A DDOS attack shut down the water heaters in two smart apartment buildings in winter. This incident is an example of the possible security risk around IoT devices in SG
Electricity grid cyber-attack	UK	2017	A spearphishing attack hacked the control systems of the UK and Ireland’s power grid. This sent an email scam to specific company employees, pretending to be original emails containing technical information on the grid, causing them to click on links to install malicious malware to shut down the electricity grid
Petrochemical plant cyber-attack	Saudi Arabia	2017	This attack was intended to disrupt the operation of the plant and trigger an explosion, which was prevented by a mistake in the attackers’ computer code
Transport network DDOS attack	Sweden	2017	The attack disrupted travel and delayed Swedish trains. Such infrastructure is becoming more vulnerable to attacks because of the adoption of legacy systems in the infrastructure implementation. This is accompanied by attackers’ increasing capabilities, which could complicate security far more.

2.9.6 Review of the Proposed Security Models and Solutions

Security modelling has been carried out for the Smart Grid, but the studies either focused on only a part of the SG or they only partly covered the security controls. The significance of IoT-enabled attacks is not always fully assessed, and the consequences are underestimated, particularly in case of attacks to the end-user. Such attacks threaten home smart appliances, which on the face of it are considered to be indirectly connected to the critical components of the SG. However, according to the aforementioned incidents in Table 2-4, the actual target is not the IoT device itself but the critical components and systems. Also, some topics in the cybersecurity design stage, such as session mismanagement, have not been well investigated. Table 2-5 describes the security modelling efforts reported. The list of disadvantages demonstrates that many challenges relating to security are still open. It is vitally important to develop an appropriate model to address all information security challenges across the whole IoT-enabled SG. Although studies have discussed the optimisation of cost and performance, this research focuses on identifying the main potential access points that are vulnerable to internet-based threats in the SG, and all relevant security controls that could mitigate against the internet-based threats and applicable to each access point. It takes a comprehensive modelling approach that fills in the missing details in the NIST conceptual model without considering their implementation.

This research briefly reviews these related works to:

- Provide a systematic review and categorisation of related efforts
- Point out the differences and therefore the actual contribution of this work.

Security modelling efforts for the Smart Grid (SG) have been conducted, but existing studies often focus on specific aspects or inadequately cover comprehensive security controls. The studies highlighted here demonstrate ongoing challenges in SG security, emphasizing the necessity for holistic security models that address all information security challenges across the IoT-enabled SG landscape. While previous research has explored cost and performance optimization, this review concentrates on identifying vulnerable access points to internet-based threats in the IoT-enabled SG. It emphasizes relevant security controls applicable to each access point within a comprehensive modelling framework, filling gaps in the NIST conceptual model without overlooking implementation costs.

Table 2-5: Summary of security models and solutions

Study	**Function**	**Model Name/Methodology**	**Description (+ = pro, – = con, * = feature)**
(Hahn and Govindarasu, 2011)	Security framework	Algorithm and simulation AMI use cases	* Proposed a security attack exposure evaluation framework for the SG * Evaluated all paths in the SG that attackers could take to access it
(Calderaro *et al.*, 2011)	Detection of faults	Petri Net Model	– Limited to capturing the details of the protection system in a distribution system with distributed generation for failure detection (Otuoze *et al.*, 2018)
(Wei *et al.*, 2011)	Security framework	"bump-in-the-wire" approach	* Discussed the major challenges and strategies required to protect the SG against security attacks from external or internal networked sources * Proposed a conceptual layered framework to protect the automation systems in the SG – Focuses on the network, and communication attacks in particular + Possesses the desired performance.
(Kamto *et al.*, 2011)	Security scheme	Key distribution and management scheme	+ Provides integrity, privacy, and authenticity in the HAN – Vulnerable to Man-in-the-Middle (MITM) attacks – Still has a scalability issue (Benmalek *et al.*, 2019)
(Wang and Yi, 2011)	Security framework	Simulation	* Presented a wireless communication architecture * Developed a new detection mechanism called smart tracking firewall
(Bekara *et al.*, 2012)	Authentication protocol	Mutual authentication in AMI	+ Prevents impersonation attack, MITM attack, Data tampering attack, and replay attack Focuses on non-repudiation and privacy. Does not consider computation overhead and efficiency (Benmalek *et al.*, 2019).

Study	Function	Model Name/Methodology	Description (+ = pro, – = con, * = feature)
(Suleiman *et al.*, 2015)	Threat model	Attack Tree modelling	* Used Attack Tree in threat classification * Built use-case diagrams * Performed SQUARE analysis
(Mahmood *et al.*, 2016)	Authentication scheme	Diffie-Hellman protocol	* Proposed a lightweight authentication scheme – Limited to only focusing on Integrity as a security requirement
(Yan *et al.*, 2017)	Q-learning-based vulnerability analysis approach	Simulation	+ Identifies critical attack sequences and potential sequential vulnerabilities + Considers physical system behaviours – Focuses on sequential attacks only – Does not discuss security controls against sequential attacks
(Khan *et al.*, 2017)	Threat analysis modelling	STRIDE modelling	* Performed the STRIDE per-element approach * Plotted Data Flow Diagram (DFD) for system components
(Jelacic *et al.*, 2018)	Threat analysis modelling	STRIDE modelling	* Used the Industrial Control System (ICS) Smart Grid Architecture, not the SGAM reference model used in this research * Migrated SG to Cloud computing
(Ganguly *et al.*, 2018)	Security measures framework	Fuzzy logic technique	* Detected data theft in AMI smart meters using a modular algorithm called Meter Data Tampering Algorithm (MDTA) * Used a unique one-way cryptographic function * Used a flowchart in detecting data tampering

Study	Function	Model Name/Methodology	Description (+ = pro, – = con, * = feature)
(Mohammadali *et al.*, 2018)	Authenticated key agreement scheme	Key establishment protocol	* Proposed a lightweight authentication scheme + Defends against impersonation, replay, and MITM attacks – Increases computation overload (Benmalek *et al.*, 2019)
(Tonyali *et al.*, 2018)	Privacy protocol	FHE and MPC protocols	+ Uses a Fully Homomorphic Encryption (FHE) system and Multiparty Computation (MPC) system to enable multiple operations to be performed on concealed data + Viable privacy protection mechanism within overhead and performance – Resolves the privacy issue only
(Stellios *et al.*, 2018)	Threat vector identification	Analysis	+ Discusses IoT-enabled cyberattacks in critical infrastructures (CIs) including SGs + Models a threat vector that can be used against IoT devices + Identifies attacks on SCADA, generation, transmission, and distribution – The mitigations and controls are at high-level strategies – Does not cover all access points of the IoT-enabled SG
(Fadhel *et al.*, 2019)	Threat analysis modelling	Attack Tree modelling	* Analysed threats by Attack Tree * Uses Semantic Modelling through the PROV-N semantic notation
(Sani *et al.*, 2019)	Cyber security framework	Simulation	+ Provides a security framework that meets the security requirements + Formal verified model with theoretical analysis – Does not cover all common cyberattacks – Does not include Privacy in the security requirements of the framework
(Das and Zeadally, 2019)	Threat model	Dolev-Yao model and adversary model of Canetti and Krawczyk's	+ Adopts the Dolev-Yao threat model (Dolev and Yao, 1983) and the adversary model of (Canetti and Krawczyk, 2002) + Discusses security attacks, requirements, and controls – Does not include privacy as a security requirement – Does not include all common cyberattacks, such as false data injection
(Gunduz and Das, 2020)	SG cyberattack classification	Based on the CIA triad	+ Provides a taxonomy of the cyberattacks in SG + Better survey of security requirements and countermeasures – Focuses on CIA from only the security requirements

Study	Function	Model Name/Methodology	Description (+ = pro, – = con, * = feature)
(Lázaro *et al.*, 2021)	Identification of vulnerabilities and countermeasures	Survey paper, IEC protocols	+ Surveys vulnerabilities of the communication in the SG + Reveals security mechanisms + Discusses details of International Electrotechnical Commission (IEC) 62351 + Discusses the intermixing between OT and IT networks + Presents specific solutions to threats on IoT-based smart grid applications – Focuses on communication only

Early Efforts and Foundational Models

Hahn and Govindarasu (2011) established a foundational security framework that evaluates potential attack paths within the SG. Despite providing a comprehensive theoretical approach, its lack of real-world application and testing presents significant gaps, particularly in adaptability and practical effectiveness in diverse operational environments. **Calderaro et al. (2011)** utilized a Petri Net Model for fault detection in distribution systems, showing effectiveness in specific scenarios. However, this model's limited scope necessitates expansion and integration with broader system-wide fault detection mechanisms. **Wei et al. (2011)** proposed a "bump-in-the-wire" conceptual framework focusing on protecting SG automation systems from networked threats. Although it laid down significant conceptual groundwork, the model lacks detailed implementation strategies and practical validation in real-world network environments.

Progression in Threat Modeling and Authentication

Suleiman et al. (2015), **Khan et al. (2017)**, and **Jelacic et al. (2018)** significantly advanced threat modeling within SG. They utilized methodologies such as the SQUARE analysis, STRIDE per-element approach, and adapted ICS architectures. These studies deepened the understanding of SG vulnerabilities but often did not integrate their findings into a cohesive, standardized model applicable across different SG architectures. **Kamto et al. (2011)** and **Bekara et al. (2012)** focused on key distribution and authentication schemes, addressing fundamental security requirements such as integrity, privacy, and non-repudiation. However, vulnerabilities such as scalability and susceptibility to MITM attacks highlighted critical areas needing further enhancement.

Recent Advances and Emerging Technologies (2022 - 2024)

(Arshad et al., 2023) introduced a blockchain-based security framework aimed at enhancing trust management by ensuring integrity, thereby enhancing reliability, availability, and privacy of trust information. This framework serves as a guide for future research directions aiming to harness blockchain technology for more secure and reliable IoT applications. (Yaseen, 2023) developed a machine learning-based intrusion detection system (IDS). While this system marks a significant improvement in threat detection, it requires ongoing updates and training to handle new and evolving cyber threats effectively. (Aurangzeb et al., 2024) proposed a quantum-resistant encryption technique designed to secure SG privacy. This approach addresses an emerging concern in cybersecurity but introduces complexities in implementation and integration with existing SG systems.

Gaps and Opportunities for Further Research

The literature review reveals critical gaps in integrating and standardizing security models across the SG ecosystem. While individual studies have advanced knowledge in specific areas, a unified framework that addresses all facets of SG security—including threat detection, data integrity, privacy, and resilience against emerging threats—is still lacking.

Future research should focus on developing adaptable, scalable, and efficient security frameworks that can be standardized across different SG components and operational scenarios. This includes enhancing blockchain frameworks for better energy efficiency, refining machine learning models for dynamic threat detection, and ensuring that new encryption methods are backward compatible with existing infrastructure.

By addressing these gaps, future studies can contribute to creating a more secure, resilient, and reliable Smart Grid environment, safeguarding against both current and emergent cyber threats.

2.10 **Formal Methodologies and Event-B**

In software engineering, one of the ways of constructing reliable systems, despite their complexity, is by formal methods (Clarke *et al.*, 1996). Formal methods have been widely used to specify systems rigorously. Systems built on mathematical elements, such as sets or functions, enable the analysis of these elements to achieve accurate properties, such as completeness and consistency (Butler *et al.*, 2004).

Verification using formal methods is achieved in two ways: model checking; and theorem proving. Model-checking is an automatic process that provides useful information about a system's correctness by building a finite model of the system, while theorem proving uses the system's axioms to prove a property (Clarke *et al.*, 1996).

Event-B is a formal method that can be used to model systems for specification and verification. Event-B was extended from the B-Method (Abrial, 2010), and is a state-based method that uses set theory as a distinctive attribute (Butler, 2013). A system can be modelled gradually to reflect its complexity by the use of abstraction and refinements. Mathematical proofs in Event-B are used to ensure the correctness of each level and consistency among all modelled levels (Butler, 2013).

In the area of computer security, formal methods provide a structured approach to modelling systems, using accurate notation (Butler *et al.*, 2004). The notations can capture the security policies, system properties, and underlying assumptions. Formal methods are used as a verification and validation tool to assure that the system as-built meets its security goals, and as a specification tool to ensure that the system's design is captured (Wing, 1998).

Event-B allows models to be developed gradually via mechanisms such as context extension and system refinement. These techniques enable users to develop target systems from their abstract specifications and, subsequently, introduce more complexity and implementation details. More importantly, properties that are proved at the abstract level are maintained through refinement, hence are guaranteed to be satisfied also by later refinements. As a result, correctness proofs of systems are broken down and distributed amongst various levels of abstraction, which is easier to manage (Hoang, 2013). As Event-B is adopted as the main formal language in this research, an overview of Event-B modelling language notations and syntax is discussed in Appendix C.

When verifying and validating the security model of a Smart Grid system, several methods are available, each with its unique strengths and limitations. Event-B, with its mathematical rigor, systematic refinement process, and robust tool support, makes it the selected formal method to be used in this research. It ensures system correctness at every stage through formal proofs and provides a high degree of assurance, crucial for the critical infrastructure of Smart Grids (Abrial, 2010; Butler, 2013).

Model checking offers a high degree of automation and thorough state exploration, making it useful for detecting concurrency issues. However, it suffers from the state explosion problem, which limits its scalability for complex systems like Smart Grids. Recent advancements in probabilistic model checking have addressed some of these issues, but challenges remain (Katoen, 2010; Parker, 2023). Theorem proving provides unmatched assurance through detailed proofs and can handle complex properties. However, it requires significant manual effort and expertise, making it less practical for large-scale models(Bertot & Cast, 2013). Simulation-based approaches, such as those using MATLAB/Simulink, offer practical and intuitive insights into system behavior, facilitating rapid prototyping and dynamic analysis. However, they lack formal guarantees and can only cover a finite number of scenarios(Alotaibi et al., 2020).

2.11 Conclusion

The literature review has helped in understanding the main concept behind SG implementation, and has highlighted the pressing need to tackle security and adopt efficient solutions. The chapter reviewed the threat incidents and actors, The literature consistently identifies several challenges and risks associated with the implementation of IoT-enabled Smart Grids, particularly in the context of the Saudi energy sector. The literature review highlighted the importance of addressing security concerns during the system design stage of SGs. This directly informed one of the primary research objectives: to develop and propose security control mechanisms tailored for IoT-enabled SGs at the design stage.

Given the complexity and multifaceted nature of the challenges identified, this research adopted a mixed-methods approach. The literature emphasized the need for both qualitative and quantitative analyses to fully understand the security and operational challenges of IoT-enabled SGs. Thus, this research combined statistical data analysis with qualitative case studies to capture a comprehensive view of the issues and potential solutions. Moreover, the literature review contributes to maintaining lightweight security controls that are compatible with the limited computational power and storage capabilities of the IoT infrastructure.

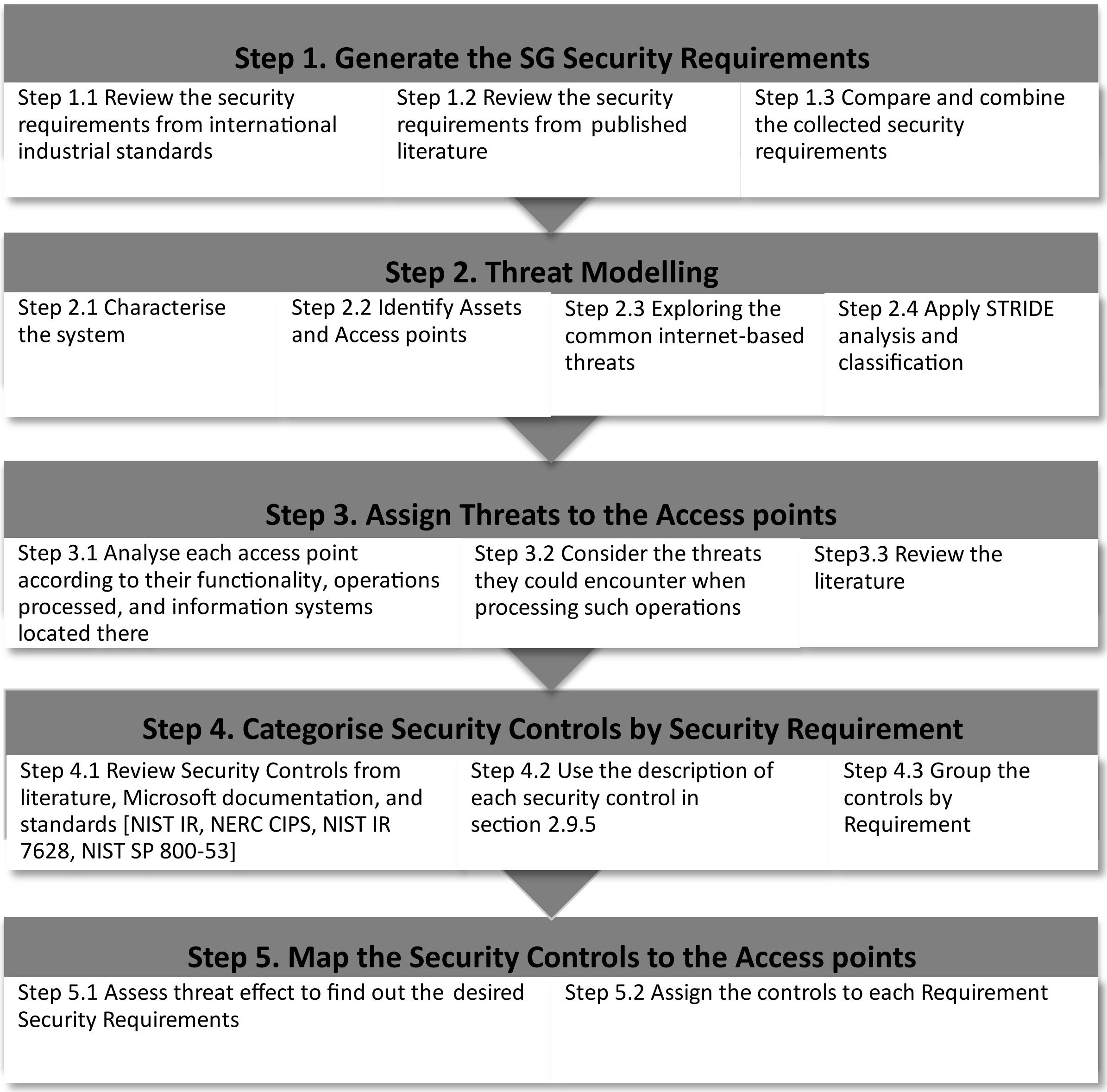
The literature review showed that multiple challenges persist and there is a lack of models that identify all main security requirements for an IoT-enabled SG. Therefore, this research will discuss the network threats and develop a security model considering all main potential access points that are vulnerable to internet-based threats and the controls needed to mitigate those threats.

Chapter 3 **Discussion of Threats and Development Methodology of the Security Model**

Many researchers consider SGs as the largest part of an IoT framework, with thousands of smart objects and entities such as smart meters, actuators, smart appliances, servers, data centres, and other sensors (Bekara, 2014; Reka and Dragicevic, 2018). Exponential growth in both the number of connected objects and demand for electricity has resulted in several issues that need to be addressed for successful implementation of IoT-enabled SG. According to Mckinsey (2019), the number of connected devices in the IoT market was due to reach 43 billion by 2023 (Dahlqvist *et al.*, 2019). This chapter presents the security requirements of the IoT-enabled Smart Grid to develop an information security model. It presents the methodology that is adopted to develop the Information Security Model for the IoT-enabled Smart Grid.

3.1 Method for Model Development

As seen in section 2.6, the NIST conceptual model represents a high-level concept that lacks detailed cybersecurity considerations. NIST case studies and scenarios are limited to privacy and some domains of SG, without linking access points to the security requirements, threats, and controls. Indeed, NIST IR and NERC CIP measure an organisation's compliance with policies. This section charts the research roadmap to develop an Information Security Model for IoT-enabled SGs that fills in the lack of detail in the NIST model. Figure 3-1 presents the steps that the research has undertaken during the development process.



Step 1. Generate the SG Security Requirements

Step 1.1 Review the security requirements from international industrial standards

Step 1.2 Review the security requirements from published literature

Step 1.3 Compare and combine the collected security requirements

Step 2. Threat Modelling

Step 2.1 Characterise the system

Step 2.2 Identify Assets and Access points

Step 2.3 Exploring the common internet-based threats

Step 2.4 Apply STRIDE analysis and classification

Step 3. Assign Threats to the Access points

Step 3.1 Analyse each access point according to their functionality, operations processed, and information systems located there

Step 3.2 Consider the threats they could encounter when processing such operations

Step3.3 Review the literature

Step 4. Categorise Security Controls by Security Requirement

Step 4.1 Review Security Controls from literature, Microsoft documentation, and standards [NIST IR, NERC CIPS, NIST IR 7628, NIST SP 800-53]

Step 4.2 Use the description of each security control in section 2.9.5

Step 4.3 Group the controls by Requirement

Step 5. Map the Security Controls to the Access points

Step 5.1 Assess threat effect to find out the desired Security Requirements

Step 5.2 Assign the controls to each Requirement

Figure 3-1: Development of the Information Security Model for IoT-enabled SG

In Step 1, the security requirements were reviewed from international industrial standards and from academic publications. Then, both sets were combined and compared to generate the security requirements.

In Step 2, threat modelling was carried out, including characterising the system based on the NIST conceptual model, which helped to identify the access points. Then, common internet-based threats were explored. Next, security threats and requirements were both identified using STRIDE analysis and classification.

Step 3 assigned the identified threats to the access points, according to functions and the information system processed at each access point. In Step 4, the security controls were grouped by security requirements. Finally, in Step 5 the security controls were mapped to the access points by assessing threats effects to find out the desired security requirements.

3.2 Developing the Model

In this section a detailed explanation of each step is given to show how the IoT-enabled SG Information Security Model shown in Figure 3-5 was developed.

3.2.1 Step 1 Generate the Security Requirements

As pointed out in section 2.9.1, security represents a challenge to the IoT-enabled SG (EPRI, 2005; Arasteh *et al.*, 2016; Risteska Stojkoska and Trivodaliev, 2017; Bedi *et al.*, 2018; Reka and Dragicevic, 2018; Shakerighadi *et al.*, 2018; Ganguly *et al.*, 2019; Kimani *et al.*, 2019). This encouraged both the academic and the industrial sectors to establish the requirements to be met to boost the security of the SG information system. The security requirements are what the SG needs to deliver to enhance security. In this section, the security requirements gleaned from literature and industrial standards, guidelines, best practices, and authorities (such as DOE, NIST 7628, EPRI, ENISA, IEC62351, and IoTSF) were reviewed and analysed, together with numerous studies (Wang and Yi, 2011; NISTIR 7628, 2014; Kumar Suman et al., 2017; Hussain et al., 2018; Kimani et al., 2019).

Step 1.1 Review the security requirements from international industrial standards, guidelines, and best practice

This sub-section reviews the security requirements of international and industrial standards, guidelines, and best practices in SG by authorised bodies. Table 3-1 summarises the security requirements from those bodies. Confidentiality, Integrity, Availability, Authentication, and Authorisation are marked as an (Explicit) mentioning for all standards since they are typically fundamental security requirements. Non-repudiation is marked as an (Implicit) for the standards shown in Table 3-1, as it is often indirectly ensured through mechanisms that support other security goals like integrity.

Table 3-1: Security requirements from industrial standards, guidelines, and best practice

Organisation / Security goals	IoT Security Foundation	DOE	NIST 7628	EPRI	ENISA	IEC 62351
1. Confidentiality	Explicit	Explicit	Explicit	Explicit	Explicit	Explicit
2. Integrity	Explicit	Explicit	Explicit	Explicit	Explicit	Explicit
3. Availability	Explicit	Explicit	Explicit	Explicit	Explicit	Explicit
4. Non-repudiation	Implicit	Implicit	Implicit	Implicit	Explicit	Explicit
5. Authentication	Explicit	Explicit	Explicit	Explicit	Explicit	Explicit
6. Authorisation	Explicit	Explicit	Explicit	Explicit	Explicit	Explicit

NIST (2014) provides NISTIR 7628 Rev.1 as a guide to organisations that are implementing SG systems in selecting and modifying their security requirements. This guide aims to achieve reliability on the SG side and privacy on the consumer side. The security requirements are identified by NIST as **Availability**, **Integrity**, and **Confidentiality** – CIA.

Not only the NIST but many other authorities are developing security standards, such as the International Electro-Technical Commission (IEC) IEC 62351, the European Network and Information Security Agency (ENISA), and the IoT Security Foundation (IoTSF). All these bodies similarly consider CIA in their security requirements for the SG. Besides the CIA, ENISA considers the related security requirements of **authentication, authorisation,** and **non-repudiation** (Egozcue *et al.*, 2012):

- **Authentication** is the process of verifying that the actor is the one who actually alleges to act (Egozcue *et al.*, 2012)
- **Authorisation** is the process of Granting the access to an authenticated actor (Egozcue *et al.*, 2012)
- **Non-repudiation** is the process of verifying that an action has been produced by the original responsible node (Egozcue *et al.*, 2012)

In terms of consumer IoT and industrial IoT, in the international standards (IoTSF, 2020) all the security requirements identified are the same, in SG. In terms of prioritising confidentiality, integrity, and availability, as seen in Figure 3-2, some organisations such as NIST put the requirements in the case of SG consumption, ICT systems, or smart meters in the order C, I, A, and in the case of control systems and industrial SG automation in the order A, I, C, where the availability of the system is the most important (Egozcue *et al.*, 2012; Wang and Lu, 2013). This variation in priority may affect the security controls to be discussed later in this research.

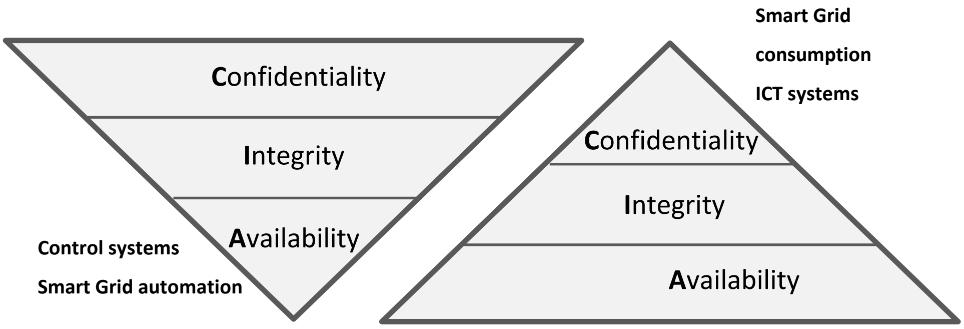


Figure 3-2: Security requirement priorities

Step 1.2 Review the security requirements from published literature

The security requirements for IoT-enabled SG are defined in various publications (Ling and Masao, 2011; Pallotti and Mangiatordi, 2011; Aloul, 2012; Wang and Lu, 2013; Bekara, 2014; Rawat and

Bajracharya, 2015; Armoogum and Bassoo, 2018; Camachi *et al.*, 2018; Mrabet *et al.*, 2018; Benmalek *et al.*, 2019; Das and Zeadally, 2019; Tufail *et al.*, 2021):

- **Confidentiality:** Ensures that access to transmitted data is restricted to authorised people. It prevents the unauthorised disclosure of information. In SG, the transmitted data could be sensitive, such as personal information about a consumer's activities or home appliances, consumption rate, billing data, and whether they are away from home. If confidentiality is compromised, an attacker may target the data. All (Wang and Lu, 2013; Anwar and Mahmood, 2014; Rawat and Bajracharya, 2015; Ganguly *et al.*, 2018) have argued that confidentiality is vital to protect the privacy of both the consumer and the utility.

- **Integrity:** Guarding the information and the source of the information against any tampering or unauthorised manipulation. It also ensures the non-repudiation of information. Information could be power measurements or price signals. For instance, accurate billing integrity is vital in smart meter data sources, and a loss of integrity may lead to false decision-making about energy management. For example, modifications to the data reported to the utility may generate the wrong power totals, exposing the service to an attacker wishing to steal money.

- **Availability and Reliability:** Guarantee timely and reliable access to the information (NISTIR 7628, 2014). The power system needs to be available whenever required by authorised entities. A loss of availability may cause power cuts. Availability is about the uptime and downtime of the SG system. It is worth explaining the difference between availability and reliability in security. If an SG, as an asset, never shuts down, this represents that it is 100% reliable; however, it may have downtime once a month for routine maintenance, in which case its availability would be less, yet its reliability remains 100%, as this is planned downtime.

- **Authentication**: Validating the identity of any communicated entities (devices/users) in the SG, such as the millions of smart home appliances that are connected to smart meters. For example, smart meters need to be authenticated so that the utility company can bill the correct consumer. Data authentication plays a significant role in proving that the transmitted data are genuine, using verification features such as digital signatures. In other words, authenticity verifies that devices/users are secure from unauthorised access.

- **Authorisation and Control Access:** Granting the required rights to an authenticated device/user to access SG resources. Access control is that which guarantees that SG resources are accessed by the correctly identified entities.

- **Privacy:** Guarantee that any private data belonging to the consumer cannot be obtained without permission and is used for pre-approved purposes only. An attacker can extract information on private data from the smart meter. Private data could be consumption readings or information about habits such as sleeping times.
- **Non-repudiation and accountability**: Assuring that the accountability of any data transaction has been undertaken between entities without any denial of responsibility. It means assuring the traceability of the system by recording each transaction by node, device, consumer, and utility (Mrabet *et al.*, 2018). Non-accountability of entities could be used by an attacker negatively, so the preserved record can be used as evidence to identify the attacker (Mrabet *et al.*, 2018).

Table 3-2: Security requirements from published literature

Security requirements/ criteria	**Viswanathan et al. (2017)**	**Ganguly et al. (2018)**	**Vashistha & Barbhuiya (2019)**	**Rawat & Bajracharya (2015)**	**Bekara (2014)**	**Anwar & Mahmood (2014)**	**Wang & Lu (2013)**	**Ling & Masao (2011)**	**Pallotti & Mangiatordi (2011)**	**(Das and Zeadally, 2019)**	**El Mrabet et al. (2018)**	**Aloul et al. (2012)**	**Gunduz and Das (2020)**	**Benmalek et al. (2019)**	**Tufail et al. (2021)**
Confidentiality	E	E	I	E	E	E	E	E	E	E	E	E	E	E	E
Integrity	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E
Availability	I	E	I	E	I	E	E	E	E	E	E	E	E	E	E
Authentication	E	I	E	E	E	I	I	I	I	E	I	I	E	I	E
Authorisation	I	I	I	I	E	I	I	I	I	E	I	I	E	I	E
Privacy	I	I	E	I	E	I	I	E	I	I	I	I	E	I	I
Non-repudiation	I	E	E	I	I	I	I	I	I	E	E	I	E	I	I

Table 3-2 shows the publications that explicitly (E) or implicitly (I) address the security requirements for the SG relevant to the focus of this research. By mapping these studies to specific security requirements, this research identifies a comprehensive list of requirements related to the core aspects of the research topic. The selected studies span a range of years, demonstrating the evolution of thought and approach to security in SG.

Step 1.3 Compare and Combine the Collected Security Requirements

A list of security requirements is generated by comparing this set of security requirements to the requirement set found in the International Standards. By combining the common requirements in both sets, those considered to be the chief security requirements in this research are as follows: **confidentiality, integrity, availability, authentication, authorisation, privacy, and non-repudiation**.

These requirements serve as general guidance. There is a need for standards customised to the system that is targeted (Myagmar *et al.*, 2005). This can be done by specifying the security requirements using a systematic threat modelling, as shown in the next step.

3.2.2 Step 2 Threat Modelling

Threat modelling is the process of analysis that allows security experts to discover the potential vulnerabilities to be addressed (Swiderski and Snyder, 2004). Threat modelling for an IoT-enabled SG during system design identifies the necessary controls and countermeasures (Khan *et al.*, 2017): potential attacks need to be identified at the system design stage by the security designer, not the attacker (Myagmar *et al.*, 2005). Many modelling techniques have been reported, such as PASTA model, Attack Trees, LINDDUN, OCTAVE, and DREAD. However, the comparison between STRIDE and other threat modeling methodologies in Table 3-3 highlights the reason behind selecting STRIDE for this specific research:

Table 3-3: Comparison of Threat Modeling Techniques

Aspect	**STRIDE**	**PASTA**	**Attack Trees**	**LINDDUN**	**OCTAVE**
Focus/Perspective	Defender-centric	Risk-centric	Attacker-centric	Privacy-centric	Operational risk-centric
Scope	General security threats	Detailed risk analysis	Graphical attack representation	Privacy threats	Operational risks
Visualization	Uses DFDs	Uses various modeling techniques	Attack tree diagrams	Uses DFDs	Asset and vulnerability mapping
Automation Support	Strong (Microsoft Threat Modeling Tool)	Limited automation	Limited	Limited	Limited
Maturity	High	High	High	High	Medium
Consistency	Consistent results	Variable depending on complexity	Consistent but can be complex	Consistent within privacy scope	Variable, context-dependent
Ease of Implementation	Straightforward	Complex and resource-intensive	Can become complex	Specific to privacy issues	Complex and resource-intensive

Real-world Validation	Validated through case studies	Effective for risk management	Effective but dependent on complexity	Effective in privacy contexts	Effective in operational contexts

Compared to other methodologies, STRIDE offers strong automation support through Microsoft's Threat Modeling Tool, ensuring consistent and reliable results. While PASTA provides a detailed risk analysis, it is complex and resource-intensive (Khan et al., 2017; Kim et al., 2022). Attack Trees offer a graphical representation of attacks but can become cumbersome for large systems (Kim et al., 2022; Myagmar et al., 2005). LINDDUN focuses specifically on privacy concerns, limiting its scope, and OCTAVE, while effective in operational risk contexts, is also complex and resource-intensive. DREAD went out of use at Microsoft by 2008 (Shostack, 2008).

Given the need for a systematic, and well-supported approach to threat modeling in CPS, STRIDE is the optimal choice. STRIDE is a well-known de facto and mature technique for performing architectural threat analysis and finding threats to a system (Kim et al., 2022). It includes a full breakdown of the processes, data flows, and interactions (Kim et al., 2022). Its focus is defender-centric which makes it highly effective in identifying and mitigating threats during the design phase of system development. STRIDE's integration with data flow diagrams (DFDs) allows for clear visualization and understanding of system interactions, which is crucial for complex environments like Smart Grids (Kim et al., 2022). Therefore, STRIDE technique is used for threat modeling and classification in this research because:

- It fits the identified security threats and its categorisation of threats is comprehensive
- It fits the security properties, including authentication, integrity, non-repudiability, confidentiality, availability, and authorisation, as shown in Table 3-3
- It is a systematic and relatively lightweight technique to identify relevant security mitigating controls that address the threats at key component levels of the system, making it highly suitable for distributed complex environments like Smart Grids (Kim et al., 2022).
- The effectiveness of STRIDE has been validated through real-world cybersecurity incident cases, such as Stuxnet and Triton, highlighting its robustness in identifying and mitigating threats in Industrial Control Systems such as Smart Grids (Kim et al., 2022).

STRIDE categorises security threats on the basis of their effects in six classes (Microsoft, 2009, 2021), as in the following table Table 3-4.

Table 3-4: STRIDE threat model categories (Swiderski and Snyder, 2004; Shostack, 2019)

Threat	Desired Property	Threat Definition
Spoofing	Authentication	Impersonating something or someone else
Tampering	Integrity	Modifying data
Repudiation	Non-repudiability	Claiming to have not acted

Information disclosure	Confidentiality	Exposing information to someone not authorised to see it
Denial of service	Availability	Reducing the ability of valid users to access resources or services
Elevation of privilege	Authorisation	Gaining capabilities without proper authorisation

Since there is no standard approach to applying STRIDE in cyber-physical systems (Khan *et al.*, 2017), three high-level steps are applied in this research to establish the security controls of the system.

1. **Characterising the System:** This aims to identify the system components and the interconnections between them
2. **Identifying the Assets and Access Points:** This determines the assets and access points. Assets are the resources that need to be secure from attack and could be processes or data. Access points represent the points of vulnerability that the attacker may use to gain access to the SG
3. **Identifying Threats:** This describes all possible attacks that need to be countered.

In the next section, each of these threat modelling steps is elaborated in the context of an IoT-enabled SG.

Step 2.1 Characterising the System

This research characterises the system according to the following aspects:

- The NIST generic conceptual model for the main components of SG and their interconnection: generation, transmission, distribution, customer, service providers, markets, and operations (NIST, 2014), as seen in Figure 2-1. Details of each component are given in Table 2-2
- Addressing the information flow, not the power flow based on the NIST generic model
- Concerning internet-based threats against an IoT-enabled SG
- Characterising the system by element. According to (Shostack, 2014), STRIDE can be undertaken in two ways: STRIDE per element; and STRIDE per interaction. The first analyses the operations of each system component. The second analyses the function of the interaction between two components in the system, for example identifying all possible threats for the interaction "utility sends data to the server".

Step 2.2 Identifying Assets and Access Points

This step articulates the main access points that are vulnerable to internet-based threats in the IoT-enabled SG by reviewing publications and the vulnerability analysis compiled by the US electric sector, issued by Idaho National Laboratory (Glenn *et al.*, 2017). This report lists the vulnerabilities as networks, communication, devices, remote access and mobile devices, and third-party/supply chain.

Access points could be SG elements or any interconnection that exchanges information over the SG. According to the NIST conceptual model, SG components are: generation plants, transmission stations, distribution substations, customers, service providers (utility), markets, and operations (NIST, 2014). **The distribution substation** is one of the SG components responsible for performing automation functions, such as receiving energy from generating plants, regulating energy distribution, and managing suddenly increased energy. It transmits operational data to SCADA for control purposes. It consists of regulators and distributors, such as (RTU), the global positioning system (GPS), the Human-Machine Interface (HMI), and intelligent electronic devices (IED) (Mrabet *et al.*, 2018).

By combining the views of the US DOE (2018), Al Khuffash (2018), and Bekara (2014), it can be seen that SG comprises two major elements: smart devices; and AMI (Bekara, 2014; Al Khuffash, 2018). Some studies consider smart meters to be a part of AMI (Mohassel *et al.*, 2014; Mrabet *et al.*, 2018).

Smart devices represent the physical infrastructure of the SG and include smart meters, smart appliances, sensors, phasors, measuring units, and circuit breakers. Smart meters are digital meters consisting of a microprocessor and local memory, and they represent the fundamental blocks with which to build an SG (Rahman, 2009). They measure and collect energy consumption data with a timestamp, which is crucial to delivering electricity in a reliable manner. Also on the utility side, smart meters transmit data in real-time to the AMI. The smart meters are installed on the consumer side and at other locations around the SG, and report information annually, monthly, daily, hourly, or even each second for the purpose of management and control. Smart meters record other information, such as voltage and current, for both consumers and utilities due to their two-way capability.

From consumers' perspective, smart meters raise consumption consciousness by informing them of their average usage, advising them of peak demand times, and alerting them when a specific usage limit is reached. Therefore, smart meters can contribute to an energy-efficient economy and energy conservation to manage the rapid growth in demand (Bekara, 2014; Al Khuffash, 2018; Ghasempour, 2019).

From the utility perspective, smart meters enable monitoring and the detection of power theft. They provide failure/shortage notifications, as well as real-time overviews of grid status, to support decision-making on electricity generation, distribution, load balancing, and scheduling. Moreover, they assure a swift response to any controlling commands, including shortage management, software upgrades, on/off turns, and pricing systems. They enhance the planning process by capturing the information so that, with sophisticated analysis, utilities can predict future usage and demand patterns (Flick and Morehouse, 2011; Bekara, 2014; Al Khuffash, 2018; Ghasempour, 2019).

AMI, like a smart device, enables two-way communication between smart meters and utilities. Before the AMI, Automatic Meter Readings (AMR) allowed only unidirectional communication: from smart meters to utilities (Ghasempour, 2019; Martins *et al.*, 2019). AMI collects, analyses, measures, and stores the energy data sent by a consumer's smart meter to the utility's information management systems. The AMI transmits requests, command signals, notifications, recommendations, pricing information, and software updates from the utilities back to the consumer's smart meter (Bekara, 2014; Mohassel *et al.*, 2014; Al Khuffash, 2018; Ghasempour, 2019). It consists of three elements: (i) a smart meter; (ii) the AMI headend; and (iii) concentrators or collectors.

The **AMI headend** is an AMI server that includes the Meter Data Management System (MDMS) that sends the metering information to the utility. Communication with smart meters is established using communication protocols such as Zigbee and Z-wave (Mrabet *et al.*, 2018).

The communications network aims to enable data-sharing and exchange between IoT smart devices and the utility side (US Department of Energy, 2018). It includes the network itself and transmission and distribution devices such as switches, voltage regulators, capacitors, and transformers (PTI, 2011; Al Khuffash, 2018). The network collects information from smart meters and transmission and distribution devices to aid in diagnosing and monitoring network status, thereby providing supply distribution (Gungor *et al.*, 2010). The communication network is standardised by IoT to reduce the number of protocols that have to be used to communicate. Al-Ali and Aburukba (2015) proposed the 6LowPAN communication protocol as the backbone of the IoT communication layer in SG. SG employs ICT with a centralised architecture (Al-Omar *et al.*, 2012; Al-Ali and Aburukba, 2015; Yang, 2019). According to the DOE (2018), ICT is what makes the grid smart. As a CPS, SG uses ICT to monitor, manage, and control its processes and physical assets, including substations, transformers, circuit breakers, smart meters, and cables (Khan *et al.*, 2017). Thus, ICT is the most important characteristic of the SG, and it is the key factor in designing IoT systems.

Substation Automation Systems (SAS) and The supervisory control and data acquisition system (SCADA) is the control system of the power grid that is situated on the control centre side, and is composed of three elements (Mrabet et al., 2018):

- Remote terminal unit (RTU): a device consisting of three elements used, respectively, for data acquisition, instruction execution for the Master Terminal Unit (MTU), and communication
- Master terminal unit (MTU): a device that controls the Remote terminal units (RTU)
- Human-Machine Interface (HMI): a graphic interface for the SCADA system
- Intelligent electronic devices (IED): or a Programmable Logic Controller (PLC), in other organisations' contexts.

According to the IEC 61850 standard, the Substation Automation System (SAS) comprises the SCADA system, Master Terminal Unit (MTU) that contains a server, data historian/database, Human Machine Interface (HMI), and communication network, as these are all contained in the smart modern Control Centre. This is discussed later in Chapter 5 (Zeynal et al., 2014). In this research and according to field experts, there is a SAS Control centre for each type: generation plant; transmission; and distribution substations.

The current research considers that MTU, PLC, RTU, IED, and SCADA are all names for the same access point, known as SCADA or a more intelligent SAS control centre.

Information systems are essential for processing, computing, analysing, and accessing the data collected from digital devices in the SG. Information systems of SG can be classified into the following subsystems (US Department of Energy, 2018), according to their location (Wang *et al.*, 2019), to supervise, monitor, control, and manage the generation, transmission, and distribution operations, respectively:

- On the generation side, such as Generation Management System (GMS), Supervisory Information System (SIS), and Demand Response Management (DRM)
- On the transmission side, such as Energy Management System (EMS), electricity operation system, and decision-making system
- On the distribution side, such as the Distribution Management System (DMS)
- On the utility side, such as Customer Information System (CIS)
- On the SCADA side, such as Substations Automation System (SAS). (Zeynal *et al.*, 2014)

In this research, these subsystems are represented as the Substations Automation System (SAS) control centre. Each element and asset mentioned above is a vector or access point, the most likely to be exploited to execute a cyberattack. These are vulnerable to internet-based threats. Figure 3-3 shows seven potential access points as red arrows: (1) smart meters and smart appliances; (2) transmission stations, distribution substations, and Smart automation devices for transmission and distribution (switches, sensors, actuators, transformers, voltage regulator, capacitors); (3) generation plant and Information Communication Technology (ICT) systems; (4) AMI; (5) SAS/control centre; (6) utility data centre; and (7) market.

As this research focuses on the IoT paradigm, we assume that IoT devices (including IoT sensors and actuators) communicate with the aforementioned physical access points and systems of the SG. In turn, those IoT devices are the initial entry points for an attack to gain access to physical electricity access points (Stellios *et al.*, 2018). So, access points and the IoT devices that communicate with them are considered as a single unit, viewed as an IoT-enabled access point.

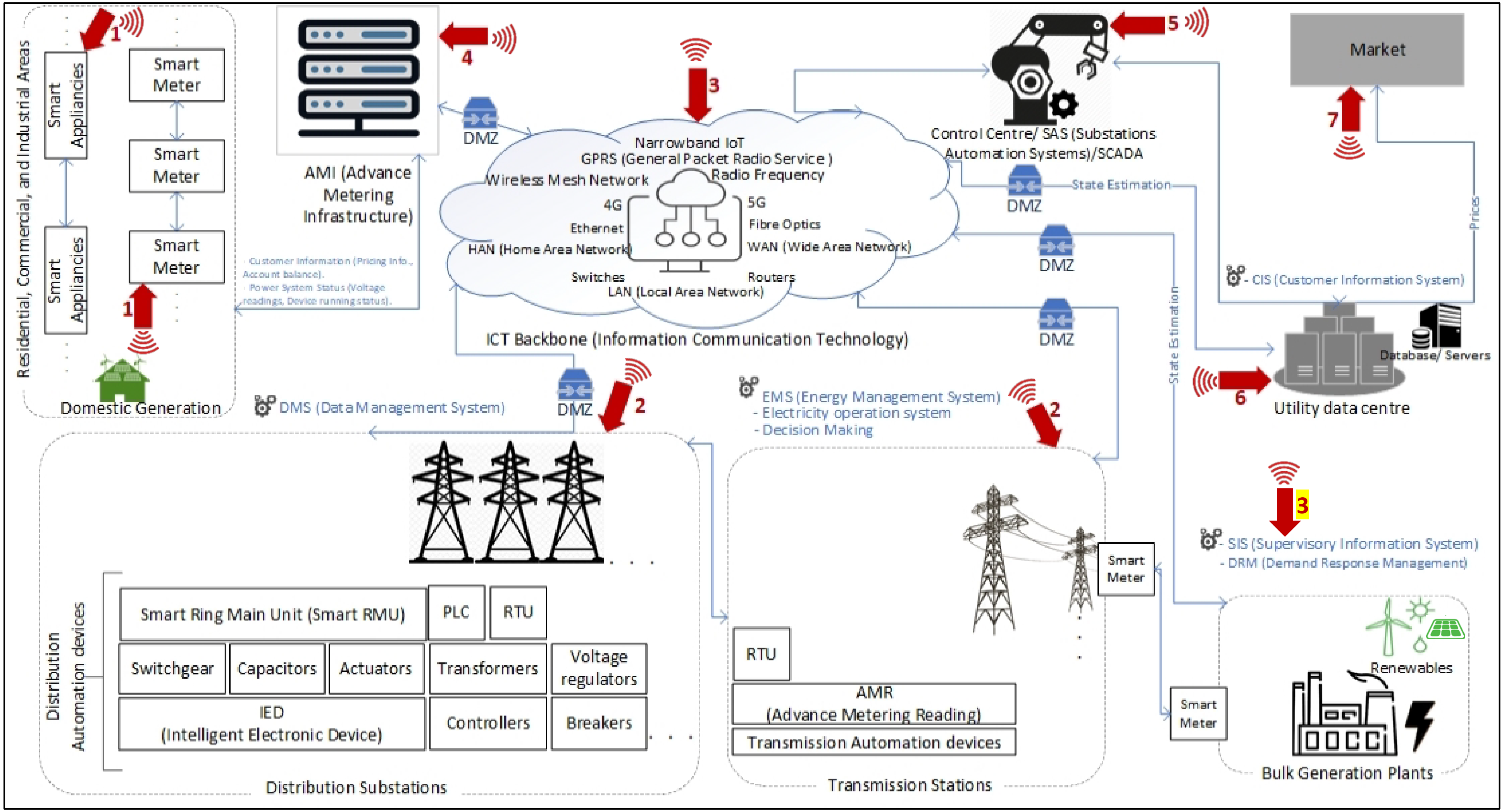

Residential, Commercial, and Industrial Areas
Smart Appliancies
Smart Appliancies
Smart Meter
Smart Meter
Smart Meter
1
1
Domestic Generation
4
AMI (Advance Metering Infrastructure)
Customer Information (Pricing Info., Account balance).
Power System Status (Voltage readings, Device running status).
DMZ
3
Narrowband IoT
GPRS (General Packet Radio Service)
Wireless Mesh Network
Radio Frequency
4G
5G
Ethernet
Fibre Optics
HAN (Home Area Network)
WAN (Wide Area Network)
Switches
Routers
LAN (Local Area Network)
ICT Backbone (Information Communication Technology)
5
Control Centre/ SAS (Substations Automation Systems)/SCADA
DMZ
State Estimation
DMZ
DMZ
Market
7
Prices
CIS (Customer Information System)
State Estimation
Database/ Servers
6
Utility data centre
DMS (Data Management System)
DMZ
2
EMS (Energy Management System)
- Electricity operation system
- Decision Making
2
3
SIS (Supervisory Information System)
- DRM (Demand Response Management)
Distribution Automation devices
Smart Ring Main Unit (Smart RMU)
PLC
RTU
Switchgear
Capacitors
Actuators
Transformers
Voltage regulators
IED (Intelligent Electronic Device)
Controllers
Breakers
Distribution Substations
Smart Meter
RTU
AMR (Advance Metering Reading)
Transmission Automation devices
Transmission Stations
Smart Meter
Renewables
Bulk Generation Plants

Figure 3-3: Access points vulnerable to internet-based threats in IoT-enabled SG

IoT device
Potential access

Step 2.3 Reviewing Threats

The focus of this research is the cybersecurity of information. The threats included in this research are those applicable to the cybersecurity of the IoT-enabled SG, namely threats aimed at information and communication assets. This means that internet-based threats are the set included in this research. This step aims to identify threat types, rather than to delve into every single specific example. So, the types of threats are internet-based cyber threats. Some target infrastructure and physical assets, while others target information and communication networks such as ICT systems and control systems such as SCADA.

The following steps were followed in reviewing the threats:

1- **Review the common types of threats that apply to SGs and are published by authorities such as ENISA (Marinos and Lourenço, 2018) and Cisco (Cisco, 2017):** In this step, only internet-based cyberattacks are included in this research. Some are excluded; such as the effects of the original attack, for instance failures of a device/system, failure of communication network, damage, and loss. Non-technical threats such as natural disasters are also excluded.

2- **Review the literature:** Aloul *et al.*, 2012; Knapp and Samani, 2013; Wang and Lu, 2013; Bekara, 2014; McCary and Xiao, 2015; Tazi *et al.*, 2016; Khan *et al.*, 2017; Mrabet *et al.*, 2018; Otuoze *et al.*, 2018; Tonyali *et al.*, 2018; Ganguly *et al.*, 2018, 2019; Benmalek *et al.*, 2019; Das and Zeadally, 2019; Kimani *et al.*, 2019; Gunduz and Das, 2020; Tufail *et al.*, 2021; Ding *et al.*, 2022.

In this step, the insider attack is excluded as it is considered in this research to be a threat actor. In addition, the information leakage attack is excluded as it is considered a vulnerability.

3- **Filter the collected SG threats to include only cyber threats:** In this step, physical attacks such as fraud, extract RAM/firmware attacks, and meter bypass attacks are excluded.

4- **Filter the cyber threats to include the cyber threats only:** In this step, only network-based threats are included. So, the social engineering attack is excluded and limited to phishing attacks. Also, the Advanced Persistent Threat (APT) is excluded as this research does not investigate such sophisticated, multi-stage threats as APT.

5- **Analyse and group the IoT cyber threats based on:**

   a. STRIDE categorisation in Table 3-3
   b. Threats behaviours
   c. Type: whether is it an active or passive threat.

In this step, many threats are regrouped on the basis of steps a, b, and c, thus hijacking, session hijacking, cryptojacking, ransomware, Worm, Stuxnet, and Trojan attacks are grouped under Malware attack. Also, DNS Tunnelling comes under Malware attack, as the concept of its behaviour is to embed malicious code into a message that appears to be DNS. Moreover, Zero-day exploits and attacks are come into this category, as an attacker releases malware to exploit a vulnerability before the developer can patch it.

Buffer overflow, jamming and DDOS are all regrouped as DOS attack. Similarly, reconnaissance and traffic analysis are regrouped into the Eavesdropping family of attacks, as they rely on exploration and Interception techniques. Popping the HMI attack is also grouped into this Eavesdropping family, because it breaches the HMI console of the SCADA/SAS/Control Centre/EMS or substation then establishes remote access to the console, so the threatened system’s desktop is returned to the attacker's machine for monitoring and obtaining unauthorised control of the system.

Masquerade attack is classified as a Spoofing attack. XSS attack is a type of injection attack, in which a threat actor injects a malicious script, therefore it is grouped under False Data Injection attacks.

Finally, Manipulation, misuse of information, and meter measurement modification are all regrouped under Data Tampering attacks. Data steal could be the desired action of many further types of attack where, for example, an attacker spoofs an identity then, as a final action, steals data. However, data steal is grouped under Tampering.

As a result of the these steps, the IoT SG’s cyber threats are concluded to be as follows, including definitions and brief descriptions:

1. **Spoofing/Impersonation:** This is an active attack that aims to communicate on behalf of a legal entity through unauthorised access by stealing its identity. An attacker may impersonate another’s smart meter identity in order to pay lower electric charges – or let the other pay.

2. **Eavesdropping/Traffic analysis/Man-In-The-Middle (MITM):** Eavesdropping is a passive attack that captures transmitted data by intercepting communications between two entities in the SG. This happens because entities on the IoT-enabled SG usually communicate using public communication infrastructure. It is also called sniffing, snooping, or traffic analysis. In traffic analysis, the attacker intercepts the communication as well as monitoring it, analyses the network traffic, then extracts information from patterns in that communication. Its aim is to locate key entities such as substations or disclose sensitive information (such as power consumption, future price information, routeing structure, or the SG control structure) (Teng *et al.*, 2012). The Man-In-The-Middle attack (MITM) is a form of eavesdropping, and it is one of the most common types of threats that capture data using a sniffer or protocol analyser. It relies on spoofing. An attacker pretends to be an intermediate entity

between two legitimate entities. By default, entities suppose that they are communicating with each other directly, but in reality the attacker is in the middle. SCADA – the control system – is intercepted in this type of threat.

3. **Replay attack**: This is an active attack that intercepts communications between two entities by recording, observing, copying the transmitted data, then replaying a selected part of the copied data back. It manipulates the data before sending it back.

4. **Data tampering:** This strikes when an attacker manipulates the exchanged data, such as dynamic prices that are announced before peak times, making them cheaper. Consequently, it can increase consumer consumption instead of reducing it. This overloads the power network and causes power outages. All smart meters, sensors, actuators, transformers, circuit breakers, and cables are susceptible to this type of threat. Data hacking is an advanced form of data tampering. This aims to earn money by stealing power, and is accompanied by a structured systematic criminal act or terrorism.

5. **Denial of Service (DOS)/Jamming channel**: This is an active attack that floods the entire system, resources, or bandwidth with a high number of fake requests to overload the system, slow it, or corrupt data transmission, thus making the SG unavailable. This congested traffic prevents authorised entities, such as ICT systems, devices such as smart meters, substations, AMI, or other network resources and services, from accessing the system. This arises from use of the IP protocol and TCP/IP stack in SG. In the presence of IoT in SG the risk increases, as described by Yilmaz and Uludag (2017). A jamming channel attack is a type of DOS threat. Jamming could hack substations and wireless networks in SG. A distributed DOS (DDOS) threat involves system servers or resources being flooded by multiple attackers.

6. **Malware injection:** This is the execution of malicious software on the SG, such as viruses, spyware, rootkits, adware, malvertising, ransomware, Trojan horses, or worms. It aims to damage, steal, delete, modify, or disable the main functions in smart meters, smart appliances, systems, or utility servers. WannaCry ransomware is an example of a malware threat that is used to deny consumers access to the core services unless they pay a ransom.

7. **Phishing:** The phishing that is included in the scope of this study takes the form of social engineering attacks that are internet-based phishing, such as email phishing and search engine/website phishing. This tricks users into believing that a message is from a trustworthy organisation, asking them to click a malicious link. The aim is to obtain sensitive information via email, malicious websites, or a search engine. When users respond, the attacker can use this information to access the system. Spearphishing is an example of internet-based phishing that targets a specific group or

type of individual, such as an organisation's system administrator (CISA, 2009). Non-internet-based social engineering attack tricks users, through human interactions, into giving sensitive data to fraudsters. It uses many techniques, such as psychological manipulation and impersonation by phone or voice calls, and this form of non-internet-based phishing, such as smishing and vishing (CISA, 2009), is beyond the scope of this research.

8. **SQL injections:** A Structured Query Language (SQL) injection executes a harmful SQL query statement on a server that uses SQL, aiming to force it to disclose information and modify or delete the database contents. According to Cisco, this SQL query is entered by the attacker using a website search box on the client-side interface of the application and is used to target database applications. It is an example of the database links threat, in which the database of the control system (SCADA) is exploited.

9. **False data injection:** This type of attack sends fake information into the network, such as false meter readings or wrong prices. The attack could be carried out against energy distribution as well as against grid state estimation (Marinos, 2013). It causes false state estimation for the SCADA system and may cause a power system failure. Thus, it influences the electricity market financially by tampering with market price information.

Step 2.4 Applying STRIDE

The threat model used in this research is STRIDE. In developing a threat model, security designers are concerned with defining threats (Myagmar *et al.*, 2005). Security requirements can be mapped to threats to show the effect of each threat and the required security criteria of the system. It is argued that security requirements for the system can be defined clearly once the threats are identified, as shown in Table 3-5 and Table 3-6.

Table 3-5 shows how threats are mapped to STRIDE categories using the STRIDE definitions shown in Table 3-4 and the threat definitions of this research that were provided in Step 3.2. Each identified threat is mapped to STRIDE categories on the basis of the main effect of that threat in the first instance. Thus, the Spoofing/Impersonation threat is mapped to Spoofing in STRIDE. Since Eavesdropping, Traffic analysis, and MITM threats are passive attacks that observe and capture the transmitted data, those threats are mapped to Information Disclosure in STRIDE. In contrast, a Replay attack is active, in which the attacker observes the data then manipulates it and sends it back to the SG. Therefore, a Replay attack is mapped to Information disclosure and Tampering, in STRIDE. The Data tampering threat is mapped to Tampering in STRIDE, as they have the same meaning. Also, the DOS/Jamming channel is mapped to the Denial of service in STRIDE. Malware is mapped to Tampering in STRIDE, as it includes the execution of malicious software on the SG. Phishing tricks users into believing that a message is from a trustworthy source to obtain sensitive information that could be used to access the system. Because of that, phishing is mapped to Information disclosure in STRIDE as its main effect. Both SQL injection and False data injection are mapped to Tampering in STRIDE, as it means changing the transmitted data and could target the database. False data injection is mapped to Repudiation, as it could change the audit logs and transaction records, leading to denial of responsibility. The Elevation of Privilege category is assigned to threats where the attacker can gain unauthorized access or higher-level permissions than they should have. Spoofing, data tampering, and phishing all fit this description. Other categories are mapped based on the primary impact of the threat as described. When an attacker successfully impersonates a legitimate user or service, they can access resources and perform actions as the impersonated user, effectively gaining higher privileges than they originally had. Although Data tampering primarily classified under "Tampering with Data," if an attacker alters data that controls access permissions (like modifying user roles or access control lists), it can lead to unauthorized elevation of privileges. Phishing attacks often aim to steal credentials. If successful, the attacker can use these credentials to access higher privilege accounts, thereby elevating their privilege level. Replay attack, Data tampering, Malware injection, SQL injections, and False data injection are mapped to Repudiation because these threats involve scenarios where actions or changes could be denied later, especially if there are insufficient logs or auditing mechanisms to prove otherwise.

Table 3-5: Threat classification with STRIDE threat modelling

	STRIDE Threat Modelling					
Identified Threat	**Spoofing Identity**	**Tampering with Data**	**Repudia-tion**	**Information Disclosure**	**Denial of Service**	**Elevation of Privilege**
1. Spoofing/Impersonation	x					x
2. Eavesdropping/Traffic analysis/MITM				x		
3. Replay attack		x	x	x		
4. Data tampering		x	x			x
5. Denial of Service (DOS)/ Jamming channel					x	
6. Malware injection		x	x			
7. Phishing				x		x
8. SQL injections		x	x			
9. False data injection		x	x			

Table 3-6 shows how threats are mapped to security requirements on the basis of STRIDE mapping in both Table 3-4 and the best allocation in the literature (Wang and Lu, 2013; McCary and Xiao, 2015; Suleiman *et al.*, 2015; Tazi *et al.*, 2016; Mrabet *et al.*, 2018; Stellios *et al.*, 2018; Gunduz and Das, 2020; Tufail *et al.*, 2021). Each threat was assessed to identify the main security requirement that is compromised. In Spoofing, Authentication is affected because they involve an attacker pretending to be someone else, as mentioned in the STRIDE in Table 3-3. Also, Authorisation is impacted because once an attacker impersonates a legitimate user, they can gain access to resources that they are not authorised to access originally. Spoofing is also affects Non-Repudiation since actions taken by the attacker can be incorrectly attributed to the impersonated user. In Eavesdropping, in the first instance Confidentiality is affected because Eavesdropping involves unauthorised interception of data, leading to the disclosure of sensitive information. Privacy is also affected by Eavesdropping because intercepted data often contains personal or sensitive information that should remain confidential. Replay attack affects integrity because it involves the unauthorised re-transmission of valid data, leading to potential duplication and unauthorised actions. Also, it impacts Non-Repudiation because the attacker can deny performing the repeated actions if the system cannot distinguish between the original and the replayed actions. Data tampering directly affects integrity because it involves unauthorised alterations to data, compromising its accuracy. Looking at the desired property column in Table 3-4, the DOS/Jamming channel is mapped to Availability by making services or resources unavailable to legitimate users. Malware threat executes malicious software that could degrade the availability of the system as well as introducing malicious code that can alter data or system functions in unauthorised ways which threaten integrity. Also, Malware injection impacts Privacy as Malware can access, collect, or transmit personal or sensitive information. Phishing is mapped to Confidentiality, as it discloses information and tricks users into revealing personal or sensitive information, leading to privacy breaches. SQL injection compromises integrity by allowing attackers to execute unauthorised commands that can alter or delete data in a database. Also it impacts Non-repudiation since attackers can manipulate database records and deny their actions if proper logging is not in place. False data injection affects Integrity by introducing incorrect or misleading data into a system, compromising the accuracy of the data. Non-Repudiation is also affected by False data injection because actions based on false data can be disputed, and attackers can deny their involvement.

Table 3-6: Threat classification with security requirements

	Security Requirements						
Identified Threat	**Confidentiality**	**Integrity**	**Availability**	**Authentication**	**Authorisation**	**Privacy**	**Non-Repudiation**
1. Spoofing/Impersonation				x	x		x
2. Eavesdropping/Traffic analysis/MITM	x					x	
3. Replay attack		x					x
4. Data tampering		x					
5. Denial of Service (DOS)/ Jamming channel			x				
6. Malware injection		x	x			x	
7. Phishing	x					x	
8. SQL injections		x					x
9. False data injection		x					x

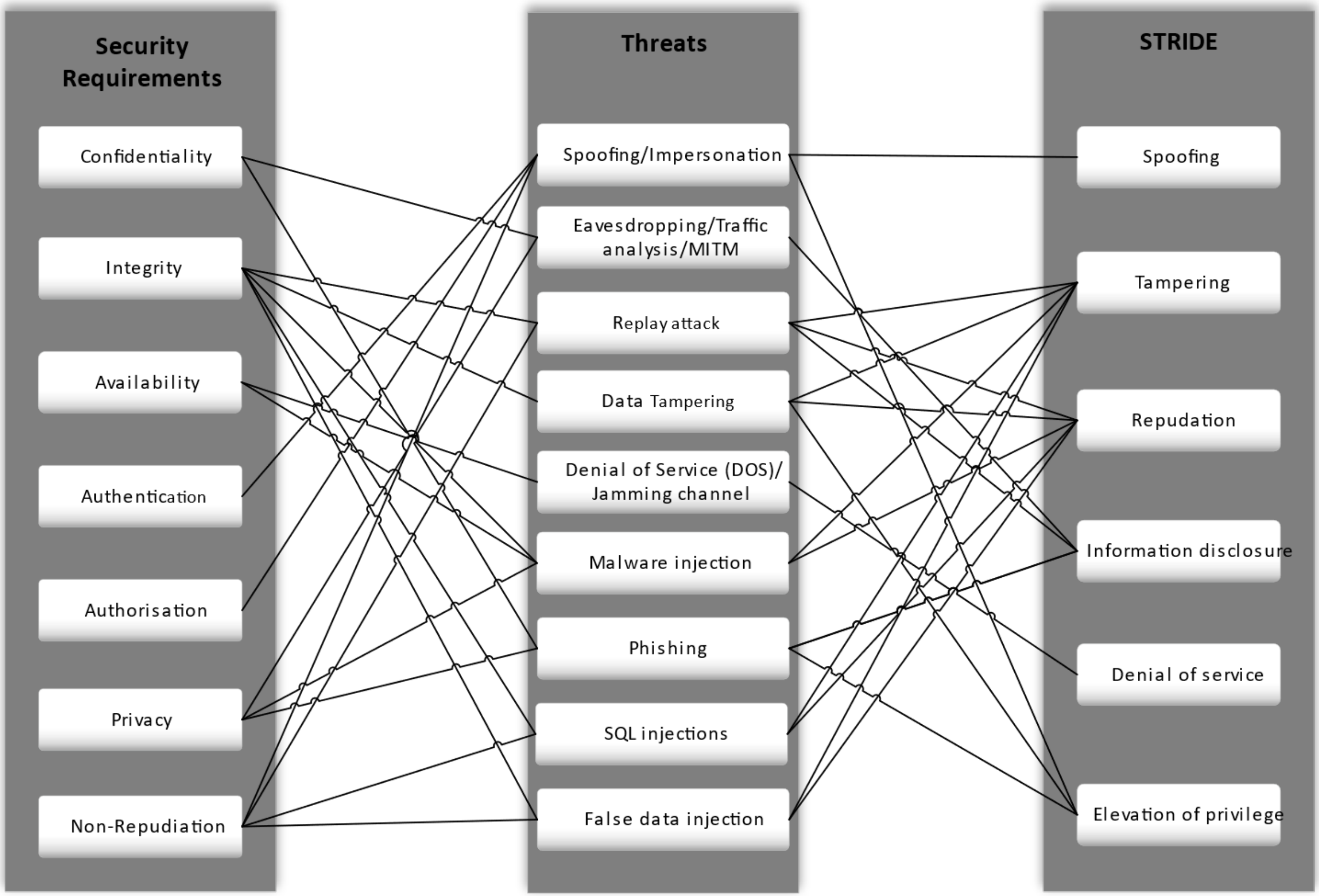


Figure 3-4: Threat modelling of the IoT-enabled Smart Grid

3.2.3 Step 3 Assign Threats to the Access Points

In order to assign the identified threats to the access points, this step analyses each access point according to its functionality, operations processed, and information systems located there, as discussed in Step 2.2. The threats that could be encountered when processing such operations are then considered, and a literature review is carried out to help in better mapping between threats and access points (McCary and Xiao, 2015; Tazi *et al.*, 2016; Mrabet *et al.*, 2018; Ganguly *et al.*, 2019; Díaz Redondo *et al.*, 2020). The STRIDE model is applied by considering how the threats in the model affect each access point, component, and interconnection (Microsoft, 2009). Essentially, looking at each access point and determining whether any threats that fall into the S, T, R, I, D, or E categories are posed for that access point (Microsoft, 2009).

Smart meters and smart appliances are devices that send readings yet cannot receive data, so False data injection does not apply to them. Moreover, no access points, including smart meters, distribution, transmission, generation, ICT, AMI, and SCADA, have a client-side interface, thus Phishing does not apply to those access points. Likewise, SQL injection attack does not apply to smart meters, distribution, transmission, generation, ICT, and SCADA, because they do not include databases. The resulting Figure 3-5 shows the threats at each access point. The details of the matrix tables for every access point are given in Appendix A (section C).

3.2.4 Step 4 Categorise Security Controls by Security Requirements

Security controls are countermeasures to mitigate, delay or prevent threats in order to strengthen the information system. The controls are approaches that fulfil security requirements. The security controls in Table 3-7 were derived from reviewing the literature and Microsoft documentation (2009) then categorised by security requirements, using the description of each security control as elaborated in Appendix E. This provided a foundational understanding of widely accepted security practices and controls.

To ensure a comprehensive mapping and alignment with best practices, various standards including NIST IR, NERC CIPS (1-9), NIST IR7628, and NIST SP 800-53, were reviewed to map the security controls to the security requirements. Additionally, a range of academic publications were consulted for mapping purposes, including works by (Hutchins *et al.*, 2011; Komninos *et al.*, 2014; McCary and Xiao, 2015; Tazi *et al.*, 2016; Mrabet *et al.*, 2018; Das and Zeadally, 2019; Ganguly *et al.*, 2019; Kimani *et al.*, 2019; Díaz Redondo *et al.*, 2020). This process involved identifying controls specified in the standards and categorizing them based on their primary security function (e.g., confidentiality, integrity, availability). A detailed descriptions and justifications for each control are provided in Appendix E.

The initial focus of this research was to list the chief potential access points that are vulnerable to the main internet-based threats and the relevant controls to mitigate against those threats. As implementation was not considered in the current phase, issues related to the performance and cost of the controls were not considered. The following Table 3-6 categorises the security controls across the security requirements as a result of a rigorous and systematic process involving literature review, standards mapping, and consultation of academic publications. This approach ensures that the identified controls are comprehensive, aligned with best practices, and effectively address the various security requirements necessary to protect information systems against the identified threats.

Table 3-7: Mapping security controls to security requirements

Security Requirement	Security Controls	Code
Authentication (Aun)	1. Keyed cryptographic hash functions (HMAC), digital signatures, and Random numbers generators	Aun1
	2. Physically Unclonable Functions (PUF)	Aun2
	3. MAC-attached, and HORS-signed messages	Aun3
	4. Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)	Aun4
	5. Multi-factor authentication mechanism	Aun5
	6. Automatic lockouts	Aun6
Authorisation (Aur)	7. Attribute-Based Encryption	Aur1
	8. Attribute Certificates	Aur2
	9. Attribute-Based Access Control System based on XACML (Extensible Access Control Markup Language)	Aur3
	10. Role-Based Access Control and allow/block listing	Aur4
Confidentiality (C)	11. Symmetric and asymmetric algorithms and Public Key Infrastructure certificate (PKI)	C1
Privacy (P)	12. Anonymisation	P1
	13. Trusted aggregators	P2
	14. Homomorphic encryption	P3
	15. Perturbation models	P4
	16. Verifiable computation models, and zero-knowledge proof systems	P5
	17. Data obfuscation techniques	P6
Integrity (In)	18. Cryptographic hashing functions and session keys	In1
	19. Digital watermarking	In2
	20. Automated patch management for flaw remediation	In3
	21. Adaptive cumulative sum algorithm	In4
	22. Secure Phasor Measurement Units (PMUs) installation	In5
	23. Load profiling algorithms	In6

	24. Timestamps	In7
	25. Sequence numbers	In8
	26. Query sanitisation	In9
	27. Nonces	In10
Availability (Av)	28. Use multiple alternate frequency channels according to a hardcoded sequence	Av1
	29. Frequency quorum rendezvous between connected nodes	Av2
	30. Anomaly Intrusion Detection Systems (IDS)	Av3
	31. Specification-based IDS	Av4
	32. Intrusion Prevention Systems (IPS)	Av5
	33. Quality of Services (QoS)	Av6
	34. Load balancing	Av7
	35. Operating system-independent applications	Av8
Non-repudiation (N)	36. Mutual Inspection technique	N1
	37. Unique keys and digital signatures	N2
	38. Transaction log	N3

3.2.5 Step 5. Map the Security Controls to the Access Points

In step 5, each access point was assigned a set of security requirements that, after assessing threats' effects, could be countered by applying appropriate security controls. For every access point, in the first instance each threat was mapped to the security requirements that might be compromised by an attack, as in Table 3-6. Then, controls were allocated to the relevant security requirements according to Table 3-7. Only relevant controls that apply to the corresponding access point were assigned, on the basis of the specifications and functionality of that access point. In Authentication, for example, cryptographic hash functions and MAC/HORS-signed messages were not applied to the Transmission and Distribution stations as they are machinery, thus are not designed for this type of data processing. Therefore, the controls (Aun1) and (Aun3) were not assigned to the Transmission and Distribution stations, yet SSL/TLS Certification (Aun4) and Physically Unclonable Functions (PUF) (Aun2) were. In this case, the controls for the Transmission and Distribution stations were mapped as shown in the sample matrix in Table 3-8.

Table 3-8:Sample matrix for the Transmission and Distribution stations access point

Access Point	**Internet-based Security Threats from the STRIDE Analysis**	**Security Requirement**	**Security Control**
2. Transmission, Distribution Stations	Spoofing	Authentication	Aun2: Physically Unclonable Functions (PUF) Aun4: Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)
		Authorisation	Aur2: Attribute Certificates Aur3: Attribute-Based Access Control System based on XACML Aur4: Role-Based Access Control and allow listing

Authorisation is also compromised in the event of Spoofing. Thus, the controls list that supports Authorisation, as in Table 3-8, was assessed to determine the relevant controls that could be applied to the Transmission and Distribution stations. Attribute-Based Encryption (Aur1) was not assigned; since these stations are not designed to perform encryption, yet (Aur2), (Aur3), and (Aur4) were.

The full mapping matrices for each access point are in Appendix A (section C). The mapping process was validated by expert review, as in Chapter 5, and the confirmed matrices are available in Appendix B.

3.3 The IoT-Enabled Smart Grid Information Security Model

The process developed in sections 3.2 and 3.2 was used to construct the Information Security Model for the IoT-enabled Smart Grid in Figure 3-5.

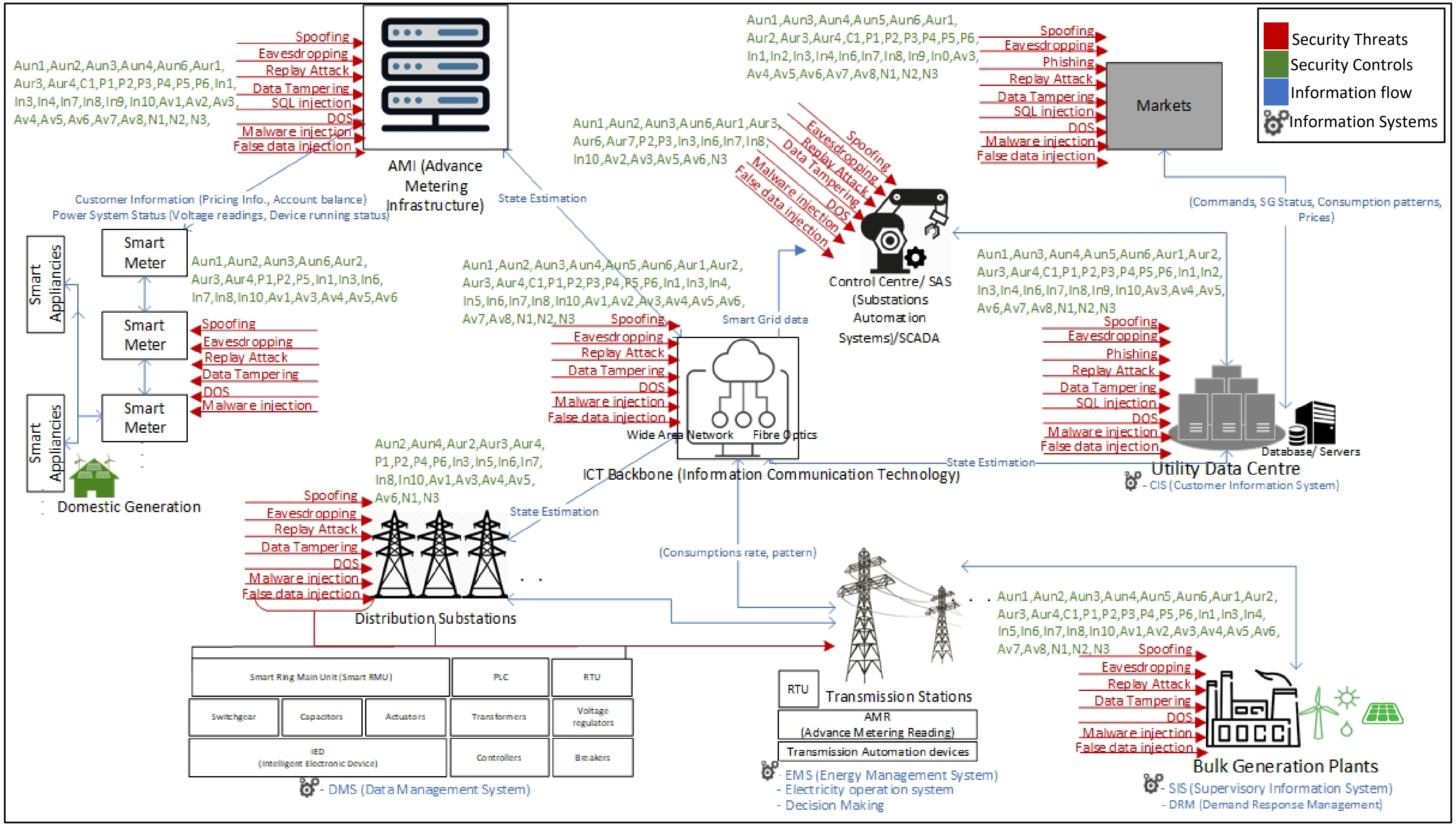

Security Threats
Security Controls
Information flow
Information Systems
Aun1,Aun2,Aun3,Aun4,Aun6,Aur1,
Aur3,Aur4,C1,P1,P2,P3,P4,P5,P6,In1,
In3,In4,In7,In8,In9,In10,Av1,Av2,Av3,
Av4,Av5,Av6,Av7,Av8,N1,N2,N3,
Spoofing
Eavesdropping
Replay Attack
Data Tampering
SQL injection
DOS
Malware injection
False data injection
AMI (Advance Metering Infrastructure)
Customer Information (Pricing Info., Account balance)
Power System Status (Voltage readings, Device running status)
State Estimation
Aun1,Aun3,Aun4,Aun5,Aun6,Aur1,
Aur2,Aur3,Aur4,C1,P1,P2,P3,P4,P5,P6,
In1,In2,In3,In4,In6,In7,In8,In9,In0,Av3,
Av4,Av5,Av6,Av7,Av8,N1,N2,N3
Spoofing
Eavesdropping
Phishing
Replay Attack
Data Tampering
SQL injection
DOS
Malware injection
False data injection
Markets
Aun1,Aun2,Aun3,Aun6,Aur1,Aur3,
Aur6,Aur7,P2,P3,In3,In6,In7,In8,
In10,Av2,Av3,Av5,Av6,N3
Spoofing
Eavesdropping
Replay Attack
Data Tampering
DOS
Malware injection
False data injection
Control Centre/ SAS (Substations Automation Systems)/SCADA
(Commands, SG Status, Consumption patterns, Prices)
Smart Meter
Smart Meter
Smart Meter
Smart Appliancies
Smart Appliancies
Aun1,Aun2,Aun3,Aun6,Aur2,
Aur3,Aur4,P1,P2,P5,In1,In3,In6,
In7,In8,In10,Av1,Av3,Av4,Av5,Av6
Spoofing
Eavesdropping
Replay Attack
Data Tampering
DOS
Malware injection
Domestic Generation
Aun1,Aun2,Aun3,Aun4,Aun5,Aun6,Aur1,Aur2,
Aur3,Aur4,C1,P1,P2,P3,P4,P5,P6,In1,In3,In4,
In5,In6,In7,In8,In10,Av1,Av2,Av3,Av4,Av5,Av6,
Av7,Av8,N1,N2,N3
Spoofing
Eavesdropping
Replay Attack
Data Tampering
DOS
Malware injection
False data injection
Smart Grid data
Wide Area Network
Fibre Optics
ICT Backbone (Information Communication Technology)
Aun1,Aun3,Aun4,Aun5,Aun6,Aur1,Aur2,
Aur3,Aur4,C1,P1,P2,P3,P4,P5,P6,In1,In2,
In3,In4,In6,In7,In8,In9,In10,Av3,Av4,Av5,
Av6,Av7,Av8,N1,N2,N3
Spoofing
Eavesdropping
Phishing
Replay Attack
Data Tampering
SQL injection
DOS
Malware injection
False data injection
State Estimation
Database/ Servers
Utility Data Centre
- CIS (Customer Information System)
Aun2,Aun4,Aur2,Aur3,Aur4,
P1,P2,P4,P6,In3,In5,In6,In7,
In8,In10,Av1,Av3,Av4,Av5,
Av6,N1,N3
Spoofing
Eavesdropping
Replay Attack
Data Tampering
DOS
Malware injection
False data injection
State Estimation
Distribution Substations
Smart Ring Main Unit (Smart RMU)
PLC
RTU
Switchgear
Capacitors
Actuators
Transformers
Voltage regulators
IED (Intelligent Electronic Device)
Controllers
Breakers
- DMS (Data Management System)
(Consumptions rate, pattern)
RTU
Transmission Stations
AMR (Advance Metering Reading)
Transmission Automation devices
- EMS (Energy Management System)
- Electricity operation system
- Decision Making
Aun1,Aun2,Aun3,Aun4,Aun5,Aun6,Aur1,Aur2,
Aur3,Aur4,C1,P1,P2,P3,P4,P5,P6,In1,In3,In4,
In5,In6,In7,In8,In10,Av1,Av2,Av3,Av4,Av5,Av6,
Av7,Av8,N1,N2,N3
Spoofing
Eavesdropping
Replay Attack
Data Tampering
DOS
Malware injection
False data injection
Bulk Generation Plants
- SIS (Supervisory Information System)
- DRM (Demand Response Management)

Figure 3-5: IoT-enabled Smart Grid Information Security Model

3.4 Summary

This chapter identified the requirements, threats, and controls that contribute to the information security of the IoT-enabled SG. First, the security requirements were explored and identified by reviewing the relevant literature and the international standards. The security requirements in the IoT-enabled SG are confidentiality, integrity, availability, authentication, authorisation, privacy, and non-repudiation. The NIST reference model was used to characterise the system. Then, nine internet-based threats were identified using the STRIDE technique for threat modelling.

Next, the threats and security requirements were mapped according to the literature, international standards, and STRIDE analysis. In addition, the security controls were mapped to the controls.

The conclusion of this part is that information security consists of seven security requirements, nine threats, seven access points, and 38 security controls. This addresses the limitations of the NIST model, which is a high-level conceptual model lacking the detail that makes this research's Information Security Model so practical and useful to related sectors.

The next chapter outlines the research methodology adopted to confirm this model. Then, Chapter 5 investigates this model and all the identified access points, security requirements, threats, and controls in order to confirm the model.

Chapter 4 **Research Methodology**

Chapter 3 introduced the Information Security Model developed by this research to enrich the NIST model with useful detail, including security requirements, internet-based threats, access points, and security controls. This chapter outlines the research methods adopted in confirming the model in Figure 3-5. The first section briefly discusses the methods, both qualitative and quantitative, and mixed methods, with an explanation of the triangulation technique applied to confirm the model. The subsequent section explains in detail the design of research methods applied in this research and includes interview designs, data collection processes and instruments, the piloting, sample size, ethical approval, and analysis.

4.1 **Qualitative Methods**

Qualitative methods are usually employed to analyse and explain non-numeric data to clarify and understand certain phenomena. In addition, such methods help scholars to investigate and question areas of research and to explore new important variables. This includes gathering and analysing data, and interpreting those data to understand the situation or the field, for example the experiences, values, and behaviour of individuals (Creswell and Creswell, 2017). The techniques include observations, interviews, discussions, and documents, all of which make the gathered data rich and holistic (Creswell and Creswell, 2017).

4.2 **Quantitative Methods**

Quantitative methods are commonly used to identify the factors that influence the results, intervene in the feasibility of the results, or detect the predictions of the results. They are usually used when factors that impact the results need to be identified, the feasibility of an intervention must be determined, or the results are to be predicted. In addition, they are used to collect, analyse, and interpret numeric data produced by surveys or questionnaires, and help to define certain phenomena (Creswell and Creswell, 2017).

4.3 **Triangulation**

Triangulation is a research technique that is defined as a combination of two or more methods in a study (Thurmond, 2001). The technique can be applied to explore the issues extensively and improve the accuracy of research findings (Fink, 2003). The idea is that if multiple sources of information or data produce similar results, the credibility of the findings is enhanced (Fink, 2003). The findings from each method can be compared to find similar conclusions (Guion, 2002).

In this research, methodological triangulation was applied. Data were collected from two methods, quantitative and qualitative. Subsequently, the results were compared to identify similar decision patterns (Golafshani, 2003). This ensured that the process of refining and verifying the security model was thoroughly explored. It involved a synthesis of assessing, combining, and comparing data discovered from analysis, a detailed literature review, expert reviews, and the use of formal methods at the next stage of the research, as shown in Figure 4-2.

4.4 Methods Applied in This Research

Both quantitative and qualitative approaches were used because this research is based on mixed methods (Creswell and Creswell, 2017). In order to achieve the objectives of the research specified in Chapter 1, section 1.2, a methodical triangulation research technique was adopted (Cohen *et al.*, 2013). This helped to create a comprehensive picture of the research topic and increase the possibility of validating and confirming the results. The research technique, as shown in Figure 4-1, comprised three methods: analysis and literature review, expert interviews, and formal methods.

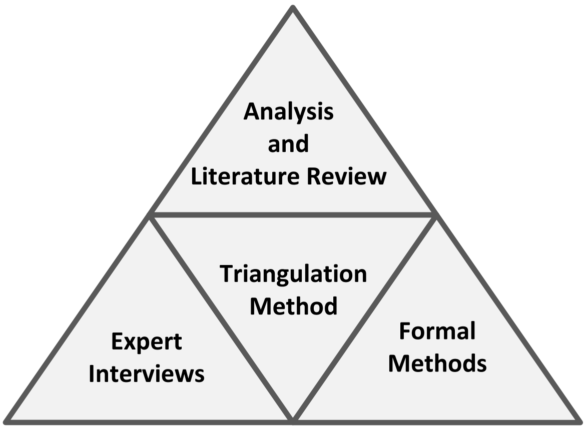


Figure 4-1: Research methods

In the next stages of the research, the methodical triangulation research technique was conducted in a sequential approach consisting of four phases, as in Figure 4-2. As soon as the interviews with the experts were complete the feedback was analysed, following the structure of the interview design. The second version of the model was then constructed. Following the findings of the expert review, the model was confirmed. Then, the functional requirements were developed in preparation for validating and verifying the model for the study's third progression report, taking a formal method approach. Then, the Abstract formal model was constructed for a segment of the IoT-enabled SG. Afterwards, the abstract model was augmented and the first refinement was developed to include more detailed functionality, for a more accurate and consistent formal model. Next, the second refinement was developed to formulate the security controls and mitigations, resulting in the verified and valid security model for the IoT-enabled Smart Grid.

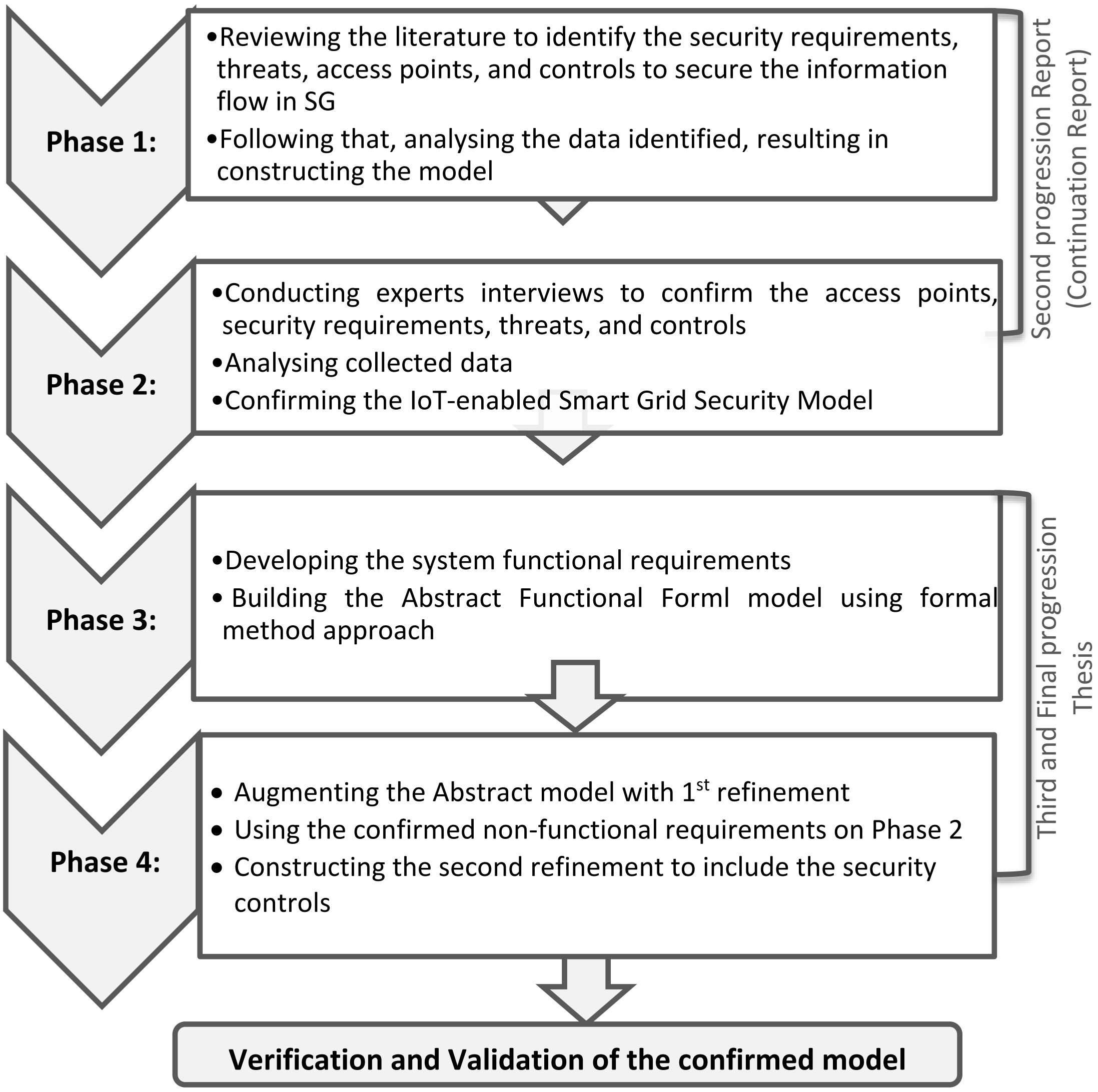

Phase 1:
•Reviewing the literature to identify the security requirements, threats, access points, and controls to secure the information flow in SG
•Following that, analysing the data identified, resulting in constructing the model
Phase 2:
•Conducting experts interviews to confirm the access points, security requirements, threats, and controls
•Analysing collected data
•Confirming the IoT-enabled Smart Grid Security Model
Phase 3:
•Developing the system functional requirements
• Building the Abstract Functional Forml model using formal method approach
Phase 4:
• Augmenting the Abstract model with 1st refinement
• Using the confirmed non-functional requirements on Phase 2
• Constructing the second refinement to include the security controls
Second progression Report (Continuation Report)
Third and Final progression Thesis
Verification and Validation of the confirmed model

Figure 4-2: Research methodology phases

Figure 4-3 showing the inegration of quantitative and qualitative methods highlighting the interaction of the research phases including the input and output of each methodological phase. The mixed triangular medthod involved using a qualitative approach initially in the form of analysis, literature review, and semi-structured interviews with experts, asking open-ended questions in order to confirm the IoT-enabled SG information security model and its access points, security requirements, threats, and controls. Afterward, a quantitative approach is integrated in the form of formal methods. In which the model was verified and validated using the Event-B formal modelling approach, which includes developing an abstract formal model to capture the system scenario and functional requirements. Augmentation stages captured further both functional and non-functional requirements, to include the security requirements. Figure 4-4 shows the research methodology process.

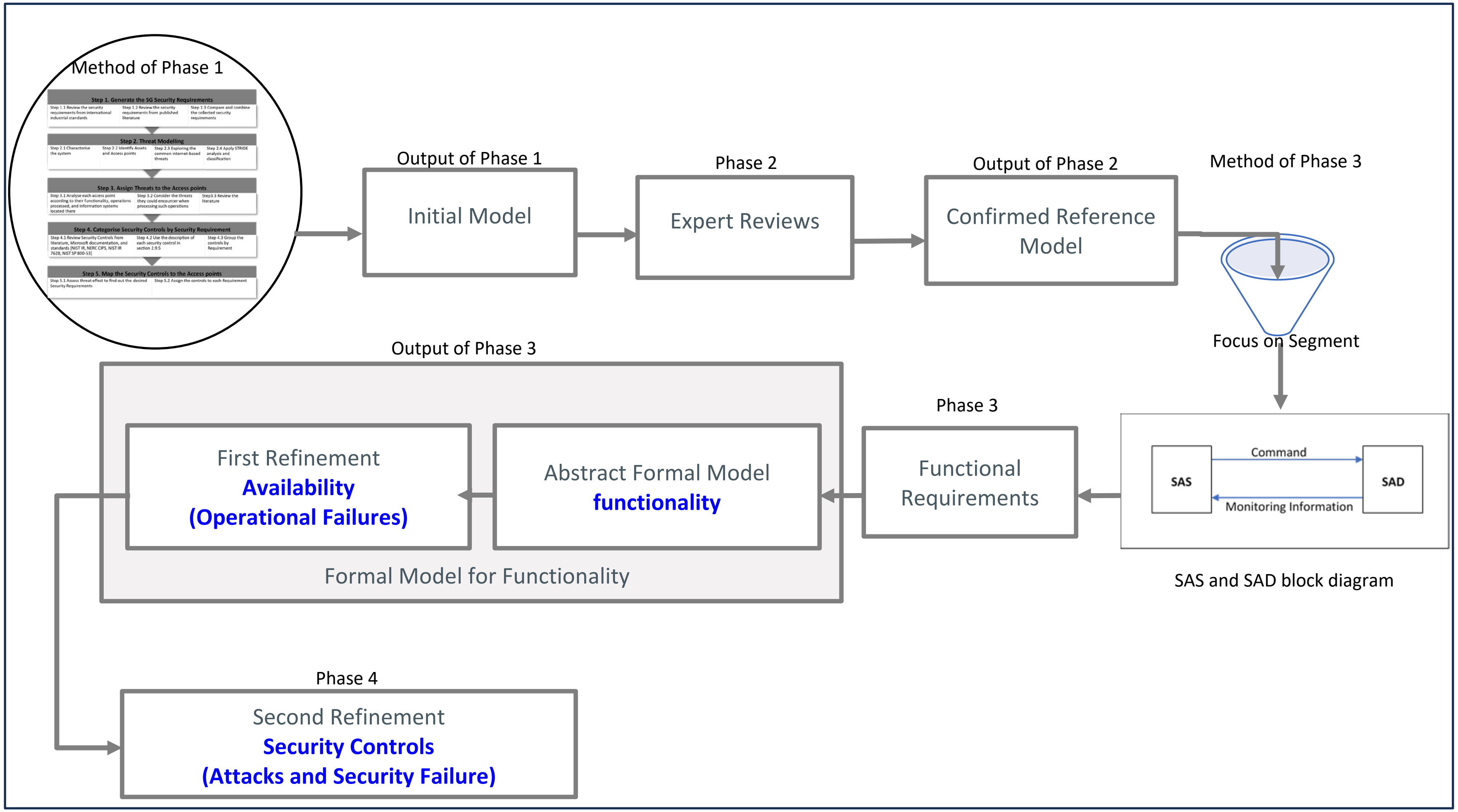

Method of Phase 1
Step 1. Generate the SG Security Requirements
Step 1.1 Review the security requirements from international industrial standards
Step 1.2 Review the security requirements from published literature
Step 1.3 Compare and combine the collected security requirements
Step 2. Threat Modelling
Step 2.1 Characterise the system
Step 2.2 Identify Assets and Access points
Step 2.3 Exploring the common internet-based threats
Step 2.4 Apply STRIDE analysis and classification
Step 3. Assign Threats to the Access points
Step 3.1 Analyse each access point according to their functionality, operations processed, and information systems located there
Step 3.2 Consider the threats they could encounter when processing such operations
Step3.3 Review the literature
Step 4. Categorise Security Controls by Security Requirement
Step 4.1 Review Security Controls from literature, Microsoft documentation, and standards [NIST IR, NERC CIPS, NIST IR 7628, NIST SP 800-53]
Step 4.2 Use the description of each security control in section 2.9.5
Step 4.3 Group the controls by Requirement
Step 5. Map the Security Controls to the Access points
Step 5.1 Assess threat effect to find out the desired Security Requirements
Step 5.2 Assign the controls to each Requirement
Output of Phase 1
Initial Model
Phase 2
Expert Reviews
Output of Phase 2
Confirmed Reference Model
Method of Phase 3
Focus on Segment
SAS
Command
SAD
Monitoring Information
SAS and SAD block diagram
Phase 3
Functional Requirements
Output of Phase 3
Abstract Formal Model
functionality
First Refinement
Availability
(Operational Failures)
Formal Model for Functionality
Phase 4
Second Refinement
Security Controls
(Attacks and Security Failure)

Figure 4-3: The inegration of quantitative and qualitative methods showing the interaction, input, and output of each methodology phase

4.5 Expert Reviews

Using qualitative research techniques, interviews were designed for experts in information security and SG to consult them on the model and to confirm the access points, the main and common security threats under each point, and the security requirements and controls to secure the information flow in SG.

In all, 14 experts were interviewed. All were experts in either SG, electricity or energy security or were electric engineering academics, or a combination. The interviews covered the Saudi Electricity Company (SEC) in specifically KSA. Interviewees' roles were engineers, developers, manufacturers, contractors, and analysts from both industrial and academic sectors, in various electric domains including cybersecurity, distribution, transmission, generation, and Information Technology (IT), as in Table 4-1. The criteria of selecting the interviewees are their years of experience in their domain, as well as, their contribution to the field for the academic experts as in Table 4-1. These interviews took place in the KSA, from where all the experts came.

Each interview was carried out in phases. Emails were sent to the experts, inviting them to participate and asking for their preferred place, time, and channel, whether online, via zoom call, or through face-to-face interview, depending on their availability and location. A guide to the topic and goals of the research were briefly explained in the email.

Table 4-1: Summary of interviewees

Expert	Job Description	Field Type	Domain	Years of Experience
1	Electrical Engineering Associate Professor	Academic	Electrical engineering	12
2	Electrical & Computer Engineering Assistant Professor	Academic	Electrical engineering	14
3	IT security engineer at the electricity company, with risk management experience	Industry	IT security in electricity systems	22
4	Cybersecurity engineer in the distribution department of the electricity company	Industry	Cybersecurity, distribution	6
5	Cybersecurity engineer and risk management expert in Governance Risk Compliance (GRC) sector at the electricity company. Certified GICSP Global Industrial Cybersecurity professional.	Industry	Cybersecurity and architect, generation, transmission, and distribution	7

6	Cybersecurity analyst at the electricity company	Industry	Cybersecurity, Distribution	7
7	Cybersecurity engineer at servers' management	Industry	IT security	15
8	Cybersecurity Engineer at the oil company, US Smart grid engineer previously, cyber security algorithms developer, and cyber security mentor for 40 trainees	Industry	Cybersecurity in electricity systems	30
9	Electrical engineer in the contractor company for smart meters project	Industry	Electricity contractor	10
10	Smart meters developer, designer and developer SW/HW firmware, experience in security standards	Industry	Smart meter manufacturing	8
11	Electrical engineer in an oil company, advanced smart meters project manager, committee secretary of a renewable energy association, a member of Standards, Metrology and Quality Organisation	Government	Electrical engineering	8
12	Telecommunication devices engineer in the transmission department of the electricity company	Industry	Transmission (National Grid), Communication	7
13	Electrical engineer at French electricity company, experience in smart meters project in KSA, experience in renewable energy	Industry	Electrical engineering, energy value chain including generation, transmission, distribution	10
14	Cybersecurity consultant at the electricity company	Industry	Cybersecurity, distribution	12

4.6 Interview Design and Data Collection

Semi-structured interviews were adopted, characterised by open-ended questions presented in the survey in Appendix A, taking into consideration the primary objectives of the interview:

- To review and confirm the access points, security requirements, threats, and controls that secure the SG information flow, identified by threat analysis and literature review
- To identify any other access points, security requirements, threats, and controls that have not been included in the model or to modify the current ones.

Before the beginning of the interview each participant was asked to review the participant information sheet and was encouraged to sign the consent form. After they were shown the diagram presenting the access points, open-ended questions were asked that related to the ways in which the model could be improved. These questions were designed to confirm each access point, security requirement, threat, and control. They also were to elicit those characteristics of the IoT-Enabled SG model that could be relocated, deleted, or added to. The questions were devided into three main sections A, B, and C in Appendix A. Section A, the experts were given a diagram of potential access points and invited to suggest changes in order to confirm the main access points. Under each access point are several security threats identified from the STRIDE analysis undertaken in section 3.2.2. The access points of attacks were highlighted by red arrows. In section B, the experts were given a table containing the controls mapped to each security requirement to confirm the security controls and requirements in a general context. Then, in section C, the experts were asked to confirm the mapping between the controls to each specific access point against a specific potential set of threats. In total, this stage lasted from 1 to 7 hours. Some interviews were conducted over three days.

All interviews were conducted online using Zoom software and were recorded using both the Zoom recording feature and a notebook. Before the beginning of any recording, permission was obtained from each interviewee. The consent form and the information sheet had been sent via email to the participants.

4.7 Ethical Approval for Experts Interviews

Before conducting any interview it was essential to obtain ethical approval from the Ethics Committee at the University of Southampton. The ethical approval was achieved on 17/12/2020 with ethics approval reference number 62423.

While the data were being collected, a consent form was given to the interviewees. This contained comprehensive participant information, and each interviewee were invited to sign the form as an agreement to take part in the research. The participants received a digital consent form via email that had to be filled out before the interview could take place.

4.8 Pilot Interview

To test the validity of the instruments used in this research, piloting of the interview was conducted before the actual interviews were held. A pilot aims to test and ensure the accuracy of the instrument before it is used in a real-life situation (Fink, 2003). In this trial, four computer science researchers at the University of Southampton provided insightful feedback and beneficial recommendations to edit some complicated sentences and instructions, which improved the clarity of the question design for better understanding by the reader. The modifications they recommended were not fundamental.

4.9 Interview Sample Size

In qualitative research the focus is usually on non-probability sampling, whereby the respondents are chosen on the basis of specific criteria (Bhattacherjee, 2012). In expert sampling, respondents are selected on the basis of their knowledge of the topic that is being investigated (Bhattacherjee, 2012). It is essential to calculate the minimum sample needed; in this way, reliable results can be acquired (Banerjee *et al.*, 2009). Therefore, to produce significant results, an adequate number of participants need to be interviewed. When conducting a confirmation study, research has not produced any consensus about how many experts should be interviewed; most publications recommend between three and 20 (Lynn, 1986; Grant and Davis, 1997). Moreover, it has been suggested that the saturation method be used (Marshall *et al.*, 2013). Saturation is achieved when no new data can be produced (Bowen, 2008). (Guest *et al.*, 2006) suggest that the saturation stage can normally be reached after about 12 respondents have been interviewed. The present research, therefore, held interviews with 14 experts in the fields of IoT-enabled SG, electricity, and cybersecurity.

4.10 Methodology to Develop the Security Formal Model of the IoT-Enabled Smart Grid – Verification and Validation Stage

The modelling process gradually built the system according to the specification. The abstract model began the process by modelling the system's functional requirements and behaviour in each of the access points, and represents the information flow used to control and monitor the IoT-enabled SG. The overall process went as follows:

1. Building the Abstract formal model for the IoT-enabled SG, capturing the functional requirements. The abstract model was 'insecure' as it represented only the information flows, with no security controls.
2. Augmentation of the model. In this first augmentation, the generation of further functionality requirements was formulated for a more accurate and better description and understanding of the targeted system. A horizontal superposition refinement was applied in steps. Therefore, both the abstract model and the first augmentation developed the functional formal model that represents the system specification.
3. Further augmentation: at this stage the security controls were included, and these capture the non-functional requirements.
4. The augmentations resulted in building the formal ideal secure model for internet-based threats to information flow.

This took the standard approach to formal modelling, tackling the complexity of the system by incorporating various aspects at different levels of abstraction and refinement/augmentation. Formal modelling started with the functional requirements of the system. Based on the threat modelling carried out in Chapter 3, this functionality is prone to specific internet-based cyber threats behaviour affecting the system. Thus, this research modelled the security controls and mitigations. To prevent such unwanted behaviour, the security controls were integrated to protect the system from outside harm affecting the existing functionality. Harm is defined in this research as internet-based cyber threats. This research secures the information flowing between the communicating access points in the IoT-enabled SG system.

4.11 **Summary**

This chapter focused on the research methods employed and the reasons for choosing them. In order to confirm, validate, and verify the IoT-enabled SG model for securing exchanged information throughout the IoT-enabled SG, it was concluded that a mixed-method approach would be the most effective. This involved using a qualitative approach initially, then a quantitative approach in the form of formal methods. To fully examine the findings and to increase their level of accuracy and reliability, a triangulation technique was selected that proved to be most useful.

Semi-structured interviews with experts were conducted, asking open-ended questions. The reasoning behind the interview process was to confirm the IoT-enabled SG information security model and its access points, security requirements, threats, and controls. After confirmation of the model by domain experts, the model was verified and validated using the Event-B formal modelling approach, which includes developing an abstract formal model to capture the system scenario and functional requirements. Augmentation stages captured further both functional and non-functional requirements, to include the security requirements. Figure 4-4 shows the research methodology process.

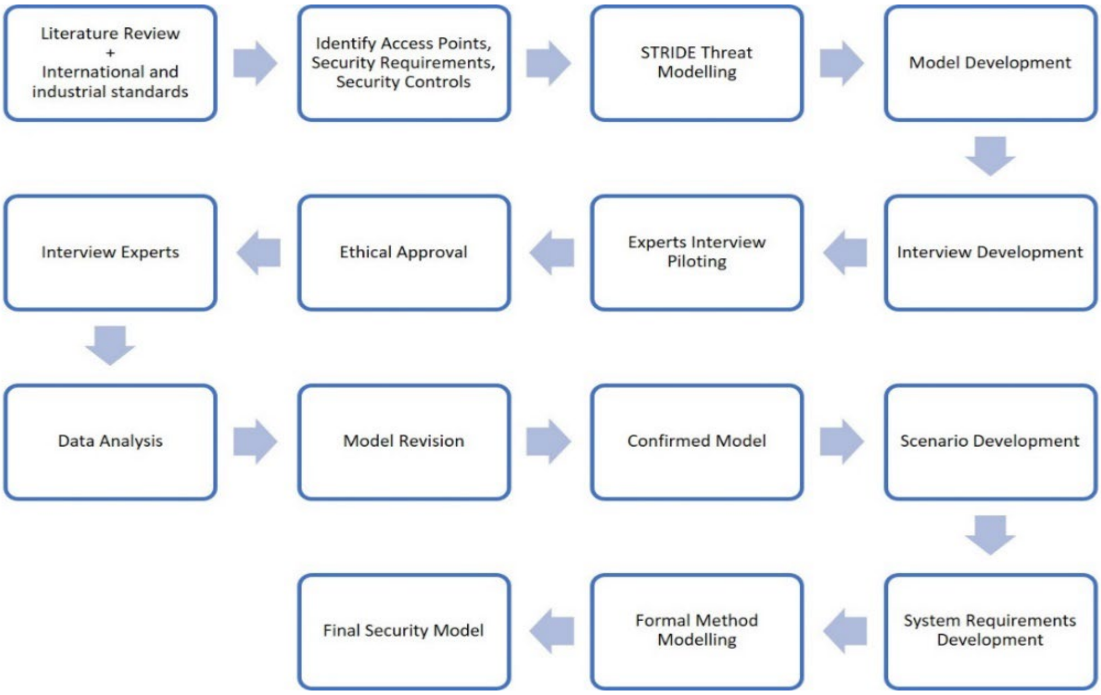


Figure 4-4: Research methodology process

Chapter 5 **Experts' Review Findings**

This chapter discusses the findings of the interviews conducted with experts to examine and confirm the IoT-enabled SG information security model and describe how the experts refined it. The semi-structured interviews were conducted in three sections with 14 experts in electricity and energy-related sectors in KSA. Each transcript was analysed to make sure that no new data were added to the findings. Saturation was reached at 14 experts.

The transcripts were analysed to find out whether the experts agreed or disagreed with each section and whether they had anything to change, add, or remove. The findings were grouped according to the design and divided into three sections. In section A, the experts were given a diagram of potential access points and invited to suggest changes in order to confirm the access points. In section B, the experts were given a table containing the controls mapped to each security requirement to confirm the security controls and requirements in a general context. In section C, the experts were asked to confirm the mapping between the controls to each specific access point against a specific potential set of threats. Analysis was applied to the findings, supported with relevant transcripts from the experts. Whenever certain experts are not referred to or mentioned, it indicates that they either did not respond or chose not to provide comments due to their lack of confidence in addressing topics outside their specific areas of expertise. At the end of each section, the findings were discussed and analysed, and the model was amended according to relevant changes. Feedback loop and validation is carried out as the changes made on the model were then re-evaluated by the same experts in subsequent rounds, allowing them to review how their input and the conflicting views were integrated. all expert opinions were documented, and discrepancies in viewpoints were flagged for further analysis. These discrepancies is addressed by subsequest calls for more clarification from the experts to reach a more refined consensus. Moreover, some conflicts were resolved by assigning weights to different opinions based on the expert's experience and the evidence supporting their views including the standards and the organisation's playbooks. These conflicts enrich the final model as it demonstrates the adaptability of the research to incorporate a range of expert insights, thereby enhancing the validity and applicability of the final model. By the end of the interviews the research assumption was verified, and the issues raised had been clarified and addressed.

It is important to note that the qualitative analysis conducted in this chapter complements the quantitative data presented in Chapter 6 and Chapter 7 of this research. This mixed-methods approach ensures a well-rounded analysis, where qualitative insights help interpret and give meaning to the quantitative findings. To enhance the rigor of the qualitative analysis in this chapter, this research employed multiple techniques such as thematic analysis and cross-validation with existing literature.

These methods ensure that the qualitative data are systematically analysed and robust, supporting the credibility and reliability of the insights derived.

The model was presented to the His Excellency the President of the Atomic & Renewable Energy City, a KSA government scientific research entity that legally contributes to the sustainable development of KSA's energy sector. His Excellency the President was supportive of the research and enabled the expert interviews that took place in the related sectors.

5.1 Review of the Model

All 14 experts stressed the importance of the information security model and the importance of securing such a critical electricity infrastructure in the country. They confirmed the contribution that the suggested model could make to the field as a useful model that supports KSA initiatives toward securing automated SGs. The following statement is by **Expert2** "*This model is excellent and could have a strong contribution to the field*." **Expert3** commented that "*Cybersecurity is a major concern for all countries and critical in any country wishing to secure the electricity infrastructure*."

Expert2, **Expert3**, **Expert7**, **Expert10**, and **Expert11** stated that "*It is rare to find experts in Smart Grid security, and the local electric companies are just now implementing the security solutions and controls without discussing any details in depth. The manufacturing companies and contractors have the best experts, who may have the required knowledge*." **Expert3** stated that "*security is a major concern for all electricity companies that looking after smart grids. The general understanding of employees in terms of cybersecurity and consequences lags behind, and many employees are still training to acquire the required experience as the Smart Grid and Smart Meters are rolled out more widely in many regions and countries*." Therefore, this research is important to fill this gap in cybersecurity knowledge. The concept of Smart Grids is a developing field, and any research in this area would be a useful contribution. This is, in turn, confirms that the IoT-Enables Smart Grid information security model might be feasible and useful to electricity companies.

Expert3, **Expert8**, **Expert9**, **Expert11**, and **Expert12** assured that "*Smart Grid cybersecurity is a recent topic introduced in many countries including KSA. Also, Smart Meters are also recently introduced in KSA and they are prone to being attacked*" **Expert3** and **Expert12** also said that "*Cybersecurity in Smart Grids were recently introduced in KSA*."

Expert9 pointed out that "*Nowadays, many organisations could be potential targets for the cybersecurity attacker because of digitisation, and this would increase the possibility of cyber-attacks*." **Expert3**, **Expert4**, **Expert5**, **Expert8**, and **Expert9** raised the point that "*In the past, the opportunity for communication was non-existent or very limited, using special-purpose communication protocols, but* *now in IoT-enabled Smart grids, the communication is using standard protocols which are vulnerable to cyber-attacks*." **Expert11** stated that "*the smart automation and smart substation could reduce the cost, fault, and number of employees taking meter readings, but it includes a high cybersecurity risk*." **Expert12** said that "*SAS Substation Automation System is an IoT-enabled infrastructure in which every component is smart automated device, and the control centre is controlling those devices. Subsequently, securing Smart Grid from threats has become crucial*." Communication in Smart Grids is important for monitoring and controlling the power production and distribution and then making decisions. On the other hand, communication increases security vulnerabilities. Therefore, it is crucial to secure these communications, and this model and the structural approach that this research is trying to tackle is becoming very significant. All previous points highlight the importance of the current research topic.

Expert4 acknowledged that the remit of this research might go beyond the expertise and the knowledge of any individual engineer, because each team is responsible for part of the network. They said that "*this research covered all expected aspects around cybersecurity controls that we are aware of in the field*." Both **Expert1** and **Expert2** mentioned that the security controls in this research are well-known to academics and researchers, but may be unfamiliar to experts working in the electricity sector. Both experts upheld the view that this research is comprehensive, reflecting possible solutions from the literature. **Expert1** indicated the high reputation of the source of these security controls by stating that "*as long as all these requirements and controls came from academic papers and literature, you can depend on and confirm them all. Especially from references that are trusted such as IEEE, NIST, and similar authorities. Academic papers are much more advanced than the current field practice. You can take the experts' views as a consultation only, or in the case of adding any new control, for example*." Consequently, all changes and recommendations were carefully reviewed and thought through before the model was changed.

5.2 Section A: Access Points

The experts were shown Figure 3-3 of the main access points of the IoT-enabled SGs envisaged during the earlier part of this research. Under each access point are several security threats identified from the STRIDE analysis undertaken in section 3.2.2. The access points of attacks were highlighted by red arrows, as in Figure 3-3. The diagram was redrawn to emphasise the information flow through each access point under investigation. Since some experts were confused about the physical flow of electricity, not the abstract concept, so the diagram was changed as shown in Figure 5-1 to help in the following:

- Make the diagram more comprehensible to the experts in electrical engineering
- Demonstrate the segregation mechanism applied to SGs
- Emphasise the information flow.

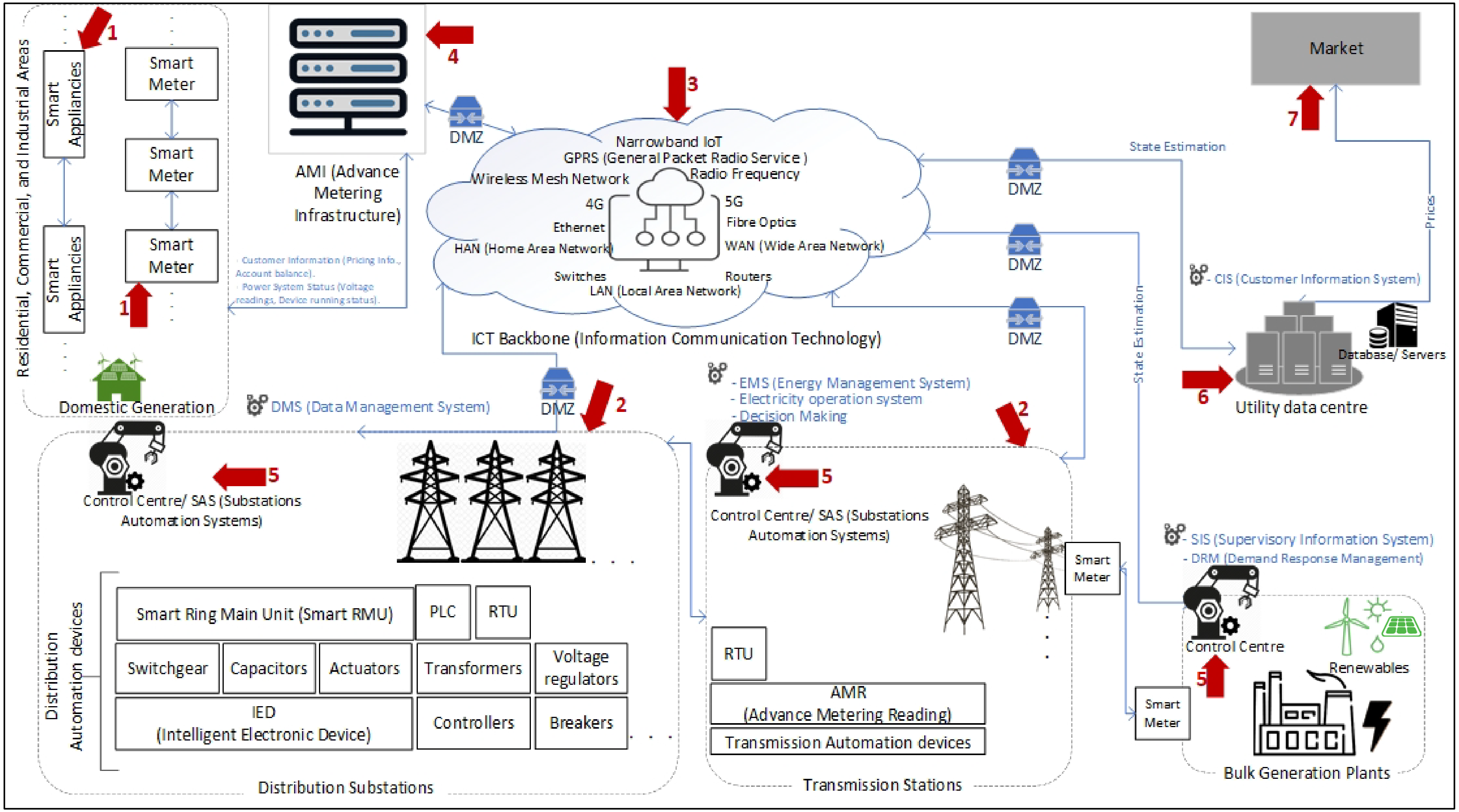

Residential, Commercial, and Industrial Areas
Smart Appliancies
Smart Appliancies
Smart Meter
Smart Meter
Smart Meter
1
1
Domestic Generation
4
AMI (Advance Metering Infrastructure)
- Customer Information (Pricing Info., Account balance).
- Power System Status (Voltage readings, Device running status).
DMZ
3
Narrowband IoT
GPRS (General Packet Radio Service)
Wireless Mesh Network
Radio Frequency
4G
5G
Ethernet
Fibre Optics
HAN (Home Area Network)
WAN (Wide Area Network)
Switches
Routers
LAN (Local Area Network)
ICT Backbone (Information Communication Technology)
DMZ
DMZ
DMZ
Market
7
State Estimation
Prices
State Estimation
- CIS (Customer Information System)
Database/ Servers
6
Utility data centre
DMS (Data Management System)
DMZ
2
- EMS (Energy Management System)
- Electricity operation system
- Decision Making
2
5
Control Centre/ SAS (Substations Automation Systems)
5
Control Centre/ SAS (Substations Automation Systems)
Smart Meter
- SIS (Supervisory Information System)
- DRM (Demand Response Management)
Control Centre
Renewables
5
Smart Meter
Bulk Generation Plants
Distribution Automation devices
Smart Ring Main Unit (Smart RMU)
PLC
RTU
Switchgear
Capacitors
Actuators
Transformers
Voltage regulators
IED (Intelligent Electronic Device)
Controllers
Breakers
Distribution Substations
RTU
AMR (Advance Metering Reading)
Transmission Automation devices
Transmission Stations

Figure 5-1: Information flow and the access points of the IoT-enabled smart grids

** 1-Smart Meters and Smart Appliances. 2-Distribution Substations, Transmission Stations, and Smart automation devices for transmission and distribution. 3-Generation Plants and Information Communication Technology (ICT) . 4-Advanced Metering Infrastructure (AMI). 5-SCADA (Supervisory Control and Data Acquisition)/Control Centre/SAS (Substations Automation Systems). 6-Utility data centre. 7-Market.

The experts were asked two questions:

- Are there any missing access points that need to be added?
- Is there anything you would change/remove concerning the access points?

Thirteen of the experts agreed on the access points shown in Figure 5-1. One could not be certain, as their specialism was IT.

Expert1 said that "*This figure is correct, and I have nothing to add to the access points*." **Expert2** agreed to the access points presented, saying "*You have mentioned all access points that could be potential points for an attacker*." **Expert3** said that "*All access points are correct and correspond to our practical datasheets*." "*Figure 5-1 is comprehensive and includes all the potential access points*," said **Expert4**. **Expert11** said that "*All access points and components presented in the diagram are excellent. It is suggested that bulk substations are added to Figure 5-1. Bulk substations are available on the edges of big cities. Also, bulk substations could supply four districts of the city, while distribution substations supply a district fully or partially*." Bulk power substations link the transmission system to the sub-transmission system, stepping the voltage down through a transformer (transformer substation), or linking high-voltage transmission lines from different parts of the system without changing the voltage (switching substation).

The access points were grouped into seven sections, based on the potential threats and controls. The same order is used in all sections of this research.

1. Smart meters and smart appliances
2. Transmission stations, distribution substations, and smart automation devices for transmission and distribution (switches, sensors, actuators, transformers, voltage regulator, capacitors)
3. Generation plant and ICT systems
4. AMI
5. SCADA
6. Utility data centre
7. Market.

Expert4 was confused about the market as an access point, and commented that "*Market here means the supply chain and it is not a part of the organisation as it is located outside the organisation. So, it could not be here in the diagram*." Despite his argument, the market still affects SG operations so is considered as an access point by this research.

Thirteen of the experts explained that two networks, operational technology and information technology, should be considered in SG. The operational technology network comprises the hardware and software used to monitor and control physical devices and processes of an industrial network such as SCADA and PLC. Conversely, the Information Technology network involves computers and solutions used in IT environments. **Expert3** said "*The infrastructure in all companies in the field contains two different networks, OT and IT. OT is the Operational Technology network of the core business that is located in the power plant or stations. This is a hundred percent isolated from the IT network.*" Both **Expert5** and **Expert6** stated that "*There are two separate networks, OT and IT.*" **Expert5** defined the OT network as "*the electrical network that contains stations for generation, transmission, and distribution, but IT is the Information Technology network.*"

Expert3, Expert4, Expert5, and Expert6 upheld the view that it is difficult for the OT to be cyber-attacked since Operational Technology (OT), including stations, devices, and SCADA, is not connected to the internet but has its own network. **Expert3** stated that "*This infrastructure is critical as any damage to a station will shut the electricity in, for example, 12 other cities connected to that station. So, Saudi Arabia is moving to Smart Grid gradually in phases. Currently, any traffic or connection between IT and OT is declined, despite all the arguments around it, because it will have resulted in electric blackouts as in Ukraine and many countries that shifted very fast towards IoT-enabled systems that are remotely accessed.*"

Despite OT being a separate network from IT, IT has been incorporated into OT recently for automation and to achieve smart environments. This is especially significant with the convergence of the industry and the IoT into (IIoT), where the line between IT and OT is becoming blurred (Caldwell, 2018; Nozomi Networks, 2020). This, in turn, expands the cyber threat landscape (Glenn *et al.*, 2017). Information breaches from OT cyber-incidents, such as electricity outages, can lead to huge financial losses and damages that can have far-reaching consequences (Hemsley and Fisher, 2018; Kimani *et al.*, 2019; Tufail *et al.*, 2021). From a security perspective, as OT and IT use the same standards they can be vulnerable to the same security threats, which could be addressed by the IoT-enabled SG model, which focuses on securing the information flow in both networks (IT and OT) in the IoT-enabled SG infrastructure.

All previous arguments led to questioning the adoption by Saudi Arabia of the smart meters project. **Expert3** explained that "*Saudi Arabia is launching phase 1 of the Smart Meters project, where the smart meters are installed, while an employee still takes the readings manually. In Phase 2, a next-generation of Smart Meters, embedded semi-cards will send the data using fibre optics provided by telecom companies.*" Accordingly, the model is important in Saudi Arabia's next stage, alongside the plans and initiatives undertaken towards an SG.

Expert3, **Expert4**, **Expert5**, **Expert6**, **Expert8**, **Expert9**, **Expert11**, and **Expert13** agreed that the smart meters network and each component of OT are isolated from the IT network using the segregation technique. "*With the introduction of the smart meter, and to secure the infrastructure, we apply a data diode with unidirectional traffic, or firewalls with unidirectional flow. The smart meter network is isolated from the IT network or ICT. A demilitarised zone (DMZ) is utilised when sending data from smart meters to the utility data centre for the billing system, for example*," said **Expert3**. **Expert5** and **Expert6** stated that "*There is no connection between the smart meter network and the IT network*."

Eleven experts confirmed the use of segregation, segmentation, data diode isolation, DMZ, and air gap. "*The infrastructure is segregated using Data Diode and LEDs (Light Emitted Diode), which is a segregation hardware tool*," said **Expert8**. "*Each component is isolated in a separate zone*," said **Expert11.** There is a common concept among experts that the segregation technique, which ensures unidirectional traffic, is secure enough against any threats. **Expert5** said that "*the electricity system architecture design applies segregation techniques on all edges in OT and IT networks, including the internet. This architecture reduces the chances of the Electricity Grid being attacked*."

Expert5 argued that "*when using an Air gap between transmission stations and the utility data centre, it is guaranteed that no attack can be carried out from the utility to the transmission stations. The data is allowed to be sent from transmission stations to the utility but cannot return the opposite way. So, the data flows from the OT to the utility one way. As a result, no attacker can enter the OT*." "*There is a flow from OT to IT, but there is no flow from IT to the OT network*," said **Expert8**.

Expert3, **Expert4, Expert5, Expert6, Expert9, and Expert13** were confident that even with segregation, there are still security risks due to patch management and remote diagnosis and control that are considered as a step toward smart automation in Saudi Arabia. Below is a selection of the noteworthy statements made.

"*Even with segregation, the attacker can break the grid in the firewall*," said **Expert3**. **Expert5** said that "*Although the system architecture applies segregation, there are many reasons that require connecting to the IT network and internet, including Remote support, patching, Anti-virus updates, and the monitoring process. These, in turn, expose the Electricity Grid to the security attacks*." **Expert5** then gave two scenarios where an attack could occur, even with an Air gap. The first is "*if we have installed the firewall but we missed out the security updates issued by Cisco patch, the attacker could utilise this vulnerability to gain access to the grid. So, even with an Air gap, the grid is exposed to security attacks*." The second scenario is where "*a human error mistakenly opens the internet port, or mistakenly sets the internet port number 23 as trusted traffic, which opens a connection vulnerability that could be utilised by an attacker*."

Expert3 mentioned that "*Current plans and new projects are focusing on automated management for SCADA, remote troubleshooting for substations, and automated support from abroad, as we have SCADA from different companies located outside the country, General Electric for example. Exploiting the technology to automate these matters is now becoming essential*." This, in turn, confirms the importance of the information security and security controls identified by this model.

Given all the above views, the conclusion is that the IoT-Enables Smart Grid information security model is required, even with segregation and isolation techniques.

While **Expert1**, **Expert3**, **Expert4**, and **Expert8** stated that an attack could be from inside the Grid instead of from outside, the model represents **all** potential access points regarding different types of cyber threat actors discussed previously, where insiders are included.

As for adding to the access points, the Handheld Unit (HHU) is one access point added as a potential vulnerability by **Expert10** and **Expert14**, and is a device used in Smart Meter maintenance. In addition, **Expert14** believed that the Meter Programming Devices should be added to the list of existing access points. This will not change the model, as this is included as Smart Meter-related devices. **Expert3** and **Expert6** were confused and added web and mobile applications and fibre optics as additional access points, but those were included in the diagram in access point 3 under Information Communication Technology. **Expert3** added the Load Distribution Centre as an access point, "*There is a Load Distribution Centre (LDC) for each Grid domain*," but this access point is already included in the control centre inside each domain. Similarly, **Expert5** pointed out that "*SCADA in the Generation segment is called the Distributed Control System (DCS), while it is called Programmable Logic Controller (PLC) in the Distribution segment that handles statistics*." **Expert13** added RTU as an access point. SCADA is the controlling part, which may have many components including LDC, DCS, RTU, PLC, RMU, and DCU. Thus, the current research considers all LDC, DCS, RTU, PLC, RMU, DCU, and SCADA are the names for the same access point, represented as the Control Centre (access point 5) in Figure 5-1, and all are considered the same from a security perspective. **Expert9** added the end-user as an access point "The *End-user needs to be added because some attacks happen due to misuse or culture issues by the end-user*." However, the end-user is one type of cyber threat actor that could be in any of the access points presented in Figure 5-1 including the Utility data centre, Control Centre, Smart Meter, etc.

As for changes to section A, nine experts from different departments (**Expert3**, **Expert4**, **Expert5**, **Expert6**, **Expert10**, **Expert11**, **Expert12**, **Expert13**, **and Expert14)** explained that each Grid domain has its own SCADA/control centre. **Expert5**, **Expert6**, and **Expert11** added that there are other smart meters between Generation plants and Transmission Stations, which calculate the amount of electricity exchanged between those different segments of the network. These changes are discussed in the next section.

5.2.1 Findings and Changes to the Model

This section summarises the findings of section A of the interviews, that is the identification of the Access points. Two changes were made:

- Adding Smart Meters between the Generation plants and Transmission domains
- Having different control centres inside each domain of the Smart Grid, including Generation plants, Transmission, Transmission, and Distribution, as shown in Figure 5-2.

5.3 Section B: Security Requirements and Controls

Table 3-6 was given to the experts showing the security controls that secure the information flow and the security requirements of the IoT-enabled Smart Grids. The experts were asked whether there was something that should be added to the security controls for the corresponding requirement. They were then asked if they would change/remove any controls. This section analyses their views on security requirements and controls in general. Then, under each requirement, their different views on the security controls will be discussed according to the codes given to each control. Four experts had no experience in the controls related to electricity networks, and could only comment on the IT aspects of the overall system.

For security requirements, all 14 experts agreed with the security requirements list, shown in Table 3-6. For example, **Expert8** said that "*The standards published in NIST IR 7628 contain all mentioned security requirements*."

As for the controls in general, all 14 experts acknowledged that the controls are significant, advanced, and comprehensive, such as **Expert1**, **Expert2**, and **Expert4** who stated that "S*ome of these controls have yet to be implemented since academic research is moving ahead of real practice*." **Expert6** said that "*All the controls listed here are recommended and applicable*." **Expert8** confirmed that "*the controls included in this research are compatible with the playbook of the company in terms of acting against threats*." **Expert8** said that "*The security controls mentioned in the research are beyond the standards in NIST IR and NERC CIP, which both measure the compliance of any organisation with the policies*."

6 experts discussed the link between the controls and the standards issued by the authorities in the field. For example, **Expert3**, **Expert4**, and **Expert8** confirmed that **NIST** is the main standard followed by the electricity company to maintain security. **Expert3** said that "*Some organisations follow NIST IR, while others merge the three standards NIST IR 7628, NERC CIP, IEC 61850, as those are the most popular standards*." **Expert4** argued against answering this section regarding the controls list, and needed to know which standard this list followed. Although the researcher explained that the controls

mainly followed NIST, **Expert4** preferred to answer generally on this part. **Expert4**, **Expert11**, and **Expert13** pointed out that the electricity company follows a combination of the four most popular standards **NIST IR 7628**, **NERC CIP**, **IEC 61850**, and **IEC 62443**. **Expert8** confirmed that "*NIST IR 7628 and NERC CIP are both measuring the compliance of any organisation to the policies*." Based on this feedback, all standards were reviewed to ensure no controls were missed, and after the review no controls were added to the model. **Expert10** and **Expert13** highlighted the use of *Device Language Messaging Specification (*DLMS). **Expert10** explained that "*electricity in Saudi Arabia follows the international standard **IEC 62056** for data, and it is also called Device Language Massaging Specification (DLMS) which is a protocol for communication between Smart Matere and AMI*." However, **Expert10** was mistaken as this is not a security standard as such, but a protocol for communication.

Concerning the controls and cost, six experts argued that because of cost limitations, prioritising is required to determine the number of security controls that need to be implemented. However, the cost is beyond the scope of this research. **Expert1**, **Expert3**, **Expert6**, **Expert7**, and **Expert 11** argued that "*it is important to balance implementing these controls with system performance, including computation time, speed and cost*." **Expert6** added that "*I agreed with the controls listed here, but should we apply them all? That is why it is important to balance Security controls, Performance, and cost. I suggest developing a scenario which defines a company's size, budget, business needs, critical infrastructure, and environmental circumstances. With this scenario, we can align business needs and maintain the budget at the same time*." **Expert6** mentioned that "*All identified controls can be implemented; however, it is the trade-off between the security and usability. More security controls will result in low usability and vice versa. Also, it is a trade-off between security and budget. Applying more countermeasures means more costs, and this is subject to business needs*." Although the experts stated that there may be a need to consider the cost when implementing the controls, it was explained that the assessment of cost is out of the scope of this research. The initial focus was to identify a list of the relevant controls and as the implementation was not considered in the current phase, issues of performance and cost do not need to be considered.

Concerning the risk and controls, **Expert3** considered all access points in the model as critical, and pointed to the SCADA as the most important point, by saying "*When it comes to Smart Grid, all these access points are critical, especially any aspect related to the SCADA system as it controls the power g*eneration, transmission, and distribution*.*" **Expert9** mentioned that "*the data/control centre is a critical access point because it contains the information, but if the Smart Meter is hacked, the damage will be less than the data centre*." Nonetheless, risk assessment is out of the scope of this research.

Expert6 commented that "*I found that many of the countermeasures listed can help in all security objectives, not specifically for one*." Therefore, a common list of security controls has been created to serve more than one security requirement. More details on the common controls list will be discussed in the next subsections.

Three experts raised some out-of-scope points. **Expert1** suggested prioritising access points based on a simulation called Contingency Analysis (CA), which is a "*what if*" scenario simulator that evaluates and prioritises the impacts on an electric power system when problems occur. However, research into this topic reveals that this analysis is about unplanned outages and failure of a small part of the power system, such as a transmission line, or transformer. This allows operators to be prepared to react to outages by using pre-planned recovery scenarios. **Expert1** also asked whether these controls are measurable, assessing and filtering them using the SMART criteria, an acronym for Specific, Measurable, Achievable, Realistic, and Timely, but normally applied to goal evaluation in administration. **Expert3** suggested running data classification for each access point to prioritise which of them requires more security controls. Similarly, **Expert3** and **Expert4** added physical security, security policies, and documentation to the controls. However, all these suggested aspects are out of the scope of the current research.

The following subsections discuss the experts' views on each control and whether the control is mapped correctly to the corresponding security requirements.

5.3.1 Authentication (Aun)

Expert1, **Expert5**, **Expert8**, **Expert9**, **Expert10**, and **Expert14** confirmed all the controls listed are used to ensure and boost Authentication in the Smart Grid. **Expert8** commented, "*I agreed on all controls mentioned here for Authentication, but we need to put one or two controls only due to cost and they depend on the risk and the level of data sensitivity at the access point*." **Expert9** commented "*I did not see Aun5 (Multi-factor authentication mechanism) is applied in the power field*," but **Expert12** said, "*Aun5 is not used currently but it is important to be used*."

Expert1, Expert4, and Expert7 confirmed Aun1, Aun3, Aun4, Aun5, and Aun6, especially Hash functions, SSL certification, and the Message Authentication Code (MAC) as authentication controls, which as expressed by **Expert1** as "*SSL control is used a lot*." **Expert1**, **Expert4**, **Expert6**, **Expert7**, and **Expert12** were not aware of Aun2 as a control, and **Expert1** mentioned that "*This is the first time I heard about this control*." **Expert6** said "*I do not know what Physically Unclonable Functions (PUF) are*." Similarly, **Expert12** stated "*I have never heard about PUF*." **Expert12** was also unfamiliar with both Aun1 and Aun4.

With Aun3 (MAC-attached, and HORS-signed messages), **Expert6** was not sure about this control and commented "*I do not know what this control is but digital signatures, in general, serve Confidentiality and Integrity*." **Expert9** commented "*MAC-attached and HORS-signed messages (Aun3) is a control that could be used in the Smart Meter and routers*."

Expert6 confirmed Aun5 and Aun6 and commented "*There are five factors of authentication which could be included here in this controls list: something you know, something you have, something you are, somewhere you are, something you do*." However, these are regulations and principles more than controls and techniques to mitigate Authentication-related threats. **Expert12** confirmed Aun6 by saying "*Automatic lockouts (Aun6) is implemented in our work. After 10 login attempts, the system will lock out the user automatically*."

Expert5 and **Expert6** added the Secure Session Management to the controls that serve Authentication. **Expert8** added the anti-spoofing algorithm as a control to kill spoofing threats. **Expert8** also added a control to the list and said, "*you may add 'comparing with baseline configuration' as a control to maintain Authentication*." However, this suggestion is considered as a detection technique rather than a security control.

5.3.2 Authorisation (Aur)

Expert4 confirmed Encryption in general, regardless of the type of this encryption as a control used to achieve Authorisation in the Smart Grid's information system. **Expert9** and **Expert10** confirmed all the controls list for Authorisation, but **Expert9** said "*I am not sure if the Attribute Certificates (Aur2) are applied in electricity here*." **Expert10** said "*Saudi Arabia plans to authorise using certificates as suggested in the control id (Aur2)*."

Expert5 and **Expert6** added another control, which is Privileged Access Management (PAM), and they commented "*PAM is wider than Role-Based Access Control (the control with code Aur4). This control is new in the field and it cuts the connection and disables the user in case of privilege escalation as well as recording the session for any investigation purposes*." **Expert6** added one more control, which is the Principle of Least Privilege (POLP), saying "*this principle is that users should only be granted the necessary privileges to complete their tasks*."

Expert6 and **Expert8** confirmed Aur4 (Role-Based Access Control and allow/block listing) and **Expert6** said "*Role-Based Access Control is correctly serving the Authorisation, and there are four types of Access Control can be included in this controls list for Authorisation: Mandatory access control, Role-Based Access Control, Discretionary access control, and Rule-based access control*." **Expert10** explained that Role-based access is considered a separate control that differs from allow/block listing, saying "*while* *allow/block listing specifies whether a user is allowed or prohibited to access the system, Role-Based access control deals with functionalities and privilege levels, such as write or read permissions. In Smart Meter, the privilege level is called the association*.” This research will include Role-based access control (Aur4) as a security control different from Allow/block listing included as (Aur5). **Expert6** did not know Aur1 and Aur2 saying “*to my knowledge and from my point of view, encryption in general (regardless if it is attribute-based or not) and certification serve Confidentiality and Integrity*.” **Expert6** was not sure about Aur3 and did not know this control.

Expert8 added ‘whitelisting’ as control, but it is another name for the allow/block listing. **Expert10** commented “*allow listing works with big-memory devices such as AMI, not Smart Meters as it has limited memory*.”

5.3.3 **Confidentiality (C)**

Expert4, **Expert5**, **Expert6**, **Expert7**, **Expert10**, and **Expert14** confirmed C1 which are symmetric and asymmetric algorithms as a security control to ensure Confidentiality. **Expert5** preferred to use asymmetric algorithms saying “*Symmetric algorithms are weak whereas asymmetric algorithms used two keys for confidentiality*.” **Expert6** said “*Symmetric algorithms are faster and have high performance, whereas Asymmetric algorithms increase the cost in terms of power consumption, processing time, and key size since it depends on two keys*.”

Expert6 added ‘session management’ to the controls serving Confidentiality, while **Expert5** mapped this control to Authentication. **Expert9** confirmed that “*Encryption, in general, is used for Confidentiality, but I do not know whether it is symmetric or asymmetric algorithms*.”

5.3.4 **Privacy (P)**

All controls were confirmed by **Expert5** and **Expert8** however **Expert5** did not know (P3) and (P4) as he expressed “*this is the first time I heard about those controls and this is the first time I have seen them from my experience*.”

In terms of P1 (Anonymisation) as a control, both **Expert6** and **Expert12** did not agree with this control saying **“***The electricity company has to know the customer’s identity for each meter reading*.” **Expert10** agreed on Anonymisation. **Expert6** said “*if the owner (Utility) is the seeker of the information about the Smart Meter, then it is useless to implement Anonymisation because the Utility has to know the identity of the Smart Meter for billing purposes. In this case, it just doubles the cost to the company. However, Anonymisation is recommended if another company seeks information about Smart Meters*.” Both (NISTIR 7628) and (IEC 62351) highlight the need for data privacy like anonymisation in their standards for energy industries. Although the researcher explained that Anonymisation allows the utility to identify the Smart Meter without revealing the customer's habits, **Expert6** argued that this is not related to Anonymisation and pointed to "*Preserving privacy in this way is what is covered by the EU GDPR (General Data Protection Regulation) that includes seven principles to maintain data privacy, which could be added to controls list here*." Reviewing EU GDPR showed that those are principles and regulations more than controls. Also, it is not special regulation for privacy only as it is a regulation for other security requirements such as Integrity and Confidentiality. For instance, the UK GDPR sets out seven key principles which are: Lawfulness, fairness and transparency, Purpose limitation, Data minimisation, Accuracy, Storage limitation, Integrity and confidentiality (security), Accountability.

Regarding P2 (Trusted aggregators), **Expert8** was not sure about including this control and argued that "*it is not totally correct, as aggregators are prone to hacking, which results in breaching thousands of Smart Meters since each aggregator covers hundreds of Smart Meters*." **Expert12** disagreed with P2 and argued that "*it is important to make meter readings clear instead of using aggregators because it is related to technical and financial matters*." **Expert9** commented that "*aggregator is a point of weakness because it is a point of centralisation. In case of attack, we will lose data for a complete district*." **Expert10**, **Expert12**, and **Expert14** were agreed on aggregators, and **Expert10** said "*it is called Data Concentrator Unit (DCU)*." Despite the views that disagree with aggregators, all practical IoT systems collect the data from multiple connected constrained IoT devices before it is sent to the cloud using the gateway, which is a Trusted aggregator (P2) that has better processing power than Smart Meters for encrypting the data and sending it as packages rather than sending individual data packets. Sending the data individually without encryption from Smart Meters to the Utility introduces weaknesses of security and communication.

Expert7 confirmed P3, P4, P5, P6 as controls, while **Expert6** argued that "*the controls P3, P4, and P6 are still theoretical solutions. They do serve the privacy 100%, but it should be tested by implementation*." **Expert9** was not sure about the controls P3, P5, and P6 and did not know them.

"The Homomorphic (P3) is new encryption and it is highly recommended as it is a solution for many security issues, but based on my research it still on paperwork and not implemented yet, and it is a high-cost solution," said **Expert6**.

"*I agree on encryption in general as a control for privacy. Consequently, I agree with the Homomorphic encryption (P3) because it is a type of encryption*," said **Expert10**.

Expert5 commented on the control P4 (Perturbation models) with "*this control can be used in the Utility and Market access points only, and cannot be applied to Generation, Transmission, and Distribution, and other operational components of the electric network, as they have trusted* *connections*." **Expert5** and **Expert9** confirmed that *"it is impossible to get benefit from the data if it has been perturbated or noise has added to it*." **Expert6**, **Expert10**, and **Expert12** argued that "*Perturbation models cannot be used in the electricity company as it needs to know the exact contents of the transmitted data accurately without any noise*." **Expert10** argued that "*the controls P3, P4, P5, P6 are not used in Saudi Arabia as electricity in Saudi is DLMS-compliant and certified*", but **Expert12** confirmed P6.

Expert5 asked the reason for choosing those specific algorithms, saying "*why you did not use Diffie-Hellman or RSA (Rivest–Shamir–Adleman) algorithm for encryption*." Based on this comment, the researcher explained that Diffie-Hellman and RSA are already included in the controls of codes Aun1, C1, and In1 respectively, which are cryptographic algorithms, but Homomorphic encryption preserves Privacy better and enables the third party to analyse or process the encrypted data without knowing the contents of decrypted data. If the utility's system is compromised, the encrypted data would remain secure.

5.3.5 **Integrity (In)**

Expert 5, **Expert8**, and **Expert14** confirmed In1, In2, In3, In6, In7, In8, and In10, as controls to ensure Integrity. **Expert6** agreed on In1, In2, In7, and In8. **Expert4** and **Expert5** were unsure about the controls In4, In5, and In9. **Expert4** did not know In11 and was not aware of this control before. **Expert5** was not sure about digital watermarking (In2), and **Expert9** was not sure about (In2) as well as (In1), because they did not know those controls. **Expert5** argued that "*The SSL and TLS controls are serving the Integrity and Authentication and they are replacing the watermarking control (In2). If I have SSL/TLS, then watermarking is not needed. SSL/TLS is widely used and there are new versions launched and developed So, it is updated*." **Expert10** commented "*I agree on the digital watermarking (In2) as a control on utility and other access points, apart from Smart Meters, as this control is not applicable to Smart Meters*." **Expert5** and **Expert10** confirmed all the controls, but **Expert5** asked for clarification for In4 and In5 before agreeing to them. **Expert6** was not sure about controls In4, In5, and In6 and did not know about them. **Expert8** stated that *"The adaptive cumulative sum algorithm (In4) is an algorithm on databases, and I did not see this control in our practical work*." **Expert10** said, "*in the adaptive cumulative sum algorithm (In4), the values are increasing. If the value is decreased, the attack is detected. So, this algorithm is for verification purposes. It is not encryption as it works on actual values*."

Expert8 commented on In5 about the installation of PMUs and stated "*using PMUs is such a double-edged sword, as PMU uses UDP (the User Data Protocol) that uses timestamps without session keys or sequence numbers. PMU slow down the encryption/decryption process. However, time is a priority in Smart Grid. Besides, PMUs are exposed to hacking*." **Expert6** commented on In10 with "*based on my* *knowledge, nonces are related to Blockchain. I have no idea about how to implement nonces outside the Blockchain. The implementation of this control needs to be checked outside the Blockchain environment*."

Load profiling (In6) is an agreed control as confirmed by **Expert8** who said, "*Load profiling is an extremely important control for Integrity, especially in the Distribution substations and market*."

Regarding the patch management control (In3), **Expert8**, **Expert9**, and **Expert10** agreed on this control, as well as **Expert5** who stated that "*Patch management serves more security requirements not only the Integrity. Microsoft is launching patches monthly for Windows servers to close some security vulnerabilities. If the system in the electricity company does not get those patches, an attacker can utilise those vulnerabilities and threaten the Availability. Also, if the vulnerabilities are related to the Active Directory or LDAP for example, the attacker may exploit* a *such vulnerability to gain access inside the system so, in this case, the Authentication is threatened. Once the attacker accesses the system, he may escalate the privileges which means the Authorisation is threatened*." In contrast, **Expert6** argued that patch management (In3) is a control that serves Availability more and said "*For example, Microsoft sends updates and patches for Windows servers. The idea behind these updates is to make the device more secure. When the device is more secure, it will be more available, but it is not related to the Integrity*." Consequently, patch management will be considered a common control that may support many security requirements. Patch management involves the regular update of software to fix vulnerabilities that could be exploited to supporting confidentiality and availability (Dissanayake et al., 2022; Mell et al., 2005).

Expert6 noted that "*Query sanitisation (In9) is related to databases and it serves Integrity and Confidentiality as well. It is also related to the web application. The attacker has a specific technique of sending a query through the login fields to retrieve the backend database. That is why we apply query sanitisation*." Of the same opinion, **Expert8** said "*I did not see Query sanitisation (In9) often used, and it is used in the backbone and databases, not in smart devices*." **Expert10** agreed to this control, but he said "*Query sanitisation (In9*) *is not applicable to Smart Meters*."

Expert8 added a control described as "*comparing with baseline configuration, and the normal situation is considered as a control for Integrity*."

5.3.6 Availability (Av)

Expert5 and **Expert8** agreed to all controls listed in this requirement, apart from Av2 (Frequency quorum rendezvous), as **Expert8** was not sure about this and did not know it, while **Expert5** stated that "*this control is dealing with hopping and routing from one node to another for the fastest or shortest* *path, which I don't think exists or used in our company*." **Expert10** stated that "*the frequency quorum rendezvous (Av2) is something specific; you don't have to go into this detail*." **Expert10** agreed to all controls listed for Availability, but commented on Av3 and Av6. "*IDS and QoS are not applicable to Smart Meters. IDS is suitable for AMI. QoS is related to the network-level*."

Expert5 added the Web Application Firewall (WAF) to the controls list serving Availability, mentioning that "*WAF is a control used to protect the Smart Grid against Distributed Denial Of Service attack (DDOS). This control may be costly for the company. Subsequently, it was recommended that this control be used when the risk exists*." Both **Expert5** and **Expert8** agreed on adding firewalls as a control that serves all listed security requirements, and stated that "*firewalls, especially the next generation firewalls, contain many solutions including IDS, IPS, proxy, and sandboxing*."

Expert3, **Expert4**, **Expert5**, **Expert6**, and **Expert8** added Segregation, Segmentation, Data Diode isolation, DMZ, and Air gap, as controls to serve Availability. **Expert5** explained that "*Segregation is using DMZ and firewalls, whereas segmentation using Virtual Local Area Networks (VLANs). Both are located between the private internal network and the public internet*." **Expert6** argued that DMZ could serve authorisation, saying "*DMZ is mainly serving Availability and could serve Authorisation as well. It is not serving Authentication or Integrity as the main goal*."

Both **Expert5** and **Expert6** discussed the idea of Defense in Depth, "*the company is using DMZ and applying Defense in Depth strategy*." Defence in depth is a cybersecurity approach that provides multiple defensive measures and mechanisms that are layered to secure valuable information. In case one mechanism fails, another mechanism will thwart an attack. Some examples of Defense in depth are Firewalls, Demilitarised zones (DMZ), Virtual private networks (VPN), and antiviruses. Consequently, Defence in Depth represents a conceptual cybersecurity strategy, whose suggested measures are already included in this research.

Both **Expert5** and **Expert6** added Redundancy as a security control, and said that "if any device had a failure and became unavailable, then a redundant device continues to work." **Expert6** explained that redundancy is different to Load balancing (Av7), and said that "Load balancing is more about task distribution across several servers, whereas Redundancy is more about backup." **Expert5**, **Expert6**, and **Expert8** added anti-DDOS protection or measure to the controls list serving Availability. **Expert7** confirmed Av3, Av4, Av5, Av6, Av7, and Av8, and commented "*IDS detects the attack and gives an alert, while IPS takes an action, such as isolating the affected device*."

5.3.7 **Non-Repudiation (N)**

All controls were confirmed by all experts. **Expert6** did not know the control coded N1, as did **Expert8**, who argued "*I agreed on Mutual inspection technique (N1) if both devices have the same manufacturer, specification, and protocol. In this case, the connection frequency could be checked and inspected. But if the devices have different manufacturers, specifications, and protocols, I cannot agree on this control in that case*."

"*Absolutely all controls listed here are correct and accurately mapped to the corresponding security requirement which is Non- repudiation*," commented **Expert4** and **Expert5**.

5.3.8 **Findings and Changes to the Model**

This section summarises the findings of section B of the interviews, mapping each controls list to the corresponding security requirement. The experts viewed the requirements and controls from their fields of expertise. Taking this observation into account they concluded that the controls are correctly mapped to the requirements. The one exception was Patch management, where the experts were of the view that it could serve many other security requirements rather than Integrity only.

All experts agreed that encryption is important for Privacy, but questioned the use of advanced specific techniques of encryption to preserve privacy: Homomorphic encryption (P3), Perturbation models (P4), Verifiable computation models, and zero-knowledge proof systems (P5), Data obfuscation techniques (P6). Although Privacy in IoT-enabled SG is important, identifiable data is not very sensitive of the criticality level, such as medical data records. Therefore, basic encryption should be sufficient to preserve Privacy. Consequently, the controls P3, P4, P5, and P6 were omitted and replaced by basic encryption as a control (P3). Similarly, the experts saw the Frequency quorum rendezvous (Av2) as an unnecessary detail, given the focus of this research. Thus, Av2 was omitted.

The Availability requirement was where the experts wanted to add more controls. These controls include Redundancy, Web Application Firewall (WAF), anti-DDOS, Segregation, Segmentation, Data Diode isolation, DMZ, and Air gap. For Authentication, the experts added two controls which were: Secure Session Management, and the anti-spoofing algorithm. For Authorisation, the experts added two controls which were: Privilege Access Management (PAM), and the Principle of Least Privilege (POLP). Moreover, Role-based access control and allow/block listing were split into two different controls Aur4 and Aur5.

A common controls list was added to the model to serve more than one security requirement, which includes: Patch management, Firewalls, and EDR.

All added controls and changes are represented in Table 5-1, which shows the confirmed modified controls list and the common list.

Table 5-1: Confirmed security controls and corresponding security requirements

Security Requirement	Security Control	Code
Authentication	1. Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators	Aun1
	2. Physically Unclonable Functions (PUF)	Aun2
	3. MAC-attached, and HORS-signed messages	Aun3
	4. Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)	Aun4
	5. Multi-factor authentication mechanism	Aun5
	6. Automatic lockouts	Aun6
	7. Secure Session Management	Aun7
	8. Anti-Spoofing algorithm	Aun8
Authorisation	9. Attribute-Based Encryption	Aur1
	10. Attribute Certificates	Aur2
	11. Attribute-Based Access Control System based on XACML (Extensible Access Control Markup Language)	Aur3
	12. Role-Based Access Control	Aur4
	13. allow/block listing	Aur5
	14. Privileged Access Management (PAM)	Aur6
	15. Principle of Least Privilege (POLP)	Aur7
Confidentiality	16. Symmetric and asymmetric algorithms and Public Key Infrastructure certificate (PKI)	C1
Privacy	17. Anonymisation	P1
	18. Trusted aggregators	P2
	19. Encryption	P3
Integrity	20. Cryptographic hashing functions and session keys	In1
	21. Digital watermarking	In2
	22. Adaptive cumulative sum algorithm	In3
	23. Secure Phasor Measurement Units (PMUs) installation	In4
	24. Load profiling algorithms	In5
	25. Timestamps	In6
	26. Sequence numbers	In7

	27. Query sanitisation	In8
	28. Nonces	In9
Availability	29. Use multiple alternate frequency channels according to a hardcoded sequence	Av1
	30. Anomaly Intrusion Detection Systems (IDS)	Av2
	31. Specification-based IDS	Av3
	32. Intrusion Prevention Systems (IPS)	Av4
	33. Quality of Services (QoS)	Av5
	34. Load balancing	Av6
	35. Operating system-independent Applications	Av7
	36. Redundancy	Av8
	37. Web Application Firewall (WAF)	Av9
	38. Anti-DDOS algorithm	Av10
	39. Segregation, Segmentation, Data Diode isolation, DMZ, and Air gap	Av11
Non-repudiation	40. Mutual Inspection technique	N1
	41. Unique keys and digital signatures	N2
	42. Transaction log	N3
Common Controls across all the above requirements	43. Patch management for flaw remediation	Common1
	44. Firewalls	Common2
	45. Endpoint for Detection and Response EDR	Common3

5.4 Section C: Threats and Controls

In this section, a total of seven tables were shown to the experts see Table A-2 to Table A-8 in Appendix A. Those tables map the controls to each access point, against the list of applicable threats to the corresponding access point. Experts were asked:

1. Whether there was something that should be added to the threats for the corresponding access point (see section 5.4.1).
2. Whether they would change/remove something concerning the threats.

In the second round, they were asked

1. Whether they would add any control for the corresponding access point.
2. Whether they would change/remove something concerning the controls.

5.4.1 Threats and Access Points

Expert3, **Expert4**, **Expert5**, **Expert6**, **Expert8**, **Expert10**, **Expert11**, **Expert12**, **Expert13**, and **Expert14** agreed on the list of attacks for each access point. **Expert3** stated "*all these attacks are potential threats in electricity systems*." **Expert4** emphasised that SQL injections are not applicable on Smart Meters and noted that "*SQL injections are applicable at any access point where the Database is found*."

Expert4 argued "*if the Distribution Substations and Transmission Stations are smart, using IP addresses and communication module, then all threats listed here are applicable. Whereas, if the stations are not smart, using a serial communication protocol i.e. legacy devices, then it is difficult to be attacked by the internet-based threats listed here. Thus, the stations could be attacked by other types of threats such as wire attacks*."

Expert6 and **Expert8** added a threat called 'Buffer overflow' on Smart Meters. **Expert8** also added the Social engineering attack to the list that could threaten Smart Meters. **Expert8** pointed out how social engineering attacks threaten Smart Meters saying "*the attacker could pretend that he is an authorised employee and can then be logged into the system and download the firmware of the Smart Meter*." However, this research focuses on internet-based threats, and so non-internet-based social engineering attack carried out by humans as described by **Expert8** such as Phishing by phone or voice calls is out of the scope. However, internet-based Phishing such as email Phishing and search engine/websites Phishing is included in the scope.

Expert8 added a SolarWinds attack on substations, which is a highly sophisticated cyber threat where the advanced persistent threat (APT) actors threaten the supply chain of a commercial software application made by SolarWinds, inserting a backdoor into the software (CIS, 2021).

Expert5 and **Expert6** added Distributed Denial Of Service threat (DDOS) to the threats list. However, DDOS is considered a type of Denial Of Service attack in which multiple systems send fake requests to a single target.

Expert4, **Expert5**, **Expert8** agreed that "*Utility and Market are directly connected to the internet using WAF, while the other access points could be connected to the internet through the utility only using Air gap or Data Diode*."

In terms of Smart Meters and data tampering, **Expert10** argued that "*a data tampering attack only happens on physical tampering, but with network tampering, it can happen to AMI*."

5.4.2 Security Controls and Access Points

Expert2 reported "*These security controls are deeply covered for each access point in this research*." **Expert8** stated "*it is of importance to invest in the security controls presented in this research*."

Expert4 and **Expert8** argued that "*if there is a system on top of the legacy low-level devices, such as substations and SCADA, then all controls suggested by this research are applied and assigned correctly to the access points*."

Expert10 commented "*this research has covered the main security controls, which is more than enough considering that Smart Meters are simple restricted devices that have been assigned with one function to perform*."

Expert5 added two controls, the first for a Malware attack and the second for a Social engineering attack. "*I suggest adding one layer to detect and respond to Malware, that will be Endpoint Detection and Response (EDR). Also, I suggest increasing the awareness level by training to mitigate the Social Engineering attack*," said **Expert5**.

Expert6 commented that "*From my point of view, one countermeasure will tackle more than one security threat. As a result, I recommend being more generic. Later on, you can have some exceptions when you see that one countermeasure is tackling a specific threa*t." This was followed by advice to "*Start implementing the basic countermeasures (IDS, Cryptography, etc.), then move gradually to the more advanced ones. Since this is how it works in the field*."

The following details the reviews on the controls for each access point.

- **Smart Meters and Smart Appliances**

Expert4, **Expert5**, **Expert6**, **Expert8**, **Expert10**, **Expert12**, and **Expert14** confirmed all the controls that mapped to Smart Meters, while **Expert10** was not sure about Verifiable computation models (P5). **Expert10** and **Expert12** commented on Homomorphic encryption (P3) saying “*I agree on (P3) as encryption in general, but am not sure about this specific type of Homomorphic encryption*.”

Expert9 commented “*MAC-attached and HORS-signed messages (Aun3) is a control that could be used with Smart Meters*.” **Expert8** commented “*Hash Functions and MAC (Aun1 and Aun3) are controls to achieve Authentication in Smart Meters*.” **Expert10** upheld the view that “*SSL certificates (Aun4) as a control is not applicable to Smart Meters because SSL is suitable for network devices such as AMI, not Smart Meters*.” **Expert10** added (DLMS) as a control for Smart Meters, saying “*it is important to add (DLMS) to the controls list for Smart Meters*.” However, DLMS, as explained before, is not a security control as such, but a communication protocol.

For Authorisation, **Expert10** said that “*the essential control for Authorisation is the Role-Based Access Control*.” **Expert12** was not familiar with the Attribute-Based Access Control System based on XACML (Aur3).

For Confidentiality, **Expert10** and **Expert12** confirmed C1 as a control. **Expert10** commented “*Symmetric and asymmetric encryption (C1) is a mandatory control for data security, but an optional control for communication in Smart Meters*.” **Expert8** argued that encryption “*is not applied to Smart Meters because it requires more computations and processing, and that causes more heating*.”

For Integrity, **Expert4** commented on cryptographic hashing functions (In1) as a control for Integrity saying “*Smart Meters are restricted devices. So, it is important to use short key length in the cryptographic hash functions*.” **Expert4** was not sure about In4, In5, and In11 and did not know those controls, which are respectively: PMUS, Adaptive cumulative algorithm, and Nonces.

Expert4, **Expert8**, and **Expert12** agreed that *watermarking (In2)* is not applicable to Smart Meters as a control. “*Digital watermarking, as a control for Integrity, is applicable for file security, not Smart Meters*,” said **Expert4**. **Expert10** argued that “*the Adaptive cumulative sum algorithm (In4) is applicable on AMI, not on Smart Meters because they cannot do this cumulative summation, only AMI can*.”

Expert8 was not sure about the Quality of Services (QoS) (Av6) and did not know of this control.

- **Distribution Substations, Transmission Stations, and Smart Automation Devices for Transmission and Distribution**

Expert5, **Expert6**, **Expert8**, and **Expert14** confirmed all the controls suggested for Distribution Substations, Transmission Stations, and Smart automation devices. **Expert8** confirmed that "*cryptographic hash functions cannot be applied to any station or substation, as it is a legacy device*." **Expert12** confirmed all assigned controls for Authentication, Authorisation, Integrity, Availability, and Non-Repudiation. For Privacy, **Expert12** agreed on trusted aggregators (P2) while disagreeing with on (P1) commenting "*The recognition of identity is required in our work*." Similarly, **Expert12** disagreed with (P4), and reasoned that "*it causes noise, and the data accuracy will be affected*." He also disagreed with (P6), reasoning "*as this control includes a buffering process that could cause a delay in transmission*."

Expert4, and **Expert8** emphasised that Transmission and Distribution stations are the same in terms of controls, "*the only difference being the voltage level; since the Transmission station has a high voltage while the Distribution station has a medium voltage*."

Expert8 said that "*it is difficult for an intrusion to reach and threaten automation devices, such as transformers, actuators, capacitors, switches, and voltage regulators. This is because of the huge size of those devices with very high voltage in an electromagnetic field. Switches could be connected to an ethernet which makes the threat possible, or it could be gear yard switches which are dangerous devices that produce sparks while working. So, these devices are dangerous, such as transformers that have oil heating that could damage a full district. The way to hack the automation devices is only via the substation. The threat has a limited effect on those devices because there is protection in the settings of these devices that prevent any changes, to the current for example*."

Expert8 added a new control which is 'installation of spy substation node or Honeypot', describing "*the attack will count this node as a substation. This node will learn how the attack hacked the system, then the damage will be treated in the affected node*." However, Honeypot is a monitoring and analysing process that provides valuable information or resources to attackers. Therefore, Honeypot has not been added to the control list in this research. **Expert8** added another control which is 'the virtual private network (VPN) encrypted tunnel', where the tunnel in transmission and distribution stations is encrypted between devices. **Expert8** also added another security control which is 'the dark fibre tunnel', where the fibre is dedicated from point to point. "*Usually, the fibre tunnel is shared between many organisations, but the dark fibre is not shared*." However, dark fibre tunnel is considered as physical security which is out of this research's scope.

- **Generation Plant and ICT**

Expert4, **Expert5**, **Expert6**, **Expert8**, and **Expert14** confirmed all controls listed for Generation Plant and ICT.

Expert4 and **Expert8** confirmed that Generation plants and ICT are both on the IT network level. Subsequently, all security controls are required to be assigned to these access points.

Expert12 confirm all assigned controls for Authentication, Authorisation, Confidentiality, Integrity, Availability, and Non-Repudiation, and confirmed the Multi-factor authentication mechanism (Aun5) saying "*we hope to use Multi-factor authentication mechanism (Aun5). It is not currently used in our work*." However, **Expert12** was not sure about PUF (Aun2) and the Attribute-Based Access Control System based on XACML (Aur3) and had not met those controls before.

For Privacy, **Expert12** agreed on Trusted aggregators (P2). Similarly, Verifiable computation models (P5) were confirmed but after explanation from the researcher. For Homomorphic encryption (P3) and Data obfuscation techniques (P6), **Expert12** was not sure and did not know those controls. **Expert12** disagreed on Anonymisation (P1) and commented "*The recognition of identity is required in our work*," also disagreeing on (P4) reasoning "*it causes noise, and the data accuracy will be affected*."

For Availability, **Expert12** agreed on all assigned controls but commented "*Quality of Services (Av6) is not used in our work*." **Expert12** further disagreed on Operating system-independent Applications (Av8), saying "*There are no applications used in our communication department*."

For Integrity, **Expert12** confirmed Cryptographic hashing functions and session keys (In1) as a control and said, "*this is a useful control, but I do not know if it is used*." Nonces (In11) were also confirmed as mentioned "*this is excellent control using a unique reference number for each exchanged message*."

- **Advanced Metering Infrastructure (AMI)**

All controls mapped to the AMI were confirmed by **Expert5**, **Expert6**, **Expert8**, **Expert10**, and **Expert14**. **Expert10** commented on SSL Certificates (Auth4), saying "*SSL and TLS (Aun4) are controls that fit the AMI besides using the Device Language Messaging Specification (DLMS) as a communication protocol*."

Expert8 confirmed that Attribute Certificates (Aur2) control is not applied to AMI because this control is expensive and costs money.

For Integrity, **Expert8**, **Expert10** agreed on all controls assigned to AMI.

For Availability, **Expert10** did not include the Frequency quorum rendezvous (Av2) as a control for AMI, arguing that "*frequency quorum rendezvous is too specific a control*."

- **SCADA (Supervisory Control and Data Acquisition)**

Expert5, **Expert6**, **Expert8**, and **Expert14** confirmed all controls assigned to SCADA. **Expert8** mentioned that "*certificates in both controls (Aun4) and (Aur2) are applied at the application level, not applied on the SCADA level. Also, Multi-factor authentication mechanism (Aun5) is applied at the application level, not applied on SCADA*." This is, in turn, confirms that the controls (Aun4, Aun5, and Aur2) are not assigned to SCADA.

Expert8 noted that "*we can take and read data from SCADA, but we cannot alter or send data to SCADA. So, SCADA does not take commands as it is a low-level device that talks to voltage in a one-way communication*."

For cryptographic hashing functions, **Expert8** commented "*SCADA does not understand hashing functions as it is not designed to have a processor. At the SCADA level, there is no hashing because it is flat files and row data. So, SCADA has a different hash technique that is required to be fast, in milliseconds*." This is, in turn, confirms that the control (In1) is not assigned to SCADA.

- **Utility Data Centre**

Expert5, **Expert6**, **Expert8**, and **Expert14** have confirmed all the controls assigned to the Utility data centre. **Expert8** was not familiar with HORS-signed messages (Aun3), and did not know the Adaptive cumulative sum algorithm (In4), or the Frequency quorum rendezvous (Av2).

For Integrity, **Expert8**'s opinion was "PMUs *installation (In5) is a tool reporting to the utility data centre from all other access points. So, it cannot report the issues with the Utility itself*." This is, in turn, confirms what was suggested by this research, that In5 cannot be assigned to the Utility.

- **Market**

Expert5, **Expert6**, **Expert8**, and **Expert14** confirmed all the controls mapped to the Market.

5.4.3 Findings and Changes to the Model

This section summarises the findings of section C of the interviews, mapping each controls list to the corresponding access points and threats. The experts viewed the threat and controls from their fields of expertise. Taking this observation into account they concluded that the controls are correctly mapped to the access points and threats. Some experts did not comment on all aspects and access points, as each team is responsible for only part of the Smart Grid. One control previously added in section B was the installation of 'Endpoint for Detection and Response (EDR)', which is mapped against Malware threats for all access points. Staff training is not been added to the controls list as it is not an internet-based technical control. Non-technical controls and human-based controls are out of the research's scope.

Since this research focuses on IoT-enabled Smart Grids, it assumes using IP addresses. and legacy systems are therefore not considered as potential physical threats, only internet-based threats. For the same reason, non-internet-based social engineering is out of the scope of this research except for internet-based Phishing such as email Phishing and search engine/website Phishing. Furthermore, this research focused on the common types of internet-based Security threats, identified from the literature and STRIDE analysis. Therefore, the current research did not investigate sophisticated multi-stage threats such as SolarWinds and APT.

Buffer overflow attack will not be added to the threats list as it is considered a common type of Denial Of Service attack (DOS), and is an effective method for performing (DOS) attacks (Roth and Spafford, 2011). The Buffer overflow attack is included in the DOS. Similarly, DDOS is not added to the threats list because it is considered a type of DOS attack, in which multiple systems send fake requests to a single target.

5.5 Smart Grid in the KSA

As reported by experts previously, NIST IR and NERC CIP measure the compliance of any organisation with the policies while the information security model in this research is beyond the standards of NIST. The quotations below demonstrate that this research is required and useful to support the Kingdom of Saudi Arabia in its digital transformation process and its initiatives toward implementing secure IoT-enabled Smart grids.

Expert3 said "*In the companies that deal with infrastructure, there are many Business lines. For example, in the electricity field, there is a Transmission line, Distribution line, and Generation line. All those lines have their SCADA control systems. The Transmission line in KSA is called the National Grid. The National Grid is responsible for the electricity outside the city, and it transmits the electricity from the Generation to the Distribution line or to substations inside the city. The Distribution line then supplies the electricity to the residential and commercial parts of the city, to homes and factories*."

Expert3 and **Expert13** pointed to the ECRA website and report, as it contains the Smart Metering and Smart Grids Strategy for the Kingdom of Saudi Arabia.

Expert3 advised that "*the manufacturer is always aware of the best practices for security in the market. Each manufactured device has different security solutions*." **Expert3** explained that "*the IT security department is responsible for desktops, servers, and systems for the employees in the company's building. Each OT department has its own data centre and security team (generation, transmission, distribution). Both of them are under the same management*."

Expert3, **Expert5**, **Expert10**, **Expert11**, **Expert12**, and **Expert13** reported that the transmission department is also responsible for Power transmission between Saudi Arabia and other countries, and for High voltage stations, medium voltage stations, low voltage stations, and distribution substations inside the city. Smart Grid and electricity production are available in the Oil sector as well as the electricity sectors in Saudi Arabia.

Expert3 and **Expert5** explained that "*the Transmission department is called the National Grid in Saudi Arabia*."

Expert5 stated that "*ABB, Schneider Electric, Yokogawa, and Siemens, are all vendors in Saudi Arabia developing solutions inside the OT network*."

Expert8 stated "*in our oil company, there are Smart Meters and smart solar energy. Currently, we are working on a new application which is Smart Micro Inverters and we are trying to use the Internet of Energy (IOE) to control the Smart Micro Inverters remotely, to change the algorithms inside the Smart Inverters to produce more electricity and then more power production*."

Expert10 stated that "*the smart appliances will be access points in the advanced phase of electricity, not implemented currently in Saudi Arabia, but you can keep it in Figure 5-1*." **Expert10** reported that "*the Advanced Electronics Company (AECL) is a Saudi company that purchases Smart Meters to install them into residential houses*."

Expert11 stated "*Smart Meters in Saudi Arabia are manufactured from different brands and companies and assembled in locally*." During the recent Saudi Digital Transformation effort, 4 million meters, equivalent to 40% of installed Smart Meters, were built in Saudi Arabia from locally manufactured components, and were installed all over the Kingdom by the Saudi Electricity Company.

Expert11 stated "*There are many companies that produce electricity in Saudi Arabia, such as the National Grid, ENGIE SA, NOMAC, AQUA POWER, Aramco, and AECL*." **Expert10** and **Expert11** explained that "*Mobily is the Saudi telecommunications company that supplies the network between the customer and the electricity company*," said **Expert10**. "*Mobily company is the network operator in the electricity company and it is the supplier for the semi-card in Smart Meters*," said **Expert11**.

Expert10 mentioned that "*Ernst & Young (EY) is a UK consultancy company for the Smart Meters project in Saudi Arabia*."

Expert11 explained that "*the load management relies on the network dashboard that shows the load needed by a city or an event. As a result, efficient load management allows us to raise the electric load for the events in need, such as pilgrimage or any festival*."

Finally, they stated that the next-generation Smart Meters will be shipped with semi-cards, that represent the communication module with the ICT.

5.6 **Summary**

This chapter has discussed the results of the expert interviews. In order to confirm the information security model, semi-structured interviews were conducted. This was to confirm the security requirements, threats, and controls at each access point that were previously identified by STRIDE analysis and the literature review, and to modify the existing criteria or explore other criteria that had not been mentioned.

After assessing the findings, five controls were omitted, two controls were divided, nine controls were added, and one control was re-assigned to many security requirements, which creates a common control list, as can be seen in Table 5-1.

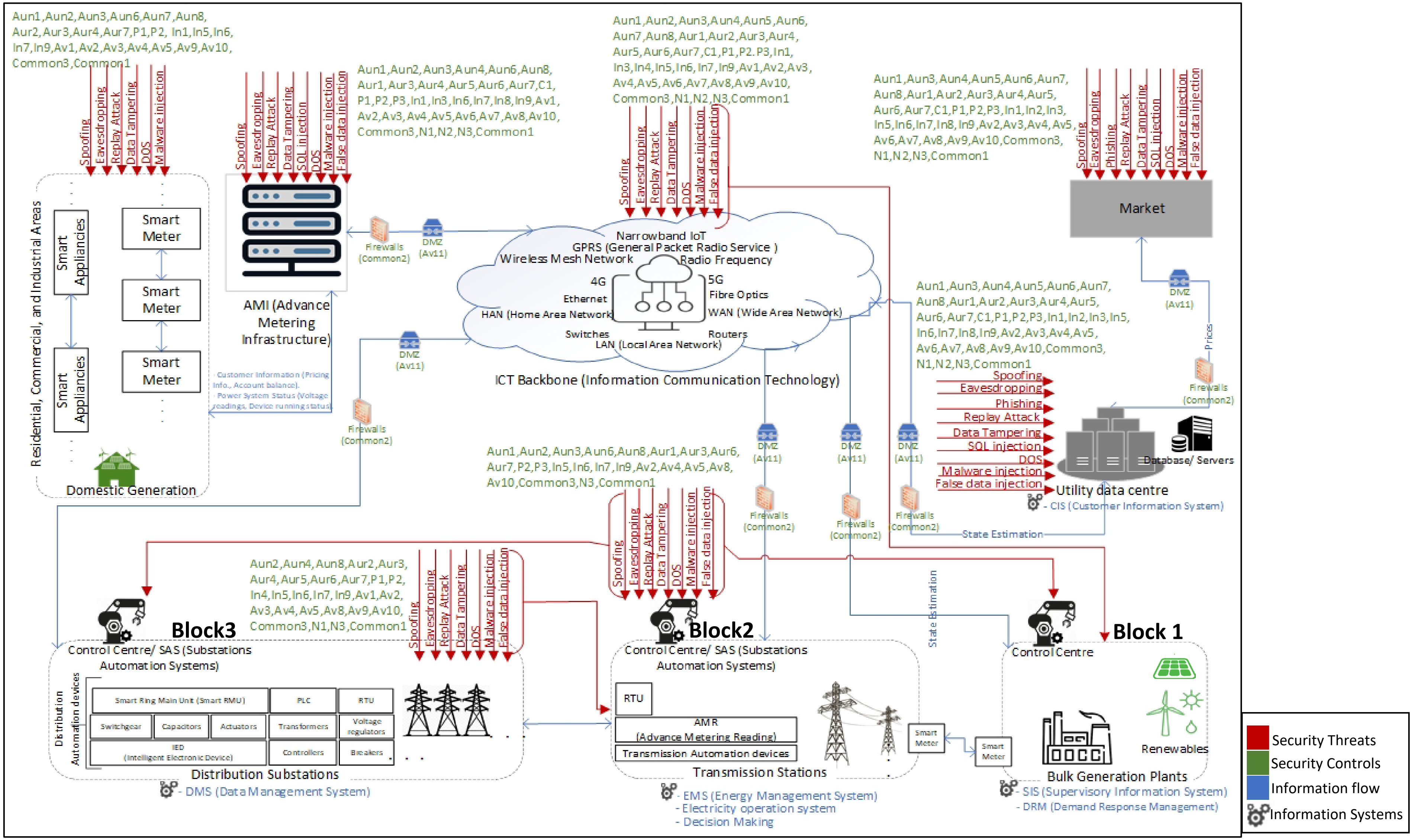

Aun1,Aun2,Aun3,Aun6,Aun7,Aun8,
Aur2,Aur3,Aur4,Aur7,P1,P2, In1,In5,In6,
In7,In9,Av1,Av2,Av3,Av4,Av5,Av9,Av10,
Common3,Common1
Spoofing
Eavesdropping
Replay Attack
Data Tampering
DOS
Malware injection
Residential, Commercial, and Industrial Areas
Smart Appliancies
Smart Meter
Smart Meter
Smart Meter
Smart Appliancies
Domestic Generation
Spoofing
Eavesdropping
Replay Attack
Data Tampering
SQL injection
DOS
Malware injection
False data injection
Aun1,Aun2,Aun3,Aun4,Aun6,Aun8,
Aur1,Aur3,Aur4,Aur5,Aur6,Aur7,C1,
P1,P2,P3,In1,In3,In6,In7,In8,In9,Av1,
Av2,Av3,Av4,Av5,Av6,Av7,Av8,Av10,
Common3,N1,N2,N3,Common1
AMI (Advance Metering Infrastructure)
Firewalls (Common2)
DMZ (Av11)
- Customer Information (Pricing Info., Account balance).
- Power System Status (Voltage readings, Device running status)
DMZ (Av11)
Firewalls (Common2)
Aun1,Aun2,Aun3,Aun4,Aun5,Aun6,
Aun7,Aun8,Aur1,Aur2,Aur3,Aur4,
Aur5,Aur6,Aur7,C1,P1,P2.P3,In1,
In3,In4,In5,In6,In7,In9,Av1,Av2,Av3,
Av4,Av5,Av6,Av7,Av8,Av9,Av10,
Common3,N1,N2,N3,Common1
Spoofing
Eavesdropping
Replay Attack
Data Tampering
DOS
Malware injection
False data injection
Narrowband IoT
GPRS (General Packet Radio Service)
Wireless Mesh Network
Radio Frequency
4G
5G
Ethernet
Fibre Optics
HAN (Home Area Network)
WAN (Wide Area Network)
Switches
Routers
LAN (Local Area Network)
ICT Backbone (Information Communication Technology)
Aun1,Aun3,Aun4,Aun5,Aun6,Aun7,
Aun8,Aur1,Aur2,Aur3,Aur4,Aur5,
Aur6,Aur7,C1,P1,P2,P3,In1,In2,In3,
In5,In6,In7,In8,In9,Av2,Av3,Av4,Av5,
Av6,Av7,Av8,Av9,Av10,Common3,
N1,N2,N3,Common1
Spoofing
Eavesdropping
Phishing
Replay Attack
Data Tampering
SQL injection
DOS
Malware injection
False data injection
Market
DMZ (Av11)
Prices
Firewalls (Common2)
Aun1,Aun3,Aun4,Aun5,Aun6,Aun7,
Aun8,Aur1,Aur2,Aur3,Aur4,Aur5,
Aur6,Aur7,C1,P1,P2,P3,In1,In2,In3,In5,
In6,In7,In8,In9,Av2,Av3,Av4,Av5,
Av6,Av7,Av8,Av9,Av10,Common3,
N1,N2,N3,Common1
Spoofing
Eavesdropping
Phishing
Replay Attack
Data Tampering
SQL injection
DOS
Malware injection
False data injection
Database/ Servers
Utility data centre
- CIS (Customer Information System)
DMZ (Av11)
DMZ (Av11)
DMZ (Av11)
Firewalls (Common2)
Firewalls (Common2)
Firewalls (Common2)
State Estimation
Aun1,Aun2,Aun3,Aun6,Aun8,Aur1,Aur3,Aur6,
Aur7,P2,P3,In5,In6,In7,In9,Av2,Av4,Av5,Av8,
Av10,Common3,N3,Common1
Spoofing
Eavesdropping
Replay Attack
Data Tampering
DOS
Malware injection
False data injection
Aun2,Aun4,Aun8,Aur2,Aur3,
Aur4,Aur5,Aur6,Aur7,P1,P2,
In4,In5,In6,In7, In9,Av1,Av2,
Av3,Av4,Av5,Av8,Av9,Av10,
Common3,N1,N3,Common1
Spoofing
Eavesdropping
Replay Attack
Data Tampering
DOS
Malware injection
False data injection
Block3
Control Centre/ SAS (Substations Automation Systems)
Distribution Automation devices
Smart Ring Main Unit (Smart RMU)
PLC
RTU
Switchgear
Capacitors
Actuators
Transformers
Voltage regulators
IED (Intelligent Electronic Device)
Controllers
Breakers
Distribution Substations
- DMS (Data Management System)
Block2
Control Centre/ SAS (Substations Automation Systems)
RTU
AMR (Advance Metering Reading)
Transmission Automation devices
Transmission Stations
- EMS (Energy Management System)
- Electricity operation system
- Decision Making
State Estimation
Smart Meter
Smart Meter
Block 1
Control Centre
Renewables
Bulk Generation Plants
- SIS (Supervisory Information System)
- DRM (Demand Response Management)
🟥 Security Threats
🟩 Security Controls
🟦 Information flow
Information Systems

Figure 5-2: Confirmed Information Security Model for the IoT-enabled Smart Grid

Chapter 6 Formal Verification and Validation of the Proposed Model

In Chapter 5, the overall Information Security Model of the IoT-enabled Smart Grid was confirmed by experts' interviews, literature reviews, and threat analysis. This proposed model is a very high-level model for a large and complex infrastructure with multiple access points and complex interactions. Using this proposed high-level model as a reference, this chapter focuses on a specific segment of this system to incorporate a more detailed analysis of threats and assess the effectiveness of the proposed controls. The purpose of this approach is to achieve two objectives:

- To demonstrate that the high-level proposed model could be a good starting point to conduct a more detailed and tangible analysis.
- To illustrate how this analysis can be approached rigorously to achieve the goal of this research, namely building a robust IoT-enabled Smart Grid system that can address the security concerns identified by the overall model.

In this chapter, a formal method called Event-B and its associated toolset, Rodin, are utilised to conduct a thorough analysis. The details of this formalism and its associated toolset is discussed in Section 2.10. The research initially engages in the formal modelling of the system's functioning and subsequently incorporates the security requirements throughout the later stages.

This chapter is structured as follows. Section 6.1 provides an overview of the targeted subsystem. Section 6.2 presents the system phases, including a high-level representation of the functional requirements for this subsystem. The state machine and Event-B represenations are also provided. Section 6.3 introduces more details about the operation of the system by refining some aspects of internal processing and states. Section 6.4 shows the refinement strategy, while section 6.5 discusses other failures. Finally, section 6.6 concludes the chapter.

6.1 Overview of the System

As stated above, to manage the system's complexity this chapter focuses on a representative segment or subsystem of the confirmed model in Figure 5-2 in Chapter 5. Concentrating on a specific part of this infrastructure provides the researcher with the opportunity to conduct a more detailed analysis of the functionality of the IoT-enabled Smart Grid. Furthermore, the nature of communication between the access points that have been chosen in this chapter is representative of many further aspects of communication between other access points in the IoT-enabled Smart Grid (see Appendix D for the analysis of AMI – Smart Meters interaction).

The system block diagram in Figure 6-1 shows a high-level overview of the selected subsystem. In this chapter, the focus is on the bi-directional flow of information and the interaction between two access points: the Station Automation System/Control Centre (SAS) and a cluster of IoT-enabled Station Automation Devices (SAD).

SAS sends commands to SAD. A command is a signal that controls the system. In response, SAD sends monitoring information, which is the signal reports status or execution result to SAS.

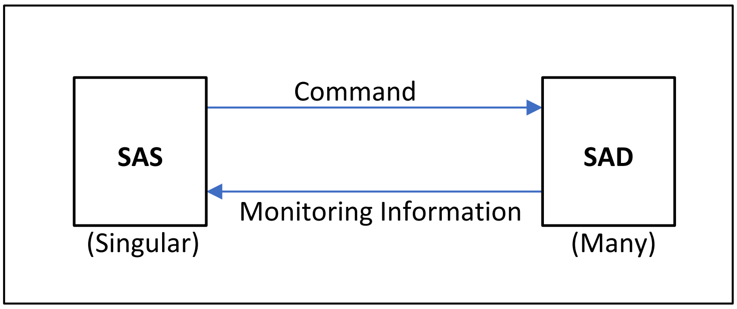


Figure 6-1: System block diagram

The system block diagram in Figure 6-1 is a high-level representation of the communication between SAS and SAD in any block including Block 1, Block 2, or Block 3 of the reference model in Figure 5-2 on Chapter 5. The reason is that the communication architecture between SAS and SAD in Block 1 is the same architecture as in Block 2 and Block 3.

The pair of SAS and SAD is selected, in this chapter, because it represents a major element of the IoT-enabled SG at which information flows and interfaces through the internet. SAS is the key access point (Stellios *et al.,* 2018) that controls and monitors electricity operations in real time via ICT networks. SAS here is an abstract representation of the control centre in the generation, transmission, and distribution access points in Block 1, Block 2, and Block 3 in the reference model in Figure 5-2 in Chapter 5.

SAD represents the IoT-enabled station automation devices in those blocks. The stations in generation, transmission, and distribution are equipped with the IoT-enabled SAD. The SAD IoT-sensors send monitoring information, whereas SAD IoT-actuators receive commands. ICT is the backbone of the communication infrastructure for transferring information between SAS and SAD access points.

6.2 System Phases

This interaction between SAS and SAD is complex, and includes many phases, aspects and types of commands, and monitoring Information. Therefore, in the interests of comprehending the whole system, interactions are grouped into three phases:

- Commissioning and registration phase
- Network configuration phase
- Core operations phase.

First, SAS manages the commissioning and registration phase for IoT-enabled SAD. Once registered, SAD is regulated and can communicate, and it is assigned to the standby state. If SAS detects overloading, the Network Configuration Phase is triggered, whereby SAS sends an Operate Command to activate SAD from standby to operational. In response, SAD executes the command and sends monitoring information to SAS. Next, the core operations phase is triggered, whereby SAS checks the monitoring information to decide the next appropriate state of SAD that should be converted to it and the command will be sent back if needed to SAD, which in turn, responds according to those commands and sends further monitoring information.

Accordingly, commands and monitoring information exchanges between SAS and SAS can be divided into those three categories. The command could be Operate Command, Release Command, Normal Command, or Corrective Command, while the monitoring information describes several conditions, including Normal Information, Transient failure, and Permanent failure, that can be sent to SAS. On the SAS side, there is a corresponding perception of the received monitoring information represented by several states/modes of SAD respectively inside SAS, including standby SAD, operational SAD, unoperational SAD, and retired SAD

As mentioned above, the state of SAD changes throughout theses phases. Figure 6-2 shows the different states of SAD that will be modelled in the formal model in this chapter, and how the state coverts from one to another.

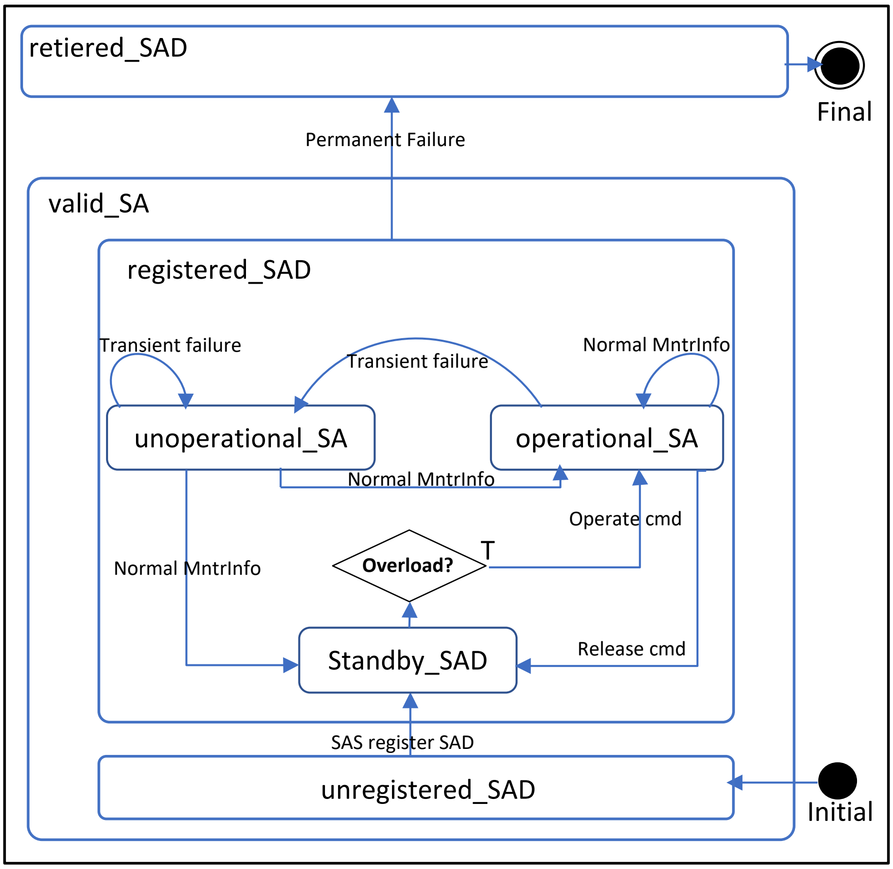


Figure 6-2: SAD states

6.2.1 Commissioning and Registration Phase

This function is managed by the SAS control centre, which acts as the operations controller. According to NISTIR 7628, this phase is initiated by the installer that powers “ON” the IoT-enabled SAD and follows the manufacturer’s instructions. Once the SAD device has completed the commissioning phase and is present on the network, it sends an admission request to SAS through the communication Channel (NISTIR 7628, 2014), as shown in FUN-1 in Table 6-1. SAS system should accept admission requests only from valid IoT-SAD devices with pre-identified manufacturing ID numbers, approved and supplied by the Utility (NISTIR 7628, 2014). Also, SAS should not re-register already registered SAD devices.

It is worth highlighting that the admission and registration phase is a prerequisite for any other function of the whole system of the IoT-enabled Smart Grid. Indeed, the commissioning and registration phase is a vital phase in terms of security, as only authenticated devices are allowed to communicate (send/receive) information with other access points over the IoT-enabled Smart Grid. For this reason, the admission requests included in the commissioning and registration phase are highlighted separately from monitoring information or commands exchanged between SAS and SAD in the network. For the same reason, NISTIR 7628 discusses the commissioning and registration phase separately in an in-depth look at security and privacy concerns (sections 5.6 and 5.6.3 of the referenced source, NISTIR 7628, 2014).

- **Functional Requirements of the Commissioning and Registration Phase**

The construction of the formal model of the system's functionality is built based on the functional requirements of the system, taking into consideration the following aspects:

1. The available information on the system functionality used in this formal model was gleaned from the experts' review records by Expert3, Expert4, Expert5, Expert8, Expert10, Expert11, Expert12, and Expert13 (see transcripts in Appendix D and Chapter 5).
2. The available information on the system functionality used in this formal model is augmented by NIST (NISTIR 7628, 2014), as well as published reports by authorised bodies such as the Saudi Electricity Company (2019), Marcelo *et al.* (2013), DEO (U.S. Department of Energy, 2012), and the Saudi ECRA report (Marcelo *et al.*, 2013).
3. This formal model focuses on the information flow between devices, not how each device (access point) operates. It shows related operations that affect the targeted system behaviour, interaction, and information flow. Also, the scope of this research is not concerned with the internal architecture of SAS and SAD, nor any function describing how the power is generated or stored, including net metering, plug-in electric vehicles, and Distributed Energy Resources (DER).
4. This research is concerned with general types of commands and monitoring information that capture the main general functionality of the access points. This research does not make use of specific devices or access points by a specific standard or manufacturer.

Table 6-1 shows the detailed functional specifications of the commissioning and registration phase:

Table 6-1: Functional requirements of the commissioning and registration phase

Serial No.	**Functional Requirement**
FUN-1	The valid SAD sends an admission request , through the communication channel, to SAS to join the network
FUN-2	The SAS system receives the admission request from the channel
FUN-2.1	If the registration succeeds; that is, it is a valid and unregistered SAD, SAS registers the valid SAD and lists it to standby SAD
FUN-2.2	If the registration fails; that is, the SAD not valid or was registered before, the admission request is removed

- **UML State Machine of the Commissioning and Registration Phase**

In this subsection, the Unified Modeling Language (UML) state machine is presented for the commissioning and registration phase. This consists of four SAD states as in Figure 6-3, in which standby_SAD and registered_SAD is nested states as shown in Figure 6-2.

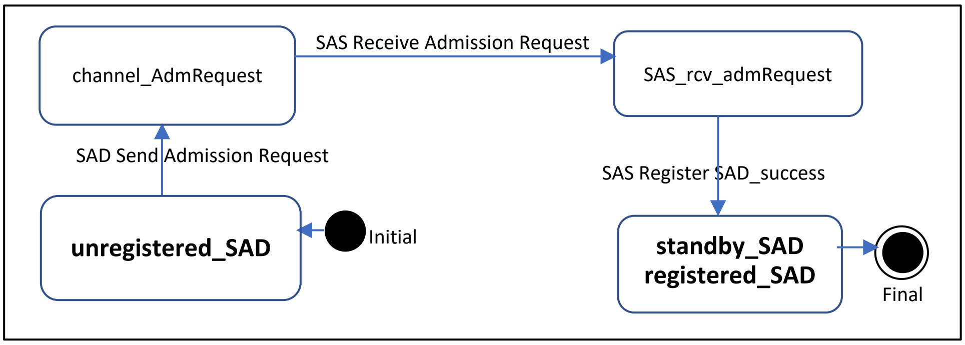


Figure 6-3: UML State machine of the commissioning and registration process

- **Formal Model of the Commissioning and Registration Phase**

This subsection shows the Event-B representation of the commissioning and registration phase, with screenshots of the Event-B codes. The formal model has the addition of some critical invariants, guards, and actions in two parts: the Context (C0) and the Machine (M0).

6.2.1..1 Context (C0)

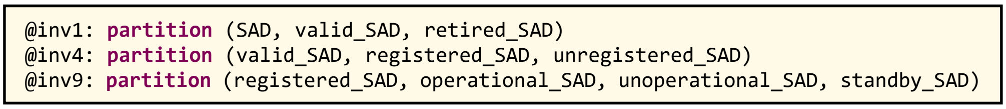
Addition of Axioms to the context element of the model to add properties to the constants and carrier sets represented therein: Axiom (@axm4) defines SAS and SAD of the type Access Point Carrier Set:

partition (AccessPoint, SAS, SAD, OTHER_AccessPoint), AccessPoint is partitioned into SAS and SAD, and an OTHER_AccessPoint that could be used in defining any other type of access point in a further refinement, supporting the flexibility and scalability of the formal model.

6.2.1..2 Machine (M0)

In the formal model of the system functionality, SAS is modelled as a singular that manages and communicates with a multitude of SAD devices. Therefore, Machine M0 defines SADs as a variable (registered_SAD) by the invariant (@inv4), as shown in Code 6-1:

partition (valid_SAD, registered_SAD, unregistered_SAD)



```
@inv1: partition (SAD, valid_SAD, retired_SAD)
@inv4: partition (valid_SAD, registered_SAD, unregistered_SAD)
@inv9: partition (registered_SAD, operational_SAD, unoperational_SAD, standby_SAD)
```

Code 6-1: Invariants of the registration process

In which SAD is partitioned according to the invariant (@inv1) to a valid set and a retired set of SADs:

@inv1: **partition** (SAD, valid_SAD, retired_SAD). This partition means that valid_SAD ∩ retired_SAD = ∅

After SAS receives admission request, the registration process has two events, one for successful registration and a second for failed registration. The following events shown in Code 6-2 formalise the actions of the registration process according to the functional requirements in Table 6-1.

If the registration succeeds, SAD is registered and added to the standby list of SADs:

standby_SAD ≔ standby_SAD ∪ {sad}. Where standby_SAD is defined by the partition:

@inv9: **partition** (registered_SAD, operational_SAD, unoperational_SAD, standby_SAD)

So, the device is in the standby state until it is activated in the event of overloading, as discussed next in the network configuration phase.

```
event SAD_Send_Admission_Request  //correspond to FUN-1 in Table 6-1
any
   sad
where
   @grd1: sad ∈ unregistered_SAD
   @grd3: sad ∉ channel_AdmRequest
   @grd5: sad ∉ SAS_rcv_admRequest
then
   @act1: channel_AdmRequest ≔ channel_AdmRequest ∪ {sad}
end
event SAS_Receive_Admission_Request //correspond to FUN-2 in Table 6-1
any
   sad
where
   @grd1: sad ∈ unregistered_SAD
   @grd3: sad ∈ channel_AdmRequest
then
   @act1: SAS_rcv_admRequest ≔ SAS_rcv_admRequest ∪ {sad}
   @act4: channel_AdmRequest ≔ channel_AdmRequest \ {sad}
end
event SAS_RegSAD_Success //correspond to FUN-2.1 in Table 6-1
   any
      sad
   where
      @grd1: sad ∈ unregistered_SAD
      @grd2: sad ∈ SAS_rcv_admRequest
      @grd3: sad ∉ channel_AdmRequest
   then
      @act1: registered_SAD ≔ registered_SAD ∪ {sad}
      @act2: standby_SAD ≔ standby_SAD ∪ {sad}
      @act3: SAS_rcv_admRequest ≔  SAS_rcv_admRequest \ {sad}
      @act4: unregistered_SAD ≔ unregistered_SAD \ {sad}
   end
   event SAS_RegSAD_Fail //correspond to FUN-2.2 in Table 6-1
   any
      sad
   where
      @no-double-registered: sad ∈ registered_SAD ∨ sad ∉ valid_SAD
      @grd3: sad ∈ SAS_rcv_admRequest
   then
      @act2: SAS_rcv_admRequest ≔  SAS_rcv_admRequest \ {sad}
   end
```

Code 6-2: Registration events

If registration fails, SAS does not register the SAD and the received registration request is ignored. Registeration fails if SAD is registered previously, or not of the valid list of devices. The registration failure case is modelled due to security checks which will be discussed later in Chapter 7 section section 7.4.3.

6.2.2 **Network Configuration Phase**

This phase occurs in the event of overloading, when SAS converts SAD from the standby state to the operational state, whereby the Operate command is sent to SAD. Also, this phase occurs in the event of Oversupplying, when SAS converts SAD from the operational state to the standby state, whereby the Release command is sent to SAD.

SAS detects overloading or oversupplying of the whole Smart Grid using SCADA Fault Passage Indicators (FPI) or relays that are installed in Generation Bulk stations, Transmission stations, and Distribution substations. This phase contains two configurations:

1. SAD Activation (Operate command): If SAS detects an overload, SAS sends the Operate command to the standby SAD to convert it to operational SAD, as shown in FUN-5 in Table 6-4 (Mavridou and Papa, 2012; Marcelo *et al.*, 2013; Penghou, 2018). The standby state means that SAD device is valid and registered, but is not yet operational until overloading is detected.

   As a result of the SAD activation phase, the device is either activated and become an operational SAD if the activation succeeds or is converted to unoperational SAD or retired SAD if the activation fails due to transient or permanent failure.
2. SAD Deactivation (Release command): If SAS detects oversupply, SAS sends the Release command to convert operational SAD to standby SAD, as shown in FUN-13 in Table 6-5 (Mavridou and Papa, 2012; Marcelo *et al.*, 2013; Penghou, 2018).

The commands that are sent in the network configuration phase are the Operate, Release, Normal, and Corrective commands, and Table 6-2 summarises the classification of commands (NISTIR 7628, 2014; Narayanan et al., 2018; electrical technology, 2020).

Table 6-2: Command classification

Command Category	**Definition**
Normal	Acknowledgement of the received normal monitoring Information as an indication to the SAD device to carry on in its state
	Command to fetch the production information required to run the operations of the IoT-enabled SG, such as retrieving Information required for trend analysis, state estimation, or comprehensive reports
Corrective	Corrective signals in response to transient failures such as Acknowledge_powerON, Troubleshooting_Cmd, Reboot_Cmd, Acknowledge_finishPrev, and Acknowledge_unlock
Operate	SAS sends the Operate command to the standby IoT-SAD in the event of overloading
Release	SAS sends the Release command to convert operational IoT-SAD to standby IoT-SAD in the event of oversupply

The monitoring information that could be sent in this phase in response is summarised and defined in Table 6-3. Transient failure triggers SAS to send a Corrective command in response, such as troubleshooting and upgrading commands. Otherwise, SAS sends a Normal command.

Table 6-3: Monitoring information classification

Category	**Definition**
Normal information	Normal monitoring Information means normal status for SAD, or normal information related to power information required to run the operations of the IoT-enabled SG, such as electric measurements and current/voltage magnitudes, phase angles, and frequency, or comprehensive reports
	Successful execution of the previously received command, without failure
Transient failure	An operational failure that causes failed execution of the command, because SAD has a temporary failure in its status, such as malfunction or overheating, which converts the IoT-SAD device to an unoperational state for a time until the failure is resolved. Then IoT-SAD devices return to the operational state and send normal information as monitoring information
Permanent failure	An operational failure that causes failed execution for the Command because SAD has a failure that is not temporary but persistent such as memory failure or life span expiration of the IoT-SAD device

- **Functional Requirements of the Network Configuration Phase (SAD Activation)**

In the network configuration phase, Table 6-4 shows the detailed functional specifications for SAD activation:

Table 6-4: Functional requirements of the network configuration phase (SAD activation)

Serial No.		**Functional Requirements**
FUN-3		If SAS detects overload, the SAS system sends the Operate command to a standby SAD, through the communication channel
Communication Pattern	FUN-4	SAD receives the command from the channel
	FUN-5	SAD executes the command and sends the execution result as monitoring information to SAS through the channel
	FUN-6	SAS system receives the monitoring Information from SAD through the channel
FUN-7		SAS checks the monitoring information of the Operate command execution
FUN-7.1		If it succeeds; that is, monitoring information is Normal information, SAS converts the standby SAD to operational SAD and sends the Normal command to the channel
FUN-7.2		If the execution fails because of a Transient failure, SAS converts the standby SAD to an unoperational SAD and sends the Corrective command to the channel
FUN-7.3		If the execution fails because of a Permanent failure, SAS deregisters the standby SAD and converts it to retired SAD
Communication Pattern	FUN-8.1	SAD receives the command from the channel
	FUN-8.2	SAD executes the command and sends the execution result as monitoring information to the SAS through the channel
	FUN-8.3	SAS system receives the monitoring information from SAD through the channel

- **Functional Requirements of the Network Configuration Phase (SAD Deactivation)**

Table 6-5 shows the detailed functional specifications of the network configuration phase (SAD deactivation):

Table 6-5: Functional requirements of the network configuration phase (SAD deactivation)

Serial No.		**Functional Requirements**
FUN-9		If SAS detects oversupply, the SAS system sends the Release command to the operational SAD through the communication channel to convert it to standby SAD
Communication Pattern	FUN-10	SAD receives the command from the channel
	FUN-10.1	SAD executes the command and sends the execution result as monitoring information to SAS through the channel
	FUN-10.2	SAS system receives the monitoring information from SAD through the channel
FUN-11		SAS checks the monitoring information for the execution of the Release command
FUN-11.1		If the execution succeeds; that is, Normal information, SAS converts the operational SAD to standby SAD
FUN-11.2		If the execution fails; that is, there is Transient failure, SAS converts the operational SAD to unoperational SAD and sends the Corrective command to the channel
Communication Pattern	FUN-12	SAD receives the command from the channel
	FUN-12.1	SAD executes the command and sends the execution result as monitoring information to SAS through the channel.
	FUN-12.2	SAS system receives the monitoring information from SAD through the channel
FUN-12.3		The core operations phase is performed cyclically

- **UML State Machine of the Network Configuration Phase (SAD Activation and Deactivation)**

SAS keeps track of all registered SAD,s besides all retired SADs. So, in the event of permanent failure, SAS manages the de-registration process that converts SAD to retired SAD and removes it from the registered and valid set of devices. Permanent failure includes lifespan expiration, memory failure, and if troubleshooting fails and SAD must be returned to the manufacturer to be fixed, as shown in FUN-7.3 in Table 6-4. This phase consists of four states for SAD (standby_SAD, operational_SAD, unoperational_SAD, and retired_SAD) as shown in Figure 6-4.

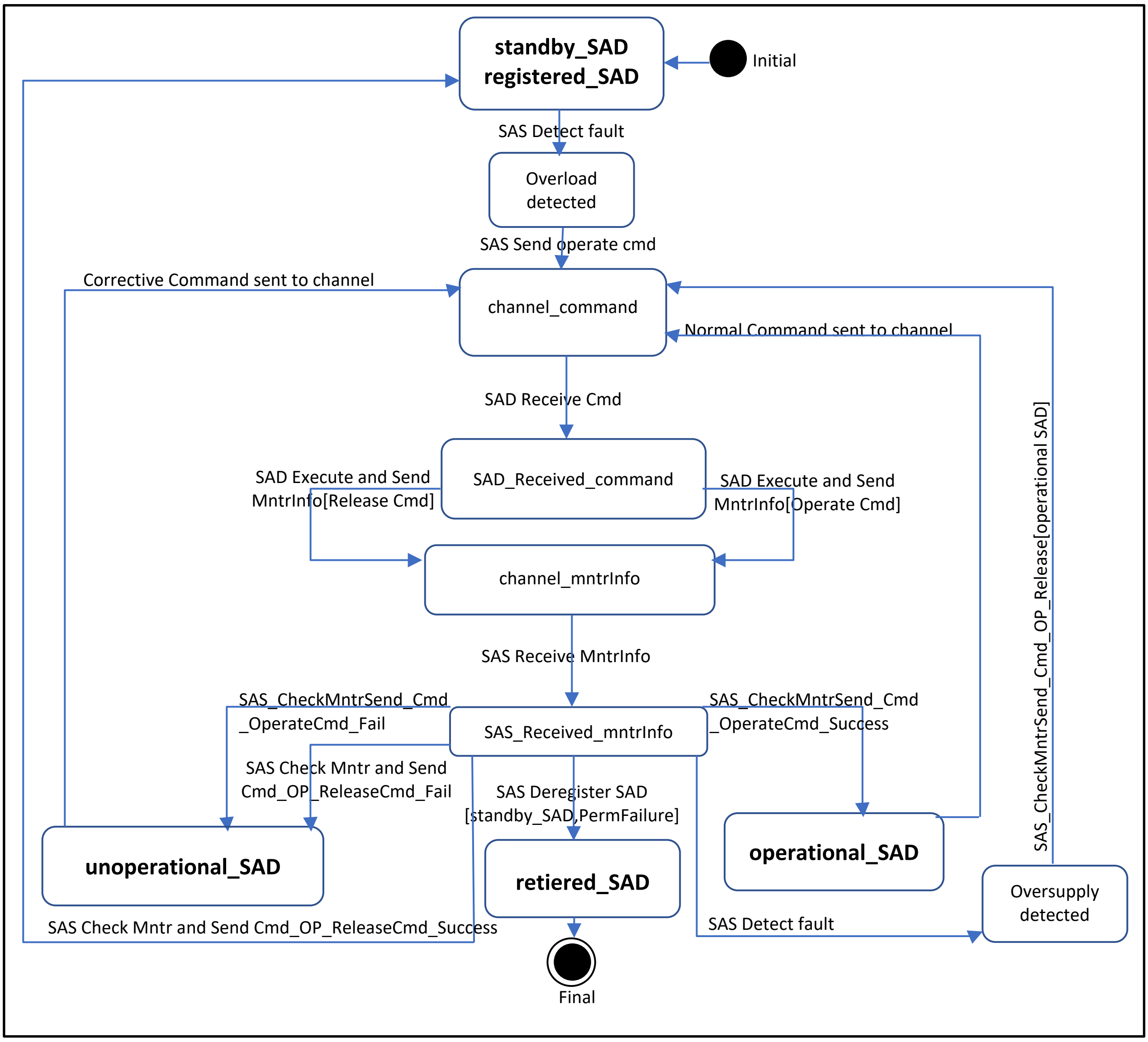
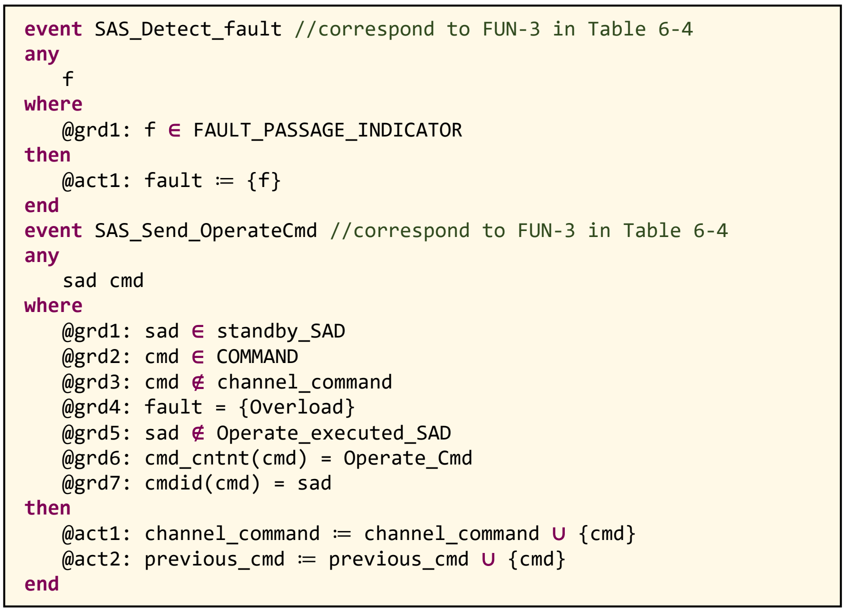

standby_SAD
registered_SAD
Initial
SAS Detect fault
Overload detected
SAS Send operate cmd
Corrective Command sent to channel
channel_command
Normal Command sent to channel
SAD Receive Cmd
SAD Execute and Send MntrInfo[Release Cmd]
SAD_Received_command
SAD Execute and Send MntrInfo[Operate Cmd]
channel_mntrInfo
SAS Receive MntrInfo
SAS_CheckMntrSend_Cmd _OperateCmd_Fail
SAS_Received_mntrInfo
SAS_CheckMntrSend_Cmd _OperateCmd_Success
SAS Check Mntr and Send Cmd_OP_ReleaseCmd_Fail
SAS Deregister SAD [standby_SAD,PermFailure]
SAS_CheckMntrSend_Cmd_OP_Release[operational SAD]
unoperational_SAD
retiered_SAD
operational_SAD
Oversupply detected
SAS Check Mntr and Send Cmd_OP_ReleaseCmd_Success
SAS Detect fault
Final

Figure 6-4: UML state machine of the network configuration process (SAD activation and deactivation)

- **Formal Model of the Network Configuration Phase (SAD Activation and Deactivation)**

SAS detects overloading/oversupplying through the event (**event** SAS_Detect_fault) shown in Code 6-3, where f is a parameter of type FAULT_PASSAGE_INDICATOR carrier set that is partitioned in the context axiom: @axm12: **partition** (FAULT_PASSAGE_INDICATOR, {Overload}, {Oversupply}):



```
event SAS_Detect_fault //correspond to FUN-3 in Table 6-4
any
   f
where
   @grd1: f ∈ FAULT_PASSAGE_INDICATOR
then
   @act1: fault ≔ {f}
end
event SAS_Send_OperateCmd //correspond to FUN-3 in Table 6-4
any
   sad cmd
where
   @grd1: sad ∈ standby_SAD
   @grd2: cmd ∈ COMMAND
   @grd3: cmd ∉ channel_command
   @grd4: fault = {Overload}
   @grd5: sad ∉ Operate_executed_SAD
   @grd6: cmd_cntnt(cmd) = Operate_Cmd
   @grd7: cmdid(cmd) = sad
then
   @act1: channel_command ≔ channel_command ∪ {cmd}
   @act2: previous_cmd ≔ previous_cmd ∪ {cmd}
end
```

Code 6-3: SAS sends Operate command to avtivate SAD

- **Channel buffer of commands**

If SAS detects overloading @grd4: fault = {Overload}, it sends the Operate command (Operate_Cmd) to the standby_SAD through the channel. Therefore, the variable channel_command acts as a buffer that synchronises commands from the SAS side to the SAD side: @act1: channel_command ≔ channel_command ∪ {cmd}, where cmd is a parameter of type COMMAND carrier set.

- **Use of records structures**

The command is structured as a record in event B that has many properties, such as the ID of the SAD that the command was sent to and the content of that command, which is the type of that command. See Figure 6-5.

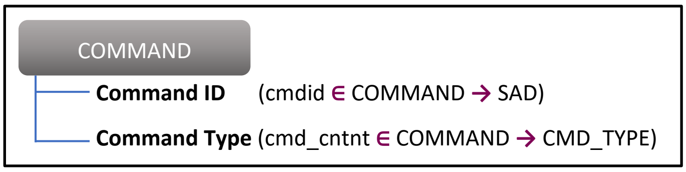

COMMAND
Command ID (cmdid ∈ COMMAND → SAD)
Command Type (cmd_cntnt ∈ COMMAND → CMD_TYPE)

Figure 6-5: Command record

where (cmdid) and (cmd_cntnt) are defined by the following axioms in the context C0:

@axm18: cmdid ∈ COMMAND → SAD, @axm19: cmd_cntnt ∈ COMMAND → CMD_TYPE

where CMD_TYPE is partitioned by the following axiom:

@Valid_Command: **partition** (CMD_TYPE, {Normal_Cmd}, {Operate_Cmd}, {Release_Cmd}, Corrective_Cmd)

In the machine element of the model, the below guards are added to specify the type of the command and the ID of SAD that the command sent to, the guard @grd6 assigned the type to the Operate Command because this event is for SAD activation: @grd6: cmd_cntnt(cmd) = Operate_Cmd, @grd7: cmdid(cmd) = sad Afterwards, SAD will receive the Command from the channel in the variable SAD_received_command as shown in the event (**event** SAD_Receive_Cmd) in Code 6-4.

Then, in the event (**event** SAD_ExecuteSend_MntrInfo_OperateCmd), SAD checks that the received command is (Operate_Cmd) and checks the ID is correct by the following guards:

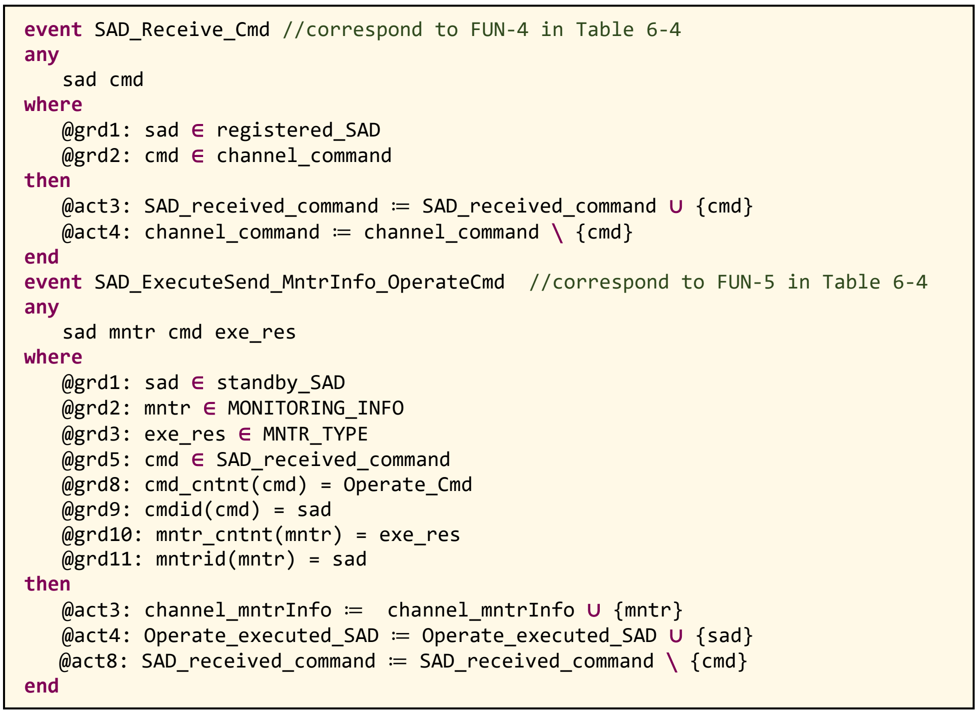
@grd5: cmd ∈ SAD_received_command,   @grd8: cmd_cntnt(cmd) = Operate_Cmd, @grd9: cmdid(cmd) = sad

In this event, SAD executes the Operate command and sends the execution result back to SAS in the action:

@act3: channel_mntrInfo ≔ channel_mntrInfo ∪ {mntr}, where:

- **mntr** is a parameter of type MONITORING_INFO carrier set
- **channel_mntrInfo** is the sync buffer of the Monitoring Information in the channel
- **mntr_cntnt** is the execution result (exe_res) of type MNTR_TYPE which is partitioned by the following axiom that describes the valid Monitoring Information:

@Valid_Mntr: **partition** (MNTR_TYPE, {Normal_info}, Permanent_failure, Transient_failure, OTHER_MNTR_TYPE)



```
event SAD_Receive_Cmd //correspond to FUN-4 in Table 6-4
any
    sad cmd
where
    @grd1: sad ∈ registered_SAD
    @grd2: cmd ∈ channel_command
then
    @act3: SAD_received_command ≔ SAD_received_command ∪ {cmd}
    @act4: channel_command ≔ channel_command \ {cmd}
end
event SAD_ExecuteSend_MntrInfo_OperateCmd  //correspond to FUN-5 in Table 6-4
any
    sad mntr cmd exe_res
where
    @grd1: sad ∈ standby_SAD
    @grd2: mntr ∈ MONITORING_INFO
    @grd3: exe_res ∈ MNTR_TYPE
    @grd5: cmd ∈ SAD_received_command
    @grd8: cmd_cntnt(cmd) = Operate_Cmd
    @grd9: cmdid(cmd) = sad
    @grd10: mntr_cntnt(mntr) = exe_res
    @grd11: mntrid(mntr) = sad
then
    @act3: channel_mntrInfo ≔  channel_mntrInfo ∪ {mntr}
    @act4: Operate_executed_SAD ≔ Operate_executed_SAD ∪ {sad}
   @act8: SAD_received_command ≔ SAD_received_command \ {cmd}
end
```

Code 6-4: Command reception and execution events by SAD

In Event B, monitoring information is also structured as a record that has the ID of the SAD that sends it, and the content of that monitoring information, which is the monitoring type. See Figure 6-6.

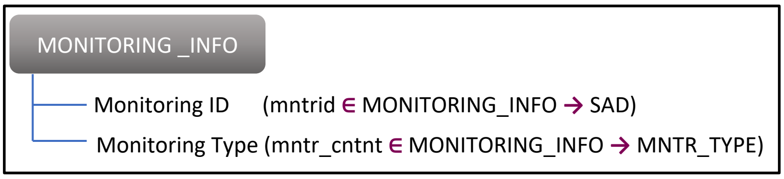
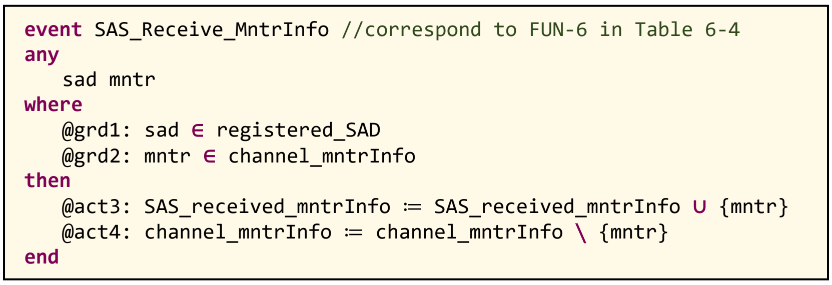

MONITORING _INFO
Monitoring ID   (mntrid ∈ MONITORING_INFO → SAD)
Monitoring Type (mntr_cntnt ∈ MONITORING_INFO → MNTR_TYPE)

Figure 6-6: Monitoring information record

Afterwards, SAS receives the monitoring information in the variable SAS_received_mntrInfo after checking that this monitoring information is received from registered SAD as stated in the guard (@grd1), as shown in the following event in Code 6-5:



```
event SAS_Receive_MntrInfo //correspond to FUN-6 in Table 6-4
any
   sad mntr
where
   @grd1: sad ∈ registered_SAD
   @grd2: mntr ∈ channel_mntrInfo
then
   @act3: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∪ {mntr}
   @act4: channel_mntrInfo ≔ channel_mntrInfo ∖ {mntr}
end
```

Code 6-5: Monitoring information reception event by SAS

Then, SAS checks the received monitoring information of the Operate_Cmd execution. This could be successful or failure, as follows shown in Code 6-6, Code 6-7, and Code 6-8:

1- **event** SAS_CheckMntrSend_Cmd_OperateCmd_Success: the below guards check that the received monitoring information is Normal:

   @grd2: mntr ∈ SAS_received_mntrInfo

   @grd4: mntr_cntnt(mntr) = Normal_info

Then, SAS sends the Normal_Cmd as specified in the guard @grd7: cmd_cntnt(cmd) = Normal_Cmd, and add SAD to the operational list of SADs as per the following actions:

   @act1: operational_SAD ≔ operational_SAD ∪ {sad}

   @act2: channel_command ≔ channel_command ∪ {cmd}

2- **event** SAS_CheckMntrSend_Cmd_OperateCmd_Fail: the below guards check that the received Monitoring Info is Transient failure:

   @grd2: mntr ∈ SAS_received_mntrInfo

   @grd5: mntr_cntnt(mntr) ∈ Transient_failure

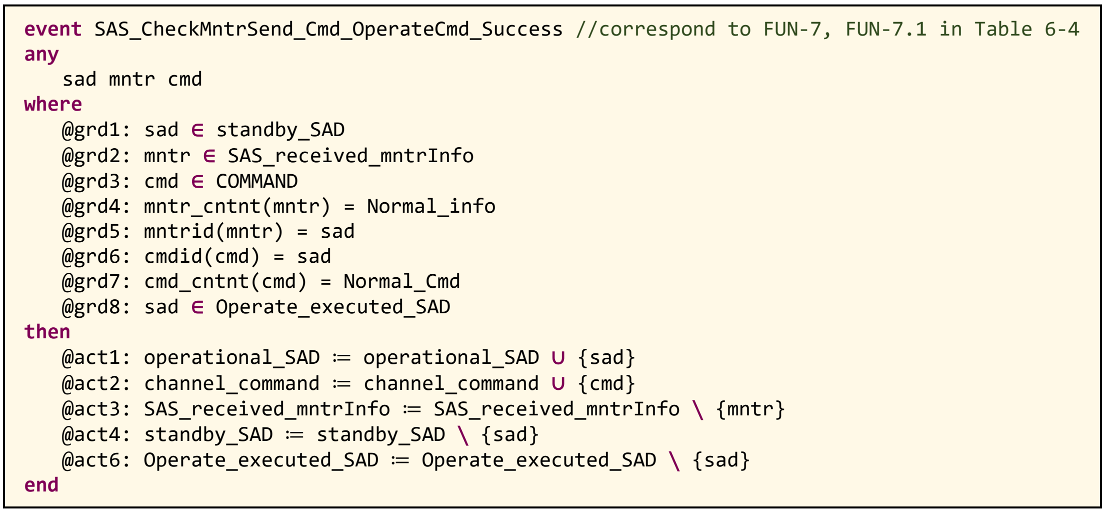
Then, SAS sends the Corrective_Cmd as specified by the guards:

   @grd3: corCmd ∈ Corrective_Cmd,   @grd8: cmd_cntnt(cmd) = corCmd

Also, SAS adds the SAD to the unoperational list of SADs as per the following actions:

   @act1: unoperational_SAD ≔ unoperational_SAD ∪ {sad}

   @act3: channel_command ≔ channel_command ∪ {cmd}



```
event SAS_CheckMntrSend_Cmd_OperateCmd_Success //correspond to FUN-7, FUN-7.1 in Table 6-4
any
   sad mntr cmd
where
   @grd1: sad ∈ standby_SAD
   @grd2: mntr ∈ SAS_received_mntrInfo
   @grd3: cmd ∈ COMMAND
   @grd4: mntr_cntnt(mntr) = Normal_info
   @grd5: mntrid(mntr) = sad
   @grd6: cmdid(cmd) = sad
   @grd7: cmd_cntnt(cmd) = Normal_Cmd
   @grd8: sad ∈ Operate_executed_SAD
then
   @act1: operational_SAD ≔ operational_SAD ∪ {sad}
   @act2: channel_command ≔ channel_command ∪ {cmd}
   @act3: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∖ {mntr}
   @act4: standby_SAD ≔ standby_SAD ∖ {sad}
   @act6: Operate_executed_SAD ≔ Operate_executed_SAD ∖ {sad}
end
```

Code 6-6: Successful execution of the Operate command

```
vent SAS_CheckMntrSend_Cmd_OperateCmd_Fail //correspond to FUN-7.2 in Table 6-4
any
   sad mntr cmd corCmd
where
   @grd1: sad ∈ standby_SAD
   @grd2: mntr ∈ SAS_received_mntrInfo
   @grd3: corCmd ∈ Corrective_Cmd
   @grd4: cmd ∈ COMMAND
   @grd5: mntr_cntnt(mntr) ∈ Transient_failure
   @grd6: mntrid(mntr) = sad
   @grd7: cmdid(cmd) = sad
   @grd8: cmd_cntnt(cmd) = corCmd
   @grd9: sad ∈ Operate_executed_SAD
then
   @act1: unoperational_SAD ≔ unoperational_SAD ∪ {sad}
   @act2: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∖ {mntr}
   @act3: channel_command ≔ channel_command ∪ {cmd}
   @act4: standby_SAD ≔ standby_SAD ∖ {sad}
   @act5: Operate_executed_SAD ≔ Operate_executed_SAD ∖ {sad}
end
```

Code 6-7: Fail execution of the Operate command due to Transient failure

3- **event** SAS_DeregisterSAD_Standby_PermFailure: the below guards check that the received monitoring information is Permanent_failure: @grd2: mntr ∈ SAS_received_mntrInfo

@grd4: mntr_cntnt(mntr) = Permanent_failure

Then, SAS adds SAD to the retired list of SADs and removes it from the standby and registered list :

@act1: operational_SAD ≔ operational_SAD ∪ {sad}

@act2: channel_command ≔ channel_command ∪ {cmd}

```
event SAS_DeregisterSAD_Standby_PermFailure //correspond to FUN-7.3 in Table 6-4
any
   sad mntr
where
   @grd1: sad ∈ standby_SAD
   @grd2: mntr ∈ SAS_received_mntrInfo
   @grd4: mntr_cntnt(mntr) ∈ Permanent_failure
   @grd5: mntrid(mntr) = sad
then
   @act1: retired_SAD ≔ retired_SAD ∪ {sad}
   @act2: standby_SAD ≔ standby_SAD ∖ {sad}
   @act3: registered_SAD ≔ registered_SAD ∖ {sad}
   @act4: valid_SAD ≔ valid_SAD ∖ {sad}
   @act5: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∖ {mntr}
   @act9: Operate_executed_SAD ≔ Operate_executed_SAD ∖ {sad}
   @act10: Release_executed_SAD ≔ Release_executed_SAD ∖ {sad}
   @act11: AllCmd_executed_SAD ≔ AllCmd_executed_SAD ∖ {sad}
end
```

Code 6-8: Deregistration event for SAD due to Permanent failure

In the case of Deactivation, similar aspects are modelled as for the Release command, using the same events for receiving the command and monitoring information (SAD_Receive_Cmd) and (SAS_Receive_MntrInfo). The Event B code for the Release command part is available in Appendix D.

6.2.3 Core Operations Phase

As the title suggests, this phase represents the core functionality of the network, where there are some operational SADs and the SAS is actively monitoring their states and sending appropriate commands to the operational SADs if necessary, as shown in FUN-13 in Table 6-6. Three cases are specified in FUN-13.1, FUN-13.2, and FUN-13.3 in Table 6-6 when SAS receives the monitoring information from **operational SAD.**

The monitoring information that could be sent in this phase includes Normal information, Transient failure, and Permanent failure. In response, the command that could be sent in this phase includes the Normal command and Corrective command. Definitions of those commands and monitoring information are explained in Table 6-2 and Table 6-3.

As noted, when SAS receives Normal monitoring information from **unoperational SAD**, then un-deterministically SAS converts SAD to the operational state or sends the Release command to convert SAD to standby SAD, as explained in FUN-13.4 and FUN-13.5 in Table 6-6. This non-determinism occurs because some SAD devices recover quickly from the transient failure, while others do not depending on the type of transient failure. The precise types of transient failure will be discussed in the refinement, where the non-determinism will be refined and become deterministic.

Afterwards, SAS checks the monitoring information for the execution result of the Release command from unoperational SAD, as shown in FUN-13.8 and FUN-13.9 in Table 6-6.

The aforementioned cycle of the core operations phase is iterated continuously until SAD is retired, or in the event of oversupply where the operational SAD is converted to standby state (FUN-15 in Table 6-6).

- **Functional Requirements of the Core Operations Phase**

Table 6-6 shows the detailed functional specifications of the core operations phase:

Serial No.		**Functional Requirements**
FUN-13		The SAS system checks the monitoring information of the execution result of the command from SAD
FUN-13.1		If the SAD is operational SAD and the received monitoring information is Normal Information, the SAS sends the Normal command to the channel and SAD is remains in the operational state
FUN-13.2		If the SAD is operational SAD and the received monitoring information is Transient failure, the SAS converts the operational SAD to unoperational SAD and sends the Corrective command to the channel
FUN-13.3		If the SAD is operational SAD and the received monitoring information is Permanent failure, SAS deregisters the operational SAD and converts it to retired SAD
FUN-13.4		If the SAD is unoperational SAD and the received monitoring information is Normal Information, the SAS sends the Release command to the channel and SAD remains in the unoperational state
FUN-13.5		If the SAD is unoperational SAD and the received monitoring information is Normal Information, the SAS converts the unoperational SAD to operational SAD and sends the Normal command to the channel
FUN-13.6		If the SAD is unoperational SAD and the received monitoring information is Transient failure, the SAS sends the Corrective command to the channel and SAD remains in the unoperational state
FUN-13.7		If the SAD is unoperational SAD and the received monitoring information is Permanent failure, SAS deregisters the unoperational SAD and converts it to retired SAD
FUN-13.8		If the SAD is unoperational SAD and the monitoring information for the execution of Release command is Normal information, SAS converts the unoperational SAD to standby SAD
FUN-13.9		If the SAD is unoperational SAD and the monitoring information for the execution of Release command is Transient failure, SAS sends the Corrective command to the channel and SAD remains in the unoperational state
Communication Pattern	FUN-14	SAD receives the command from the channel
	FUN-14.1	SAD executes the command and sends the execution result as monitoring information to SAS through the channel
	FUN-14.2	SAS system receives the monitoring information from SAD through the channel
FUN-15		The cycle of the Core Operations from FUN-13 to 13.9 is performed repeatedly until terminated in the event of deregistration, where SAD is converted to the retired state, or in the event of oversupply where the operational SAD is converted to standby state

Table 6-6: functional requirements of the core operations phase

- **UML State Machine of the Core Operations Phase**

The core operations phase consists of four states for SAD (standby_SAD, operational_SAD, unoperational_SAD, and retired_SAD), as shown in Figure 6-7

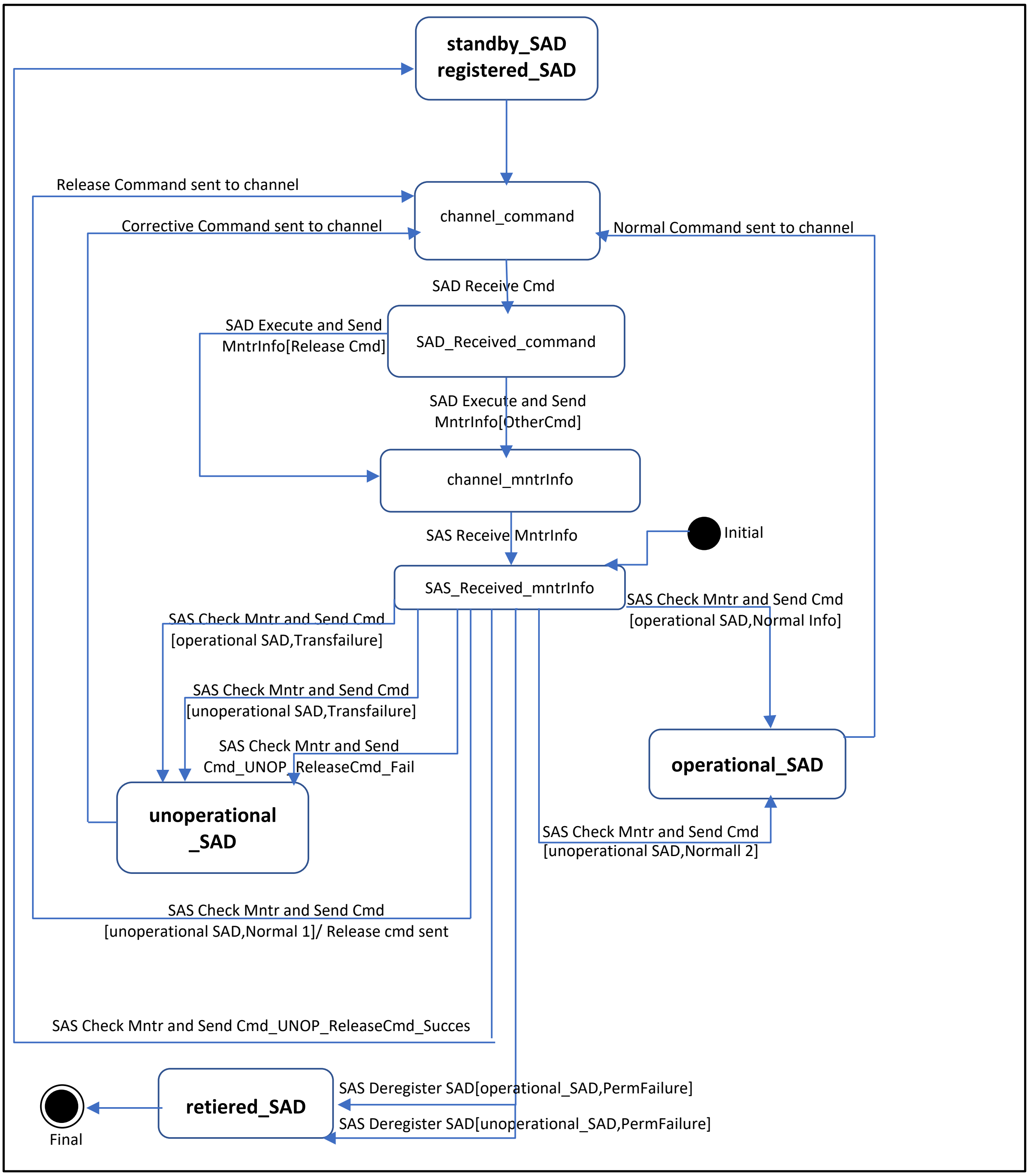

standby_SAD
registered_SAD
Release Command sent to channel
Corrective Command sent to channel
channel_command
Normal Command sent to channel
SAD Receive Cmd
SAD Execute and Send MntrInfo[Release Cmd]
SAD_Received_command
SAD Execute and Send MntrInfo[OtherCmd]
channel_mntrInfo
SAS Receive MntrInfo
Initial
SAS_Received_mntrInfo
SAS Check Mntr and Send Cmd [operational SAD,Transfailure]
SAS Check Mntr and Send Cmd [operational SAD,Normal Info]
SAS Check Mntr and Send Cmd [unoperational SAD,Transfailure]
SAS Check Mntr and Send Cmd_UNOP_ReleaseCmd_Fail
operational_SAD
unoperational _SAD
SAS Check Mntr and Send Cmd [unoperational SAD,Normall 2]
SAS Check Mntr and Send Cmd [unoperational SAD,Normal 1]/ Release cmd sent
SAS Check Mntr and Send Cmd_UNOP_ReleaseCmd_Succes
retiered_SAD
SAS Deregister SAD[operational_SAD,PermFailure]
SAS Deregister SAD[unoperational_SAD,PermFailure]
Final

Figure 6-7: UML state machine of the core operations process

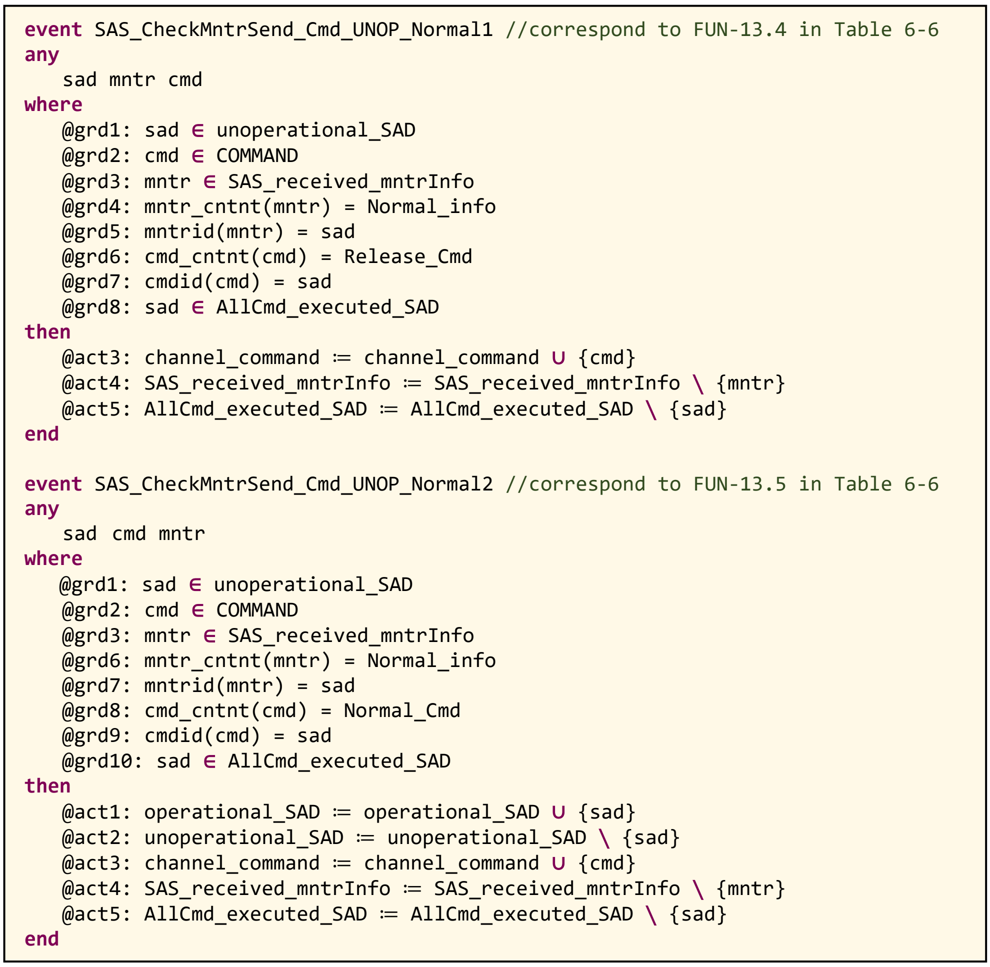
- **Formal Model of the Core Operations Phase**

Once the SAD is operational, it receives a Normal command, executes it, and sends the result back to SAS through the channel. In the formal model of the core phase, SAS checks execution results each time, then the various actions are undertaken accordingly. For instance, if the received monitoring information is Normal information and SAD is unoperational, there is a case of non-deterministic processing as follows and shown in Code 6-9:

1- Normal1: SAS sends (Release_Cmd) to SAD to be turned to standby SAD.

2- Normal2: SAS converts SAD to the operational state and sends the Normal command to SAD.

This non-determinism will be refined in the next section to provide the precise conditions under which the SAD will transition to a standby or an operational state.



```
event SAS_CheckMntrSend_Cmd_UNOP_Normal1 //correspond to FUN-13.4 in Table 6-6
any
   sad mntr cmd
where
   @grd1: sad ∈ unoperational_SAD
   @grd2: cmd ∈ COMMAND
   @grd3: mntr ∈ SAS_received_mntrInfo
   @grd4: mntr_cntnt(mntr) = Normal_info
   @grd5: mntrid(mntr) = sad
   @grd6: cmd_cntnt(cmd) = Release_Cmd
   @grd7: cmdid(cmd) = sad
   @grd8: sad ∈ AllCmd_executed_SAD
then
   @act3: channel_command ≔ channel_command ∪ {cmd}
   @act4: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∖ {mntr}
   @act5: AllCmd_executed_SAD ≔ AllCmd_executed_SAD ∖ {sad}
end

event SAS_CheckMntrSend_Cmd_UNOP_Normal2 //correspond to FUN-13.5 in Table 6-6
any
   sad cmd mntr
where
   @grd1: sad ∈ unoperational_SAD
   @grd2: cmd ∈ COMMAND
   @grd3: mntr ∈ SAS_received_mntrInfo
   @grd6: mntr_cntnt(mntr) = Normal_info
   @grd7: mntrid(mntr) = sad
   @grd8: cmd_cntnt(cmd) = Normal_Cmd
   @grd9: cmdid(cmd) = sad
   @grd10: sad ∈ AllCmd_executed_SAD
then
   @act1: operational_SAD ≔ operational_SAD ∪ {sad}
   @act2: unoperational_SAD ≔ unoperational_SAD ∖ {sad}
   @act3: channel_command ≔ channel_command ∪ {cmd}
   @act4: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∖ {mntr}
   @act5: AllCmd_executed_SAD ≔ AllCmd_executed_SAD ∖ {sad}
end
```

Code 6-9: Non-deterministic events occur when SAD recovers from unoperational state

Similarly, other events are modelled to cover all the states of SADs and monitoring information types that SAS may receive and check, and then various command types are sent, as follows:

- SAD is operational and the received Monitoring Info = Normal_Info
- SAD is operational and the received Monitoring Info = Transient failure
- SAD is operational and the received Monitoring Info = Permanent failure
- SAD is unoperational and the received Monitoring Info = Transient failure
- SAD is unoperational and the received Monitoring Info = Permanent failure

For example, the event (**event** SAS_CheckMntrSend_Cmd_OP_TransFailure) in Code 6-10 formalises the case where SAD is operational (@grd1: sad ∈ operational_SAD) and the received monitoring information indicating a Transient failure; that is:

@grd6: mntr ∈ SAS_received_mntrInfo
@grd7: mntr_cntnt(mntr) ∈ Transient_failure

Then, SAS sends Corrective_Cmd (@grd3: corCmd ∈ Corrective_Cmd) and moves SAD to the unoperational list, specified by the actions:

@act1: unoperational_SAD ≔ unoperational_SAD ∪ {sad}
@act3: channel_command ≔ channel_command ∪ {cmd}

```
event SAS_CheckMntrSend_Cmd_OP_TransFailure //correspond to FUN-13.2 Table 6-6
any
   sad cmd   mntr corCmd
where
   @grd1: sad ∈ operational_SAD
   @grd2: cmd ∈ COMMAND
   @grd3: corCmd ∈ Corrective_Cmd
   @grd6: mntr ∈ SAS_received_mntrInfo
   @grd7: mntr_cntnt(mntr) ∈ Transient_failure
   @grd8: mntrid(mntr) = sad
   @grd9: cmd_cntnt(cmd) = corCmd
   @grd10: cmdid(cmd) = sad
   @grd11: sad ∉ Release_executed_SAD
   @grd12: sad ∈ AllCmd_executed_SAD
then
   @act1: unoperational_SAD ≔ unoperational_SAD ∪ {sad}
   @act2: operational_SAD ≔ operational_SAD \ {sad}
   @act3: channel_command ≔ channel_command ∪ {cmd}
   @act4: SAS_received_mntrInfo ≔ SAS_received_mntrInfo\{mntr}
   @act7: AllCmd_executed_SAD ≔ AllCmd_executed_SAD \{sad}
end
```

Code 6-10: SAS checks monitoring information and sends command for (operational SAD, Transient failure)

So, in core operations, all events maintain a guard to check the state of SAD and that the monitoring information belongs to an authorised SAD. Moreover, a guard checks the type of monitoring information, followed by a guard to set out the consequent command type, as well as the action for converting the previous SAD state to the new corresponding state according to the type of the received monitoring information. In addition, it includes the action to send that command to SAD through the channel by the buffer channel_command. The full Event-B code is provided in Appendix D for all the above-mentioned states.

- **Communication Pattern**

As noted, each command sent from SAS to SAD is sent through the communication channel. In turn, the SAD receives that command then executes it, sending the execution result as monitoring information to SAS through the channel again.

After all that, SAS checks the monitoring information for the command that was sent initially. SAS now processes the execution result, as shown in FUN-4 to FUN-6 in Table 6-4. Figure 6-8 shows this communication pattern:

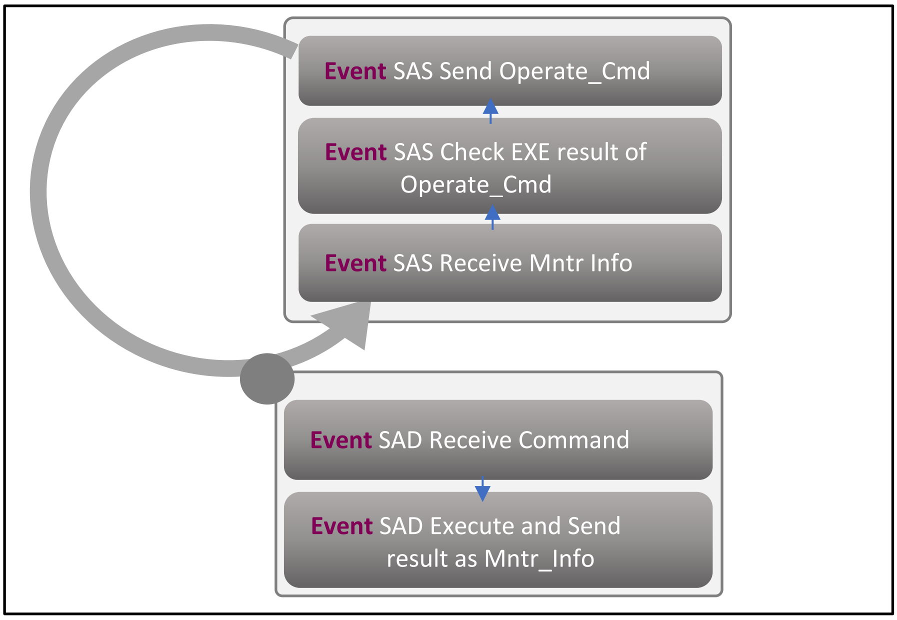

Event SAS Send Operate_Cmd
Event SAS Check EXE result of Operate_Cmd
Event SAS Receive Mntr Info
Event SAD Receive Command
Event SAD Execute and Send result as Mntr_Info

Figure 6-8: Inner and outer cycle in the communication pattern between sending the command and processing the execution result of that command

Therefore, the formal model contains the addition of variable sets that act as flags to indicate and ensure the type of the executed command, whether it is Operate/Release_Cmd in case of SAD activation/deactivation or Normal/Corrective_Cmd in case of the core operations process, as follows:

- Operate_executed_SAD: this flag indicates that Operate_Cmd is the command executed by SAD.
- Release_executed_SAD: this flag indicates that Release_Cmd is the command executed by SAD.
- AllCmd_executed_SAD: this flag indicates that Normal/Corrective_Cmd is the command executed by SAD.

6.2.4 Processing Cycles

Overall, the system phases are performed in the form of cycles. The outer cycle and inner cycle are presented in Figure 6-9. The outer cycle occurs firt for activation or deactivation of SAD devices in the network phase. The inner cycle represents the core operations process that is performed repeatedly until terminated in case of deregistration, or in case of oversupply where the operational SAD is converted to standby state.

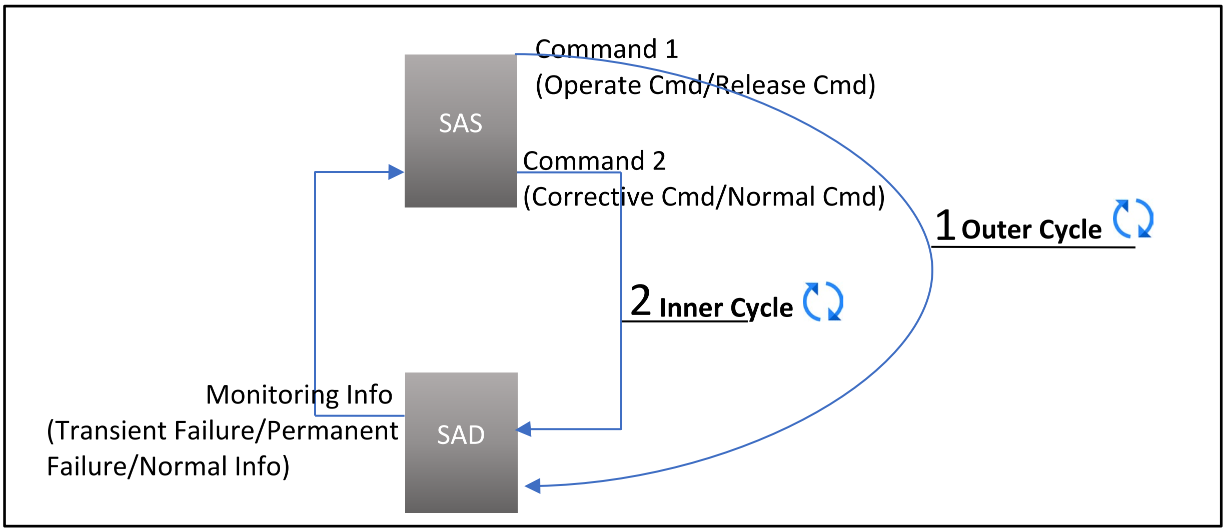


Figure 6-9: Processing cycles

6.2.5 UML State Machine Diagram of the Abstract Model

The abstract formal model for the functionality of the targeted system consists of the formal model for each phase of the system, including the commissioning and registration phase, network configuration phase, and core operations phase, as discussed in sections 6.2.1., 6.2.2, and 6.2.3. Accordingly, the state machine diagram of the abstract formal model covers all three state machine diagrams of each phase of the system. The full state machine diagram for the abstract is provided in Figure 6-10, inclusive of all three system phases.

Figure 6-11 provides the activity diagram showing other aspects of the model, based on the system events basis rather than state machine basis. Appendix D provides the full interaction diagram for detailed information flow between SAS and SAD.

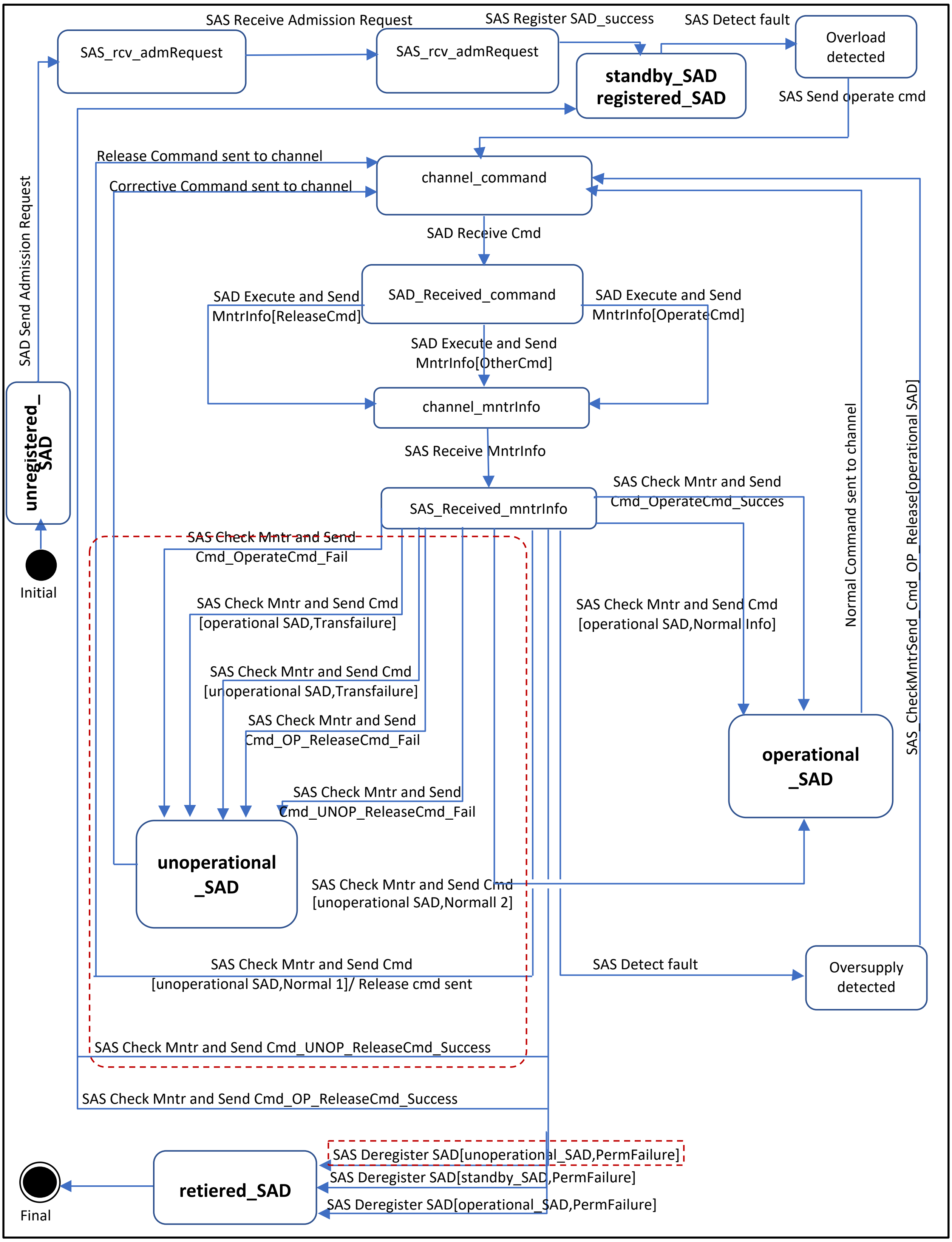

SAS Receive Admission Request
SAS Register SAD_success
SAS Detect fault
SAS_rcv_admRequest
SAS_rcv_admRequest
standby_SAD
registered_SAD
Overload detected
SAS Send operate cmd
SAD Send Admission Request
Release Command sent to channel
Corrective Command sent to channel
channel_command
SAD Receive Cmd
SAD_Received_command
SAD Execute and Send MntrInfo[ReleaseCmd]
SAD Execute and Send MntrInfo[OperateCmd]
SAD Execute and Send MntrInfo[OtherCmd]
channel_mntrInfo
SAS Receive MntrInfo
SAS_Received_mntrInfo
SAS Check Mntr and Send Cmd_OperateCmd_Succes
unregistered_SAD
Initial
SAS Check Mntr and Send Cmd_OperateCmd_Fail
SAS Check Mntr and Send Cmd [operational SAD,Transfailure]
SAS Check Mntr and Send Cmd [unoperational SAD,Transfailure]
SAS Check Mntr and Send Cmd_OP_ReleaseCmd_Fail
SAS Check Mntr and Send Cmd_UNOP_ReleaseCmd_Fail
SAS Check Mntr and Send Cmd [operational SAD,Normal Info]
Normal Command sent to channel
SAS_CheckMntrSend_Cmd_OP_Release[operational SAD]
operational _SAD
unoperational _SAD
SAS Check Mntr and Send Cmd [unoperational SAD,Normall 2]
SAS Check Mntr and Send Cmd [unoperational SAD,Normal 1]/ Release cmd sent
SAS Detect fault
Oversupply detected
SAS Check Mntr and Send Cmd_UNOP_ReleaseCmd_Success
SAS Check Mntr and Send Cmd_OP_ReleaseCmd_Success
SAS Deregister SAD[unoperational_SAD,PermFailure]
SAS Deregister SAD[standby_SAD,PermFailure]
SAS Deregister SAD[operational_SAD,PermFailure]
retiered_SAD
Final

Figure 6-10: State machine diagram of the abstract model showing the SAD states

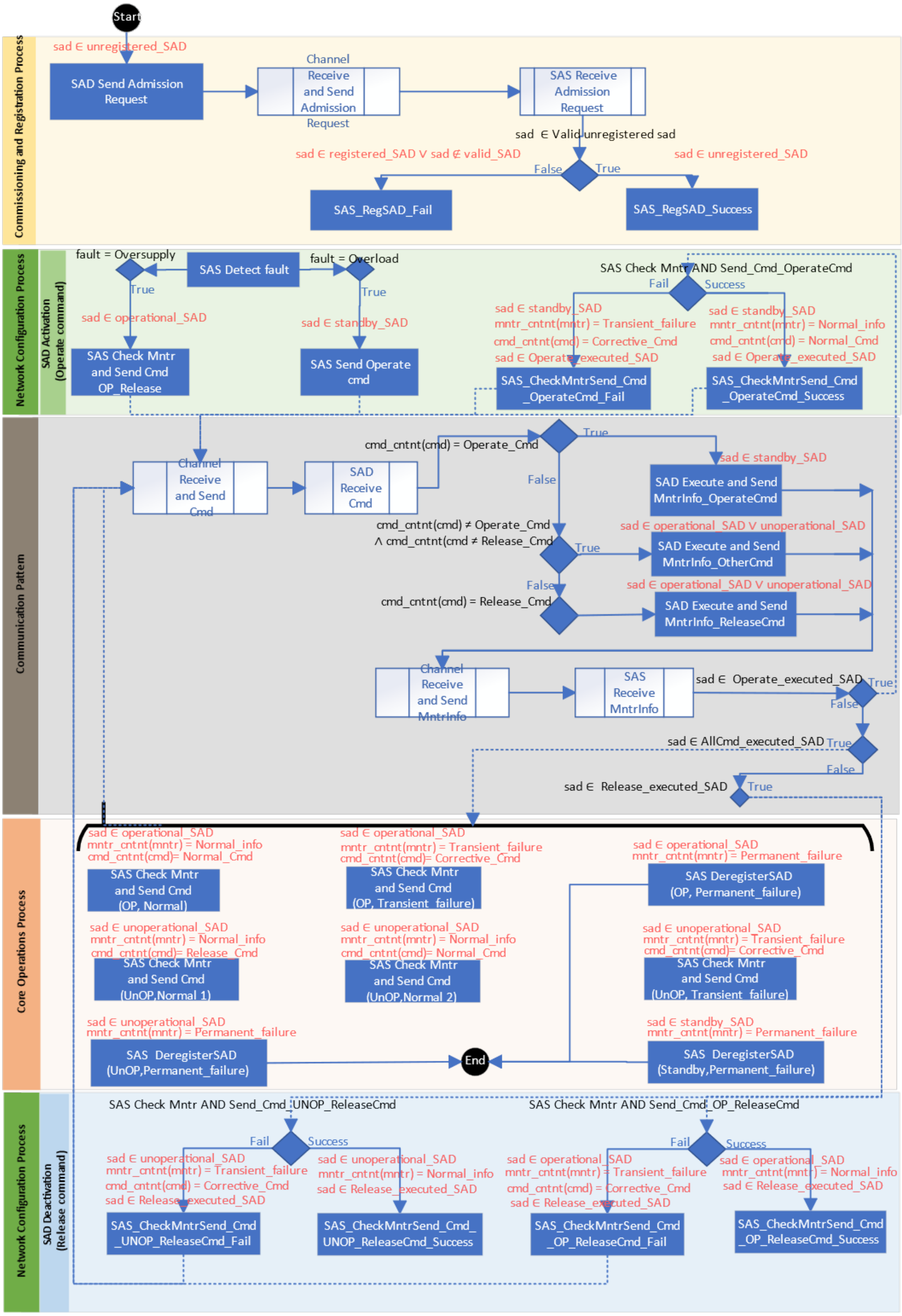

Commissioning and Registration Process
Network Configuration Process
SAD Activation (Operate command)
Communication Pattern
Core Operations Process
Network Configuration Process
SAD Deactivation (Release command)
Start
sad ∈ unregistered_SAD
SAD Send Admission Request
Channel Receive and Send Admission Request
SAS Receive Admission Request
sad ∈ Valid unregistered sad
sad ∈ registered_SAD V sad ∉ valid_SAD
sad ∈ unregistered_SAD
False
True
SAS_RegSAD_Fail
SAS_RegSAD_Success
fault = Oversupply
SAS Detect fault
fault = Overload
True
True
sad ∈ operational_SAD
sad ∈ standby_SAD
SAS Check Mntr and Send Cmd OP_Release
SAS Send Operate cmd
SAS Check Mntr AND Send_Cmd_OperateCmd
Fail
Success
sad ∈ standby_SAD
mntr_cntnt(mntr) = Transient_failure
cmd_cntnt(cmd) = Corrective_Cmd
sad ∈ Operate_executed_SAD
sad ∈ standby_SAD
mntr_cntnt(mntr) = Normal_info
cmd_cntnt(cmd) = Normal_Cmd
sad ∈ Operate_executed_SAD
SAS_CheckMntrSend_Cmd _OperateCmd_Fail
SAS_CheckMntrSend_Cmd _OperateCmd_Success
Channel Receive and Send Cmd
SAD Receive Cmd
cmd_cntnt(cmd) = Operate_Cmd
True
False
sad ∈ standby_SAD
SAD Execute and Send MntrInfo_OperateCmd
cmd_cntnt(cmd) ≠ Operate_Cmd
Λ cmd_cntnt(cmd ≠ Release_Cmd
True
sad ∈ operational_SAD V unoperational_SAD
SAD Execute and Send MntrInfo_OtherCmd
False
cmd_cntnt(cmd) = Release_Cmd
sad ∈ operational_SAD V unoperational_SAD
SAD Execute and Send MntrInfo_ReleaseCmd
Channel Receive and Send MntrInfo
SAS Receive MntrInfo
sad ∈ Operate_executed_SAD
True
False
sad ∈ AllCmd_executed_SAD
True
False
sad ∈ Release_executed_SAD
True
sad ∈ operational_SAD
mntr_cntnt(mntr) = Normal_info
cmd_cntnt(cmd)= Normal_Cmd
SAS Check Mntr and Send Cmd (OP, Normal)
sad ∈ operational_SAD
mntr_cntnt(mntr) = Transient_failure
cmd_cntnt(cmd)= Corrective_Cmd
SAS Check Mntr and Send Cmd (OP, Transient_failure)
sad ∈ operational_SAD
mntr_cntnt(mntr) = Permanent_failure
SAS DeregisterSAD (OP, Permanent_failure)
sad ∈ unoperational_SAD
mntr_cntnt(mntr) = Normal_info
cmd_cntnt(cmd)= Release_Cmd
SAS Check Mntr and Send Cmd (UnOP,Normal 1)
sad ∈ unoperational_SAD
mntr_cntnt(mntr) = Normal_info
cmd_cntnt(cmd)= Normal_Cmd
SAS Check Mntr and Send Cmd (UnOP,Normal 2)
sad ∈ unoperational_SAD
mntr_cntnt(mntr) = Transient_failure
cmd_cntnt(cmd)= Corrective_Cmd
SAS Check Mntr and Send Cmd (UnOP, Transient_failure)
sad ∈ unoperational_SAD
mntr_cntnt(mntr) = Permanent_failure
SAS DeregisterSAD (UnOP,Permanent_failure)
End
sad ∈ standby_SAD
mntr_cntnt(mntr) = Permanent_failure
SAS DeregisterSAD (Standby,Permanent_failure)
SAS Check Mntr AND Send_Cmd_UNOP_ReleaseCmd
SAS Check Mntr AND Send_Cmd_OP_ReleaseCmd
Fail
Success
sad ∈ unoperational_SAD
mntr_cntnt(mntr) = Transient_failure
cmd_cntnt(cmd) = Corrective_Cmd
sad ∈ Release_executed_SAD
sad ∈ unoperational_SAD
mntr_cntnt(mntr) = Normal_info
sad ∈ Release_executed_SAD
Fail
Success
sad ∈ operational_SAD
mntr_cntnt(mntr) = Transient_failure
cmd_cntnt(cmd) = Corrective_Cmd
sad ∈ Release_executed_SAD
sad ∈ operational_SAD
mntr_cntnt(mntr) = Normal_info
sad ∈ Release_executed_SAD
SAS_CheckMntrSend_Cmd _UNOP_ReleaseCmd_Fail
SAS_CheckMntrSend_Cmd_ UNOP_ReleaseCmd_Success
SAS_CheckMntrSend_Cmd _OP_ReleaseCmd_Fail
SAS_CheckMntrSend_Cmd _OP_ReleaseCmd_Success

Figure 6-11: System activity diagram, showing system processes and the communication pattern

6.3 Refinement Strategy

As discussed in Chapter 2 in section 2.10 and Appendix C**,** the formal modelling in Event-B starts with an abstract high level of the functional requirements presented in sections 6.2.1., 6.2.2, and 6.2.3. Then details are introduced gradually, stepwise, to manage the complexity of the system specifications in the first refinement. The overall strategy is as follows: each stage may include multiple state machines:

1. Building the formal model for the functionality that captures the functional requirements for the targeted system, including:
   a. Building the insecure system in the abstract formal model captures the abstract functionality of the system as shown in Figure 6-10
   b. Augmenting the functionality in the first refinement model, where a horizontal superposition refinement is applied to augment the machine's events by adding more details and complexity of the functional requirements to reach a better description of the targeted system. This refinement introduces availability as a security control where operational failures are introduced.
2. Building the secure system by introducing the security controls and attacks in the second refinement, which captures the non-functional requirements. Further details on security are discussed in Chapter 7.

Figure 6-12 draws the refinement strategy, showing the purpose of each refinement and relationships between stages:

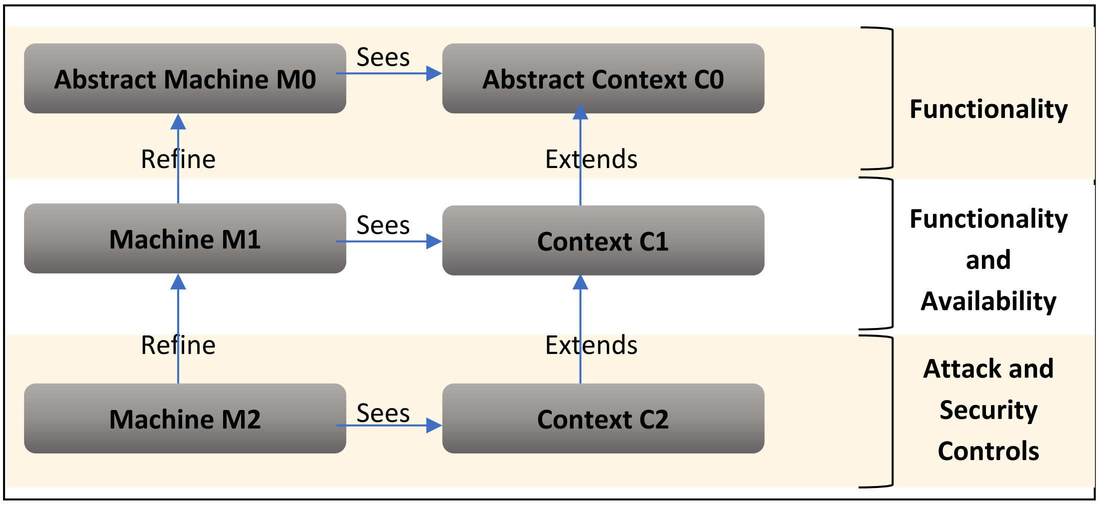


Figure 6-12: Refinement strategy

6.4 First Refinement

This refinement provides the details of Transient failure under which the SAD device is temporarily unavailable, as well as Permanent failure under which it is retired. Moreover, it provides deterministic processing when the unoperational SAD recovers from a transient failure based on the details specified in this refinement.

Transient failure is comprised of many types, including Overheat, Malfunction, Software outdated, and finishing the previous command. Each Transient failure of the registered SADs has its corresponding representation on the SAS side. Therefore, the unoperational state of SAD is made up of many states, based on the type of Transient failure consecutively (rolling_blackout_SAD, on_repair_SAD, on_upgrade_SAD, and busy_SAD).

6.4.1 Availability and Operational Failures

This refinement lists all the operational failures that cause the device to be unavailable and not to respond to the command. Modelling the operational failures is a crucial step before modelling the security failures in the next chapter. This way, if a failure occurs, it will be possible to distinguish security breaches from the operational failures.

To maximise the availability of the station/substation, the status of these stations/substations must be monitored using monitoring information and comprehensive reports handled on IoT-SAD devices. Availability can be achieved by preventing outages and, for this reason, this refinement formally defines the operational failure modes (Transient and Permanent failures) and models the troubleshooting or corrective commands to recover or maintain remediation of the failed access points of the stations/substations to prevent outages. Event B verifies that the model adheres to the availability property by showing that a failed SAD eventually becomes available again after certain recovery actions and corrective commands. Figure 6-13 shows failures where the command cannot be executed by SAD:

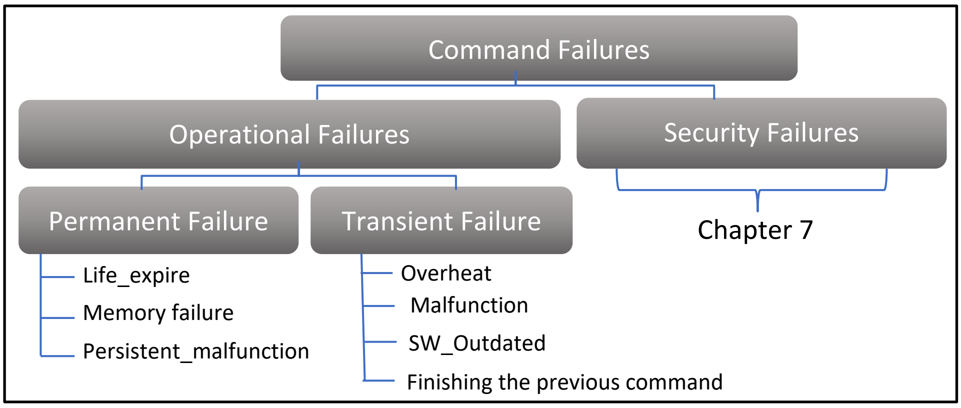


Figure 6-13: Command failures showing operational failures

6.4.2 Functional Requirements for the First Refinement

In the first refinement, more details are added in terms of disaggregated Transient failures and corresponding disaggregated unoperational states of SAD, as explained in Table 6-7.

Table 6-7: Functional requirements for the first refinement

Serial No.	Functional Requirements
ENV-1	The Transient failure that is sent by SAD as monitoring Information is disaggregated into Overheat, Malfunction, SW_outdated, and Finishing the previous command
ENV-1.1	Each Transient failure has a corresponding state, as follows: rolling blackout state, on_repair state, on_upgrade state, and busy state
FUN-16	If SAS receives Overheat, then SAS converts operational SAD to the rolling blackout set of SAD and sends the Acknowledge_power ON command to SAD through the channel
FUN-16.1	SAD executes the Acknowledge_power ON command and sends the execution result as monitoring information to SAS through the channel
FUN-16.2	If SAD is cooled down and the power is turned ON that is SAS receives Normal Information, SAS sends the Release command to SAD to convert it to standby SAD, SAD is remains in the rolling blackout state until Release command is executed
FUN-16.3	If execution fails; that is, SAS receives Overheat as monitoring information, then the rolling_blackout SAD remains in the rolling blackout set of SAD, and SAS sends the Acknowledge_power ON command
FUN-17	If SAS receives Malfunction as monitoring information, then SAS converts the operational SAD to the on_repair set of SAD and sends the Troubleshooting command to SAD through the channel
FUN-17.1	SAD executes the Troubleshooting command and sends the execution result as monitoring information to SAS through the channel
FUN-17.2	If execution succeeds and SAD is fixed that is SAS receives Normal Information, SAS sends the Release command to SAD to convert it to standby SAD, SAD is still in the on_repair state until Release command is executed
FUN-17.3	If execution fails; that is, SAS receives Malfunction as monitoring information, then the on_repair SAD remains in the on_repair set of SAD, and SAS sends the Troubleshooting command.
FUN-18	If SAS receives SW_outdated as monitoring information, then SAS converts operational SAD to the on_upgrade set of SAD and sends the Patching command to SAD through the channel
FUN-18.1	SAD executes the Patching command and sends the execution result as monitoring information to SAS through the channel
FUN-18.2	If execution succeeds and SAD is upgraded; that is, SAS receives Normal Information, SAS sends the Release command to SAD to convert it to standby SAD. SAD remains in the on_upgrade state
FUN-18.3	If execution fails; that is, SAS receives SW_outdated as monitoring information, then the on_upgrade SAD remains in the on_upgrade set of SAD, and SAS sends the Patching command

FUN-19	If SAS receives Finishing the previous command as monitoring information, then SAS converts operational SAD to the busy set of SAD and sends the Acknowledge_finish command to SAD through the channel
FUN-19.1	SAD executes the Acknowledge_finish command and sends the execution result as monitoring information to SAS through the channel
FUN-19.2	If execution succeeds and SAD finishes the previous command; that is, SAS receives Normal information, SAS converts Busy SAD to operational SAD and sends the Normal command to the channel
FUN-19.3	If execution fails; that is, SAS receives Finishing the previous command, the busy SAD remains in the busy set of SAD, and SAS sends the Acknowledge_finish command
ENV-2	The Permanent failure that is sent by SAD as monitoring information is disaggregated into Life_expire, and Memory_failure
FUN-20	At any state of SAD, either operational or any disaggregated state of unoperational, If the received monitoring information is Life_expire or Memory_failure, then SAS deregisters SAD and converts it to retired SAD.

6.4.3 Taxonomy of Transient Failure and Unoperational IoT-Enabled SAD Devices

According to ***Expert#8, Expert#10***, (Saudi Electricity Company, 2019) and (Marcelo *et al.*, 2013), Table 6-8 contains the most probable conditions of Transient failure and the correspondent disaggregated unoperational state of SAD:

Table 6-8: Disaggregated Transient failure and the correspondent unoperational state of SAD

Disaggregated Transient Failure	**Disaggregated unoperational State of SAD**
Overheat failure	1. The IoT-SAD device has rolling blackout status, due to power off in response to overheating failure
Malfunction failure	2. IoT-SAD is in repair due to malfunction
Software outdated failure	3. IoT-SAD is on a software upgrade
Finishing the previous command	4. IoT-SAD is in a busy state, because it is finishing a previous command

As explained, the unoperational state of SAD is disaggregated into multiple states on the basis of the type of Transient failure, which is also disaggregated into multiple types of failure, as presented in Figure 6-14. This diagram shows the taxonomy of Transient failure and unoperational SAD, as well as Permanent failure. Then, Table 6-9 shows the corresponding Corrective command for each unoperational state.

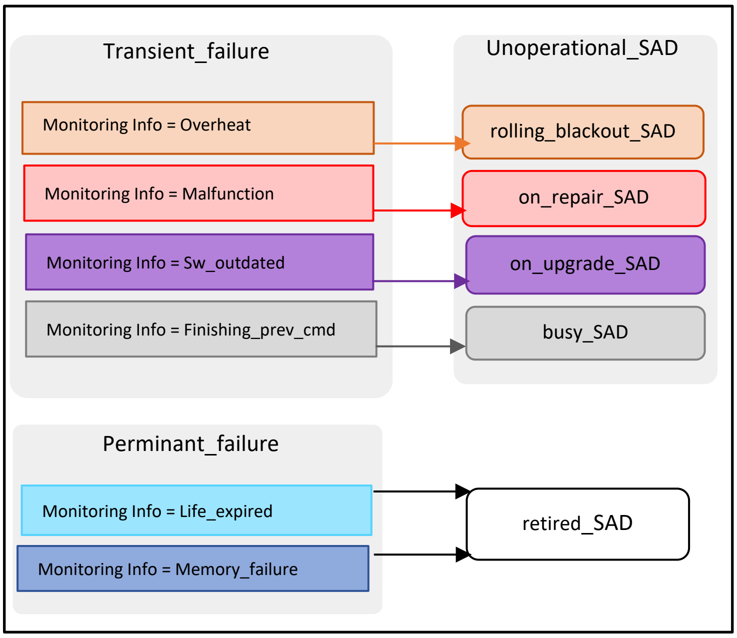

Transient_failure
Monitoring Info = Overheat
Monitoring Info = Malfunction
Monitoring Info = Sw_outdated
Monitoring Info = Finishing_prev_cmd
Unoperational_SAD
rolling_blackout_SAD
on_repair_SAD
on_upgrade_SAD
busy_SAD
Perminant_failure
Monitoring Info = Life_expired
Monitoring Info = Memory_failure
retired_SAD

Figure 6-14: Taxonomy of Transient failure, Perminant failure, and uoperational SAD, with the correspondant perception on SAS side for each Transient failure

Table 6-9: Corresponding corrective command for each unoperational state

Unoperational_SAD	**Corrective Command Sent from SAS to Channel**
rolling_blackout_SAD	Acknowledge_powerON
on_repair_SAD	Troubleshooting
on_upgrade_SAD	Patching
busy_SAD	Acknowledge_finishPrev

Connectivity issues are also a condition of Transient failure, and make SAS unable to communicate with IoT-SAD devices, but the connectivity failure is beyond this research's scope. The disconnectivity might occur when:

- The communication modem becomes faulty
- The SIM card of the IoT device, the communication service provider, or the communication network (such as Ethernet, fiber optics, or wireless) has issues with the connection signals
- Misconfiguration mishaps in the modem, which disconnects the modem from the network.

6.4.4 UML State Machine of the First Refinement

According to the taxonomy in 6.4.3, each abstract event of transient failure is refined by multiple events because transient failure is replaced now by five sub-types including Overheat, Malfunction, Software outdated, and finishing the previous command.

The first refinement is presented in the state machine diagram in Figure 6-16, where the abstract events that are refined are those surrounded by dotted red lines in Figure 6-10. Figure 6-17 maps the abstract events to the refinement events. According to this mapping, the following abstract event of the operational SAD having Transient failure:

SAS Check Mntr and Send Cmd [operational SAD, Transfailure]

is refined by five events for each sub-type of the Transient failure in Figure 6-16, as follow:

- SAS Check Mntr and Send Cmd [operational SAD, Overheat]
- SAS Check Mntr and Send Cmd [operational SAD, Malfunction]
- SAS Check Mntr and Send Cmd [operational SAD, SW_outdated]
- SAS Check Mntr and Send Cmd [operational SAD, Finishing_prev_cmd]

To assist the reader, a consistent naming convention is maintained. Figure 6-15 defines the naming convention for the event: SAS Check Mntr and Send Cmd [operational SAD, Overheat]

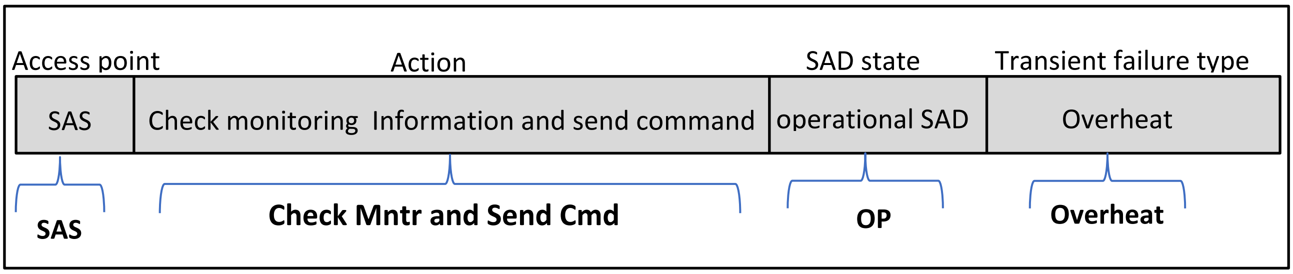


Figure 6-15: The naming convention for the event of sending command

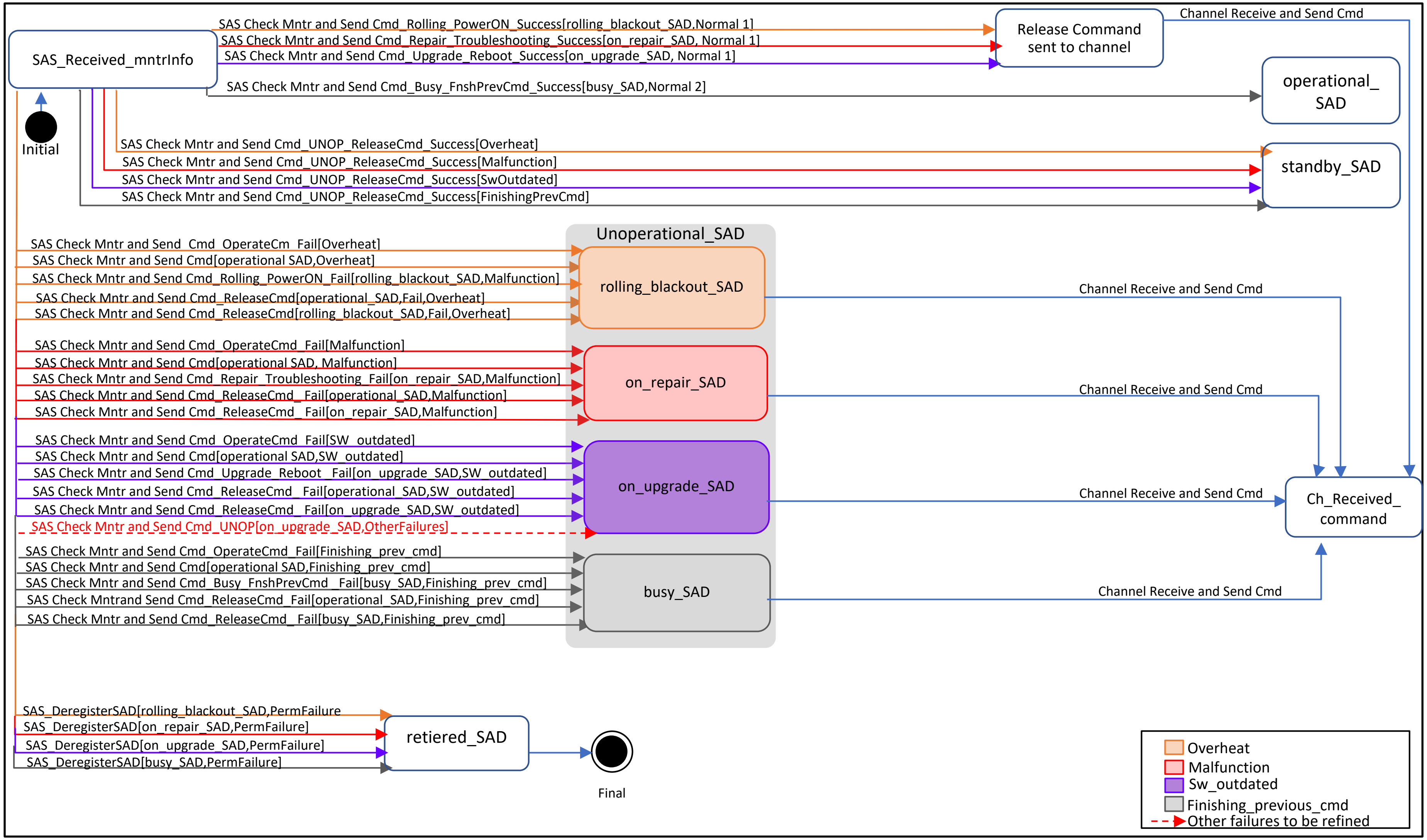

SAS_Received_mntrInfo
Initial
SAS Check Mntr and Send Cmd_Rolling_PowerON_Success[rolling_blackout_SAD,Normal 1]
SAS Check Mntr and Send Cmd_Repair_Troubleshooting_Success[on_repair_SAD, Normal 1]
SAS Check Mntr and Send Cmd_Upgrade_Reboot_Success[on_upgrade_SAD, Normal 1]
Release Command sent to channel
Channel Receive and Send Cmd
SAS Check Mntr and Send Cmd_Busy_FnshPrevCmd_Success[busy_SAD,Normal 2]
operational_SAD
SAS Check Mntr and Send Cmd_UNOP_ReleaseCmd_Success[Overheat]
SAS Check Mntr and Send Cmd_UNOP_ReleaseCmd_Success[Malfunction]
SAS Check Mntr and Send Cmd_UNOP_ReleaseCmd_Success[SwOutdated]
SAS Check Mntr and Send Cmd_UNOP_ReleaseCmd_Success[FinishingPrevCmd]
standby_SAD
Unoperational_SAD
SAS Check Mntr and Send_Cmd_OperateCm_Fail[Overheat]
SAS Check Mntr and Send Cmd[operational SAD,Overheat]
SAS Check Mntr and Send Cmd_Rolling_PowerON_Fail[rolling_blackout_SAD,Malfunction]
SAS Check Mntr and Send Cmd_ReleaseCmd[operational_SAD,Fail,Overheat]
SAS Check Mntr and Send Cmd_ReleaseCmd[rolling_blackout_SAD,Fail,Overheat]
rolling_blackout_SAD
Channel Receive and Send Cmd
SAS Check Mntr and Send Cmd_OperateCmd_Fail[Malfunction]
SAS Check Mntr and Send Cmd[operational SAD, Malfunction]
SAS Check Mntr and Send Cmd_Repair_Troubleshooting_Fail[on_repair_SAD,Malfunction]
SAS Check Mntr and Send Cmd_ReleaseCmd_ Fail[operational_SAD,Malfunction]
SAS Check Mntr and Send Cmd_ReleaseCmd_ Fail[on_repair_SAD,Malfunction]
on_repair_SAD
Channel Receive and Send Cmd
SAS Check Mntr and Send Cmd_OperateCmd_Fail[SW_outdated]
SAS Check Mntr and Send Cmd[operational SAD,SW_outdated]
SAS Check Mntr and Send Cmd_Upgrade_Reboot_ Fail[on_upgrade_SAD,SW_outdated]
SAS Check Mntr and Send Cmd_ReleaseCmd_ Fail[operational_SAD,SW_outdated]
SAS Check Mntr and Send Cmd_ReleaseCmd_ Fail[on_upgrade_SAD,SW_outdated]
SAS Check Mntr and Send Cmd_UNOP[on_upgrade_SAD,OtherFailures]
on_upgrade_SAD
Channel Receive and Send Cmd
Ch_Received_command
SAS Check Mntr and Send Cmd_OperateCmd_Fail[Finishing_prev_cmd]
SAS Check Mntr and Send Cmd[operational SAD,Finishing_prev_cmd]
SAS Check Mntr and Send Cmd_Busy_FnshPrevCmd _Fail[busy_SAD,Finishing_prev_cmd]
SAS Check Mntrand Send Cmd_ReleaseCmd_Fail[operational_SAD,Finishing_prev_cmd]
SAS Check Mntr and Send Cmd_ReleaseCmd_ Fail[busy_SAD,Finishing_prev_cmd]
busy_SAD
Channel Receive and Send Cmd
SAS_DeregisterSAD[rolling_blackout_SAD,PermFailure
SAS_DeregisterSAD[on_repair_SAD,PermFailure]
SAS_DeregisterSAD[on_upgrade_SAD,PermFailure]
SAS_DeregisterSAD[busy_SAD,PermFailure]
retiered_SAD
Final
Overheat
Malfunction
Sw_outdated
Finishing_previous_cmd
Other failures to be refined

Figure 6-16: State machine diagram of the first refinement (M1)

6.4.5 Mapping the Abstract Model to the First Refinement

Figure 6-17 maps the abstract events to the refinement events:

Abstract Events	Refinement Events
SAS Check Mntr and Send Cmd _OperateCmd(Fail)	- SAS_CheckMntrSend_Cmd_OperateCmd_Fail[Overheat] - SAS_CheckMntrSend_Cmd_OperateCmd_Fail[Malfunction] - SAS_CheckMntrSend_Cmd_OperateCmd_Fail[SwOutdated] - SAS Check Mntr and Send Cmd_OperateCmd_Fail[Finishing_prev_cmd]
SAS Check Mntr and Send Cmd (operational SAD,Transfailure)	- SAS Check Mntr and Send Cmd[operational SAD,Overheat] - SAS Check Mntr and Send Cmd[operational SAD, Malfunction] - SAS Check Mntr and Send Cmd[operational SAD, SwOutdated] - SAS Check Mntr and Send Cmd[operational SAD, Finishing_prev_cmd]
SAS Check Mntr and Send Cmd (unoperational SAD,Transfailure)	- SAS Check Mntr and Send Cmd_Rolling_PowerON_Fail[rolling_blackout_SAD,Overheat] - SAS Check Mntr and Send Cmd_Repair_Troubleshooting_Fail[on_repair_SAD,Malfunction] - SAS_CheckMntrSend_Cmd_Upgrade_Reboot _Fail[on_upgrade_SAD,SwOutdated] - SAS Check Mntr and Send Cmd_Busy_FnshPrevCmd _Fail[busy_SAD,Finishing_prev_cmd]
SAS Check Mntr and Send Cmd _OP_ReleaseCmd(Fail)	- SAS Check Mntr and Send Cmd_ReleaseCmd_Fail[operational_SAD,Overheat] - SAS Check Mntr and Send Cmd_ReleaseCmd_Fail[operational_SAD,Malfunction] - SAS Check Mntr and Send Cmd_ReleaseCmd_Fail[operational_SAD,SwOutdated] - SAS Check Mntrand Send Cmd_ReleaseCmd_Fail[operational_SAD,Finishing_prev_cmd]
SAS Check Mntr and Send Cmd _UNOP_ReleaseCmd(Fail)	SAS Check Mntr and Send Cmd_ReleaseCmd_ Fail[rolling_blackout_SAD,Overheat] SAS Check Mntr and Send Cmd_ReleaseCmd_Fail[on_repair_SAD,Malfunction] SAS Check Mntr and Send Cmd_ReleaseCmd_ Fail[on_upgrade_SAD,SwOutdated] SAS Check Mntr and Send Cmd_ReleaseCmd_Fail[busy_SAD,Finishing_prev_cmd]
SAS Check Mntr and Send Cmd (unoperational SAD,Normal 1)	- SAS Check Mntr and Send Cmd_Rolling_PowerON_Success[rolling_blackout_SAD,Normal 1] - SAS Check Mntr and Send Cmd_Repair_Troubleshooting_Success[on_repair_SAD, Normal 1] - SAS Check Mntr and Send Cmd_Upgrade_Reboot_Success[on_upgrade_SAD, Normal 1]
SAS Check Mntr and Send Cmd (unoperational SAD,Normall 2)	SAS Check Mntr and Send Cmd_Busy_FnshPrevCmd_Succes[busy_SAD,Normal 2]
SAS Check Mntr and Send Cmd _UNOP_ReleaseCmd(Success)	- SAS Check Mntr and Send Cmd_UNOP_ReleaseCmd_Success[Overheat) SAS Check Mntr and Send Cmd_UNOP_ReleaseCmd_Success[Malfunction] SAS Check Mntr and Send Cmd_UNOP_ReleaseCmd_Success[SwOutdated] SAS Check Mntr and Send Cmd_UNOP_ReleaseCmd_Success[FinishingPrevCmd]
SAS Deregister SAD (unoperational SAD,PermFailure)	- SAS_DeregisterSAD[rolling_blackout_SAD,PermFailure] - SAS_DeregisterSAD[on_repair_SAD,PermFailure] - SAS_DeregisterSAD[on_upgrade_SAD,PermFailure] - SAS_DeregisterSAD[busy_SAD,PermFailure]

Figure 6-17: Mapping abstract events to refinement events

6.4.6 Formal Model of the First Refinement

This refinement has two parts: **the context (c1) and the machine (m1)**. Each event where the monitoring content belongs to Transient failure; that is (@grd7: mntr_cntnt(mntr) ∈ Transient_failure), is now extended in this refined model to include detailed sub-modes of that Transient failure. Looking back to the abstract event (**event** SAS_CheckMntrSend_Cmd_OP_ TransFailure), this event is refined into the following events:

- **event** SAS_CheckMntrSend_Cmd_OP_Overheat
- **event** SAS_CheckMntrSend_Cmd_OP_Malfunction
- **event** SAS_CheckMntrSend_Cmd_OP_SwOutdated
- **event** SAS_CheckMntrSend_Cmd_OP_FnshPrevCmd

For instance, if the Transient failure event is Overheat and SAD is operational, then the event (**event** SAS_CheckMntrSend_Cmd_OP_Overheat) is triggered whereby two guards are added in the refinement:

Guard	**Aim**
@grd15: corCmd = Acknowledge_powerON	To set out the corresponding command, instead of the abstract Corrective command
@grd16: mntr_cntnt(mntr) = Overheat	To specify the exact type of transient failure

If SAD shows Overheat as monitoring information, it powers itself OFF from the relay. Then, SAS converts SAD to the corresponding state of SAD; that is, Rolling_blackout_SAD, in the action (@act10), instead of the abstract unoperational_SAD. Likewise, the same aspects are modelled for the other sub-modes of the abstract Transient failure, as presented in Figure 6-11:

```
event SAS_CheckMntrSend_Cmd_OP_Overheat extends SAS_CheckMntrSend_Cmd_OP_TransFailure
where //correspond to FUN-16 in Table 6-10, the Transient failure is Overheat
   @grd15: corCmd = Acknowledge_powerON
   @grd16: mntr_cntnt(mntr) = Overheat
then
   @act10: rolling_blackout_SAD ≔ rolling_blackout_SAD ∪ {sad}
end
event SAS_CheckMntrSend_Cmd_OP_Malfunction extends SAS_CheckMntrSend_Cmd_OP_TransFailure
where //correspond to FUN-17 in Table 6-11, the Transient failure is Malfunction
   @grd15: corCmd = Troubleshooting_Cmd
   @grd16: mntr_cntnt(mntr) = Malfunction
then
   @act10: on_repair_SAD ≔ on_repair_SAD ∪ {sad}
end
event SAS_CheckMntrSend_Cmd_OP_SwOutdated extends SAS_CheckMntrSend_Cmd_OP_TransFailure
where //correspond to FUN-18 in Table 6-12, the Transient failure is SW Outdated
   @grd15: corCmd = Patching_Cmd
   @grd16: mntr_cntnt(mntr) = SW_outdated
then
   @act10: on_upgrade_SAD ≔ on_upgrade_SAD ∪ {sad}
end
event SAS_CheckMntrSend_Cmd_OP_FnshPrevCmd extends SAS_CheckMntrSend_Cmd_OP_TransFailure
where //correspond to FUN-19 in Table 6-13,the Transient failure is finishing the previous command
   @grd15: corCmd = Acknowledge_finishPrev
   @grd16: mntr_cntnt(mntr) = Finishing_prev_cmd
then
   @act10: busy_SAD ≔ busy_SAD ∪ {sad}
end
```

Code 6-11: Refinement events of the abstract event (SAS_CheckMntrSend_Cmd_OP_TransFailure), where SAD is operational sending a Transient failure

The following axioms partition the Transient failure and the Corrective Command as follows:

@Valid_Mntr2: **partition** (Transient_failure, {Overheat}, {Malfunction}, {SW_outdated}, {Finishing_prev_cmd})

@Valid_Command2: **partition** (Corrective_Cmd, {Acknowledge_powerON}, {Troubleshooting_Cmd}, {Patching_Cmd}, {Acknowledge_finishPrev}, OTHER_CORRECTIVE_CMD)

After the rolling_blackout_SAD recovers and its power is turned ON, which means that the execution of Acknowledge_powerON has succeeded, then SAS sends the Released_Cmd to convert this SAD to the standby state until it is required when another overloading happen. Therefore, the abstract event Normal1 in Code 6-9 (**event** SAS_CheckMntrSend_Cmd_UNOP_Normal1) is now refined to include an extra guard (@grd9: sad ∈ rolling_blackout_SAD), as follows in Code 6-12:

```
event SAS_CheckMntrSend_Cmd_Rolling_PowerON_Success extends SAS_CheckMntrSend_Cmd_UNOP_Normal1
//correspond to FUN-16.2 in Table 6-7
where
   @grd9: sad ∈ rolling_blackout_SAD
end
```

Code 6-12: Successful execution for Power ON command

The same modelling aspects are applied for the failures of Malfunction and SW Outdated, because SAD is turned to standby after recovering from these failures. This means that the abstract event Normal1 is refined (see Code 6-9), whereas if SAD recovers from the failure of Finishing the previous command, SAD is turned into the operational state. This means that the abstract event Normal2 is refined (see Code 6-9).

These refined events remove the non-deterministic processing in the abstract event because the cases in which SAD turned to operational or standby are explicitly specified in this refinement, as shown in the following Code 6-13. So, when SAD recovers from Overheat, Malfunction, or SW_outdated, and sends Normal in the monitoring information, then SAS sends the Release command to those unoperational SADs to be converted to standby. While SAD is converted to the operational state after recovering from Finishing the previous command.

```
event SAS_CheckMntrSend_Cmd_Repair_Troubleshooting_Success extends
SAS_CheckMntrSend_Cmd_UNOP_Normal1 //correspond to FUN-17.2 in Table 6-7
where
   @grd9: sad ∈ on_repair_SAD
end

event SAS_CheckMntrSend_Cmd_Upgrade_Patch_Success extends SAS_CheckMntrSend_Cmd_UNOP_Normal1
//correspond to FUN-18.2 in Table 6-7
where
   @grd9: sad ∈ on_upgrade_SAD
end
event SAS_CheckMntrSend_Cmd_Busy_FnshPrevCmd_Success extends SAS_CheckMntrSend_Cmd_UNOP_Normal2
//correspond to FUN-19.2 in Table 6-7
where
   @grd13: sad ∈ busy_SAD
then
   @act12: busy_SAD ≔ busy_SAD \ {sad}
end
```

Code 6-13: Refinement events for abstract non-deterministic events

Returning to the previous example of Overheat failure, in the event that the execution of the Acknowledge_powerON command fails then the unoperational SAD Rolling_blackout_SAD remains in the same state, and SAS sends the same corrective command (Acknowledge_powerON). This failed execution is formalised by refining the abstract event (**event** SAS_CheckMntrSend_Cmd_UNOP_TransFailure) as in the following Code 6-14:

```
event SAS_CheckMntrSend_Cmd_Rolling_PowerON_Fail extends SAS_CheckMntrSend_Cmd_UNOP_TransFailure
where //correspond to FUN-16.3 in Table 6-7
   @grd14: sad ∈ rolling_blackout_SAD
   @grd15: mntr_cntnt(mntr) = Overheat
   @grd16: corCmd = Acknowledge_powerON
End
event SAS_CheckMntrSend_Cmd_Repair_Troubleshooting_Fail extends
SAS_CheckMntrSend_Cmd_UNOP_TransFailure //correspond to FUN-17.3 in Table 6-19
where
   @grd14: sad ∈ on_repair_SAD
   @grd15: mntr_cntnt(mntr) = Malfunction
   @grd16: corCmd = Troubleshooting_Cmd
End
event SAS_CheckMntrSend_Cmd_Upgrade_Patch_Fail extends SAS_CheckMntrSend_Cmd_UNOP_TransFailure
where //correspond to FUN-18.3 in Table 6-7
   @grd14: sad ∈ on_upgrade_SAD
   @grd15: mntr_cntnt(mntr) = SW_outdated
   @grd16: corCmd = Patching_Cmd
End
event SAS_CheckMntrSend_Cmd_Busy_FnshPrevCmd_Fail extends SAS_CheckMntrSend_Cmd_UNOP_TransFailure
where //correspond to FUN-19.3 in Table 6-7
   @grd14: sad ∈ busy_SAD
   @grd15: mntr_cntnt(mntr) = Finishing_prev_cmd
   @grd16: corCmd = Acknowledge_finishPrev
end
```

Code 6-14: Refinement events for the abstract event (SAS_CheckMntrSend_Cmd_UNOP_TransFailure) where SAD is unoperational sending a Transient failure

Following the same approach, the abstract event related to permanent failure is refined such as (**event** SAS_DeregisterSAD_UNOP_PermFailure) to include the Permanent failure partition in the axiom: @Valid_Mntr3: **partition** (Permanent_failure, {Life_expire}, {Memory_failure}, {Persist_malfunction}). Also, guards and actions are added to describe the system behaviour for each disaggregated state of unoperational SAD that encounters a permanent failure (see Event-B code in Appendix D).

6.5 Other Failures

This subsection represents failures, other than operational failures, that may occur in the target system. As shown previously in Figure 6-13, the command failures are of two types: operational and security. Both cause the device to be unavailable and not to respond to the command. In the current abstract formal model in this chapter, these security failures are represented at a high level in abstract events (see Event-B code in Appendix D) created for other failures that may have occurred, which will be refined later in the security formalisation in Chapter 7 to add more detail to such events. For example, the event SAS_CheckMntrSend_Cmd_OP_OtherFailures in Appendix D that is annotated in general as operational SAD is converted to the unoperational set of SADs when other failures are encountered. A Corrective command is sent, but it does not provide any details about the exact monitoring information and conditions under which these other failures occur, nor the exact type of corrective command.

The following UML state machine in Figure 6-17 shows those events related to other failures, highlighted in brown colour:

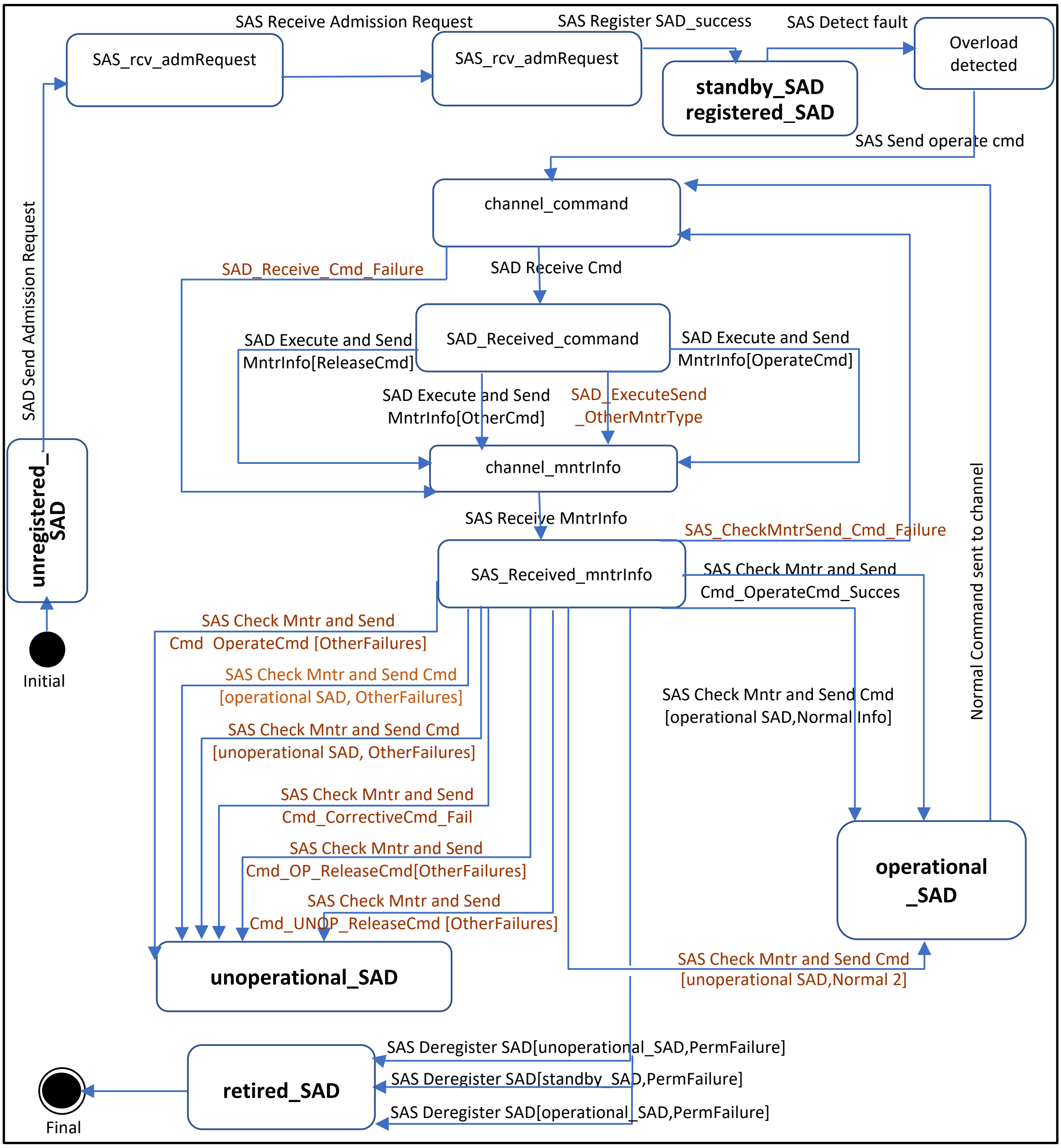


Figure 6-18: State machine diagram of the abstract model, showing the abstract events for other failures

6.6 Conclusion

In this chapter, a portion of the reference model was selected and a formal specification of the sub-grid was developed. The formal modelling helped to develop more detailed functional and operational aspects of the system in preparation for modelling the security failures in the next chapter. The modelling of the operational failures presents the modelling of the system's availability.

Event-B-based formal modelling helped to verify the informal initial system requirements that were confirmed by experts in Chapter 5. The verified functional requirements are listed and labelled in Table 6-1, Table 6-4, Table 6-5, Table 6-6, and Table 6-7. The reference model was very high-level in terms of describing the interaction between the access points. Formal modelling helped to construct a more detailed picture of the interaction between SAS and SAD and to understand their functionality adequately. Moreover, formal modelling contributes to establishing the foundation for security checks. Figure 6-18 represents the reference model and how the formal modelling zooms in the interaction perspective of the targeted sub-grid.

The correctness of the developed formal model is verified through the Proof Obligations (PO) generated in the Rodin toolset, resulting in 360 POs, of which (100%) were discharged and proved automatically via Rodin provers. Table 6-10 depicts the PO statistics. The POs ensure that the invariants are preserved by events guards and actions.

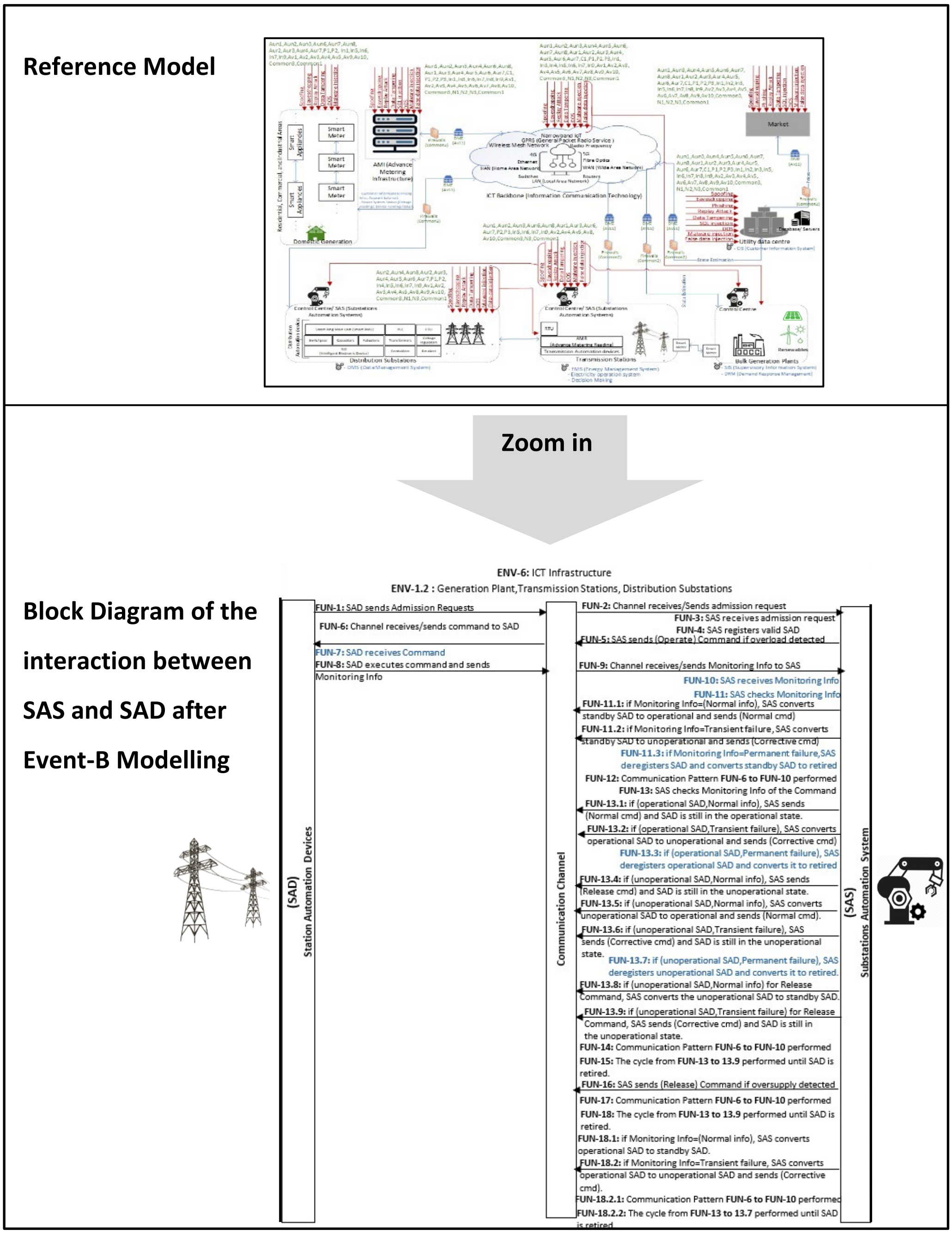
The developed formal model was validated using model checking via ProB animator. The model is animated using ProB and it follows and fulfils the expected state machines, and the events met the expected behaviour. The events were analysed using ProB to ensure that the model is deadlock-free. ProB was used to check the correctness and consistency of the specifications systematically.

Table 6-10: Proof statistics using Rodin platform

Event-B Component	Total	Auto	Manual	Undischarged
Total	**360**	**328**	**32**	**0**
M0	177	167	10	0
M1	183	161	22	0
C0	0	0	0	0
C1	0	0	0	0

6.7 Summary

As shown in this chapter, the abstract model and the first augmentation were built formally using Event-B, which is automatically proved by Rodin. The Formal Model until this point presents the functional requirements and the Availability property in preparation to augment the model by adding levels of complexity that will introduce more security controls in the formal model of the IoT-enabled Smart Grid. More details on the security aspect are introduced through refinement in the following Chapter 7.


Reference Model
Zoom in
Block Diagram of the interaction between SAS and SAD after Event-B Modelling
(SAD) Station Automation Devices
Communication Channel
(SAS) Substations Automation System

ENV-6: ICT Infrastructure
ENV-1.2 : Generation Plant,Transmission Stations, Distribution Substations

FUN-1: SAD sends Admission Requests
FUN-2: Channel receives/Sends admission request
FUN-3: SAS receives admission request
FUN-4: SAS registers valid SAD
FUN-5: SAS sends (Operate) Command if overload detected
FUN-6: Channel receives/sends command to SAD
FUN-7: SAD receives Command
FUN-8: SAD executes command and sends Monitoring Info
FUN-9: Channel receives/sends Monitoring Info to SAS
FUN-10: SAS receives Monitoring Info
FUN-11: SAS checks Monitoring Info
FUN-11.1: if Monitoring Info=(Normal info), SAS converts standby SAD to operational and sends (Normal cmd)
FUN-11.2: if Monitoring Info=Transient failure, SAS converts standby SAD to unoperational and sends (Corrective cmd)
FUN-11.3: if Monitoring Info=Permanent failure,SAS deregisters SAD and converts standby SAD to retired
FUN-12: Communication Pattern FUN-6 to FUN-10 performed
FUN-13: SAS checks Monitoring Info of the Command
FUN-13.1: if (operational SAD,Normal info), SAS sends (Normal cmd) and SAD is still in the operational state.
FUN-13.2: if (operational SAD,Transient failure), SAS converts operational SAD to unoperational and sends (Corrective cmd)
FUN-13.3: if (operational SAD,Permanent failure), SAS deregisters operational SAD and converts it to retired
FUN-13.4: if (unoperational SAD,Normal info), SAS sends (Release cmd) and SAD is still in the unoperational state.
FUN-13.5: if (unoperational SAD,Normal info), SAS converts unoperational SAD to operational and sends (Normal cmd).
FUN-13.6: if (unoperational SAD,Transient failure), SAS sends (Corrective cmd) and SAD is still in the unoperational state.
FUN-13.7: if (unoperational SAD,Permanent failure), SAS deregisters unoperational SAD and converts it to retired.
FUN-13.8: if (unoperational SAD,Normal info) for Release Command, SAS converts the unoperational SAD to standby SAD.
FUN-13.9: if (unoperational SAD,Transient failure) for Release Command, SAS sends (Corrective cmd) and SAD is still in the unoperational state.
FUN-14: Communication Pattern FUN-6 to FUN-10 performed
FUN-15: The cycle from FUN-13 to 13.9 performed until SAD is retired.
FUN-16: SAS sends (Release) Command if oversupply detected
FUN-17: Communication Pattern FUN-6 to FUN-10 performed
FUN-18: The cycle from FUN-13 to 13.9 performed until SAD is retired.
FUN-18.1: if Monitoring Info=(Normal info), SAS converts operational SAD to standby SAD.
FUN-18.2: if Monitoring Info=Transient failure, SAS converts operational SAD to unoperational SAD and sends (Corrective cmd).
FUN-18.2.1: Communication Pattern FUN-6 to FUN-10 performed
FUN-18.2.2: The cycle from FUN-13 to 13.7 performed until SAD is retired

Figure 6-19: The transformation from the reference model to a rigorous analysis on the interaction perspective of the targeted system using Event-B Modelling

Chapter 7 **Verification and Validation of the Model: Security Formalisation**

After laying the foundation of the system's functionality and basic aspects of availability in Chapter 6, this chapter continues augmenting the model by incorporating further aspects of security controls to mitigate the external harm affecting the system's functionality. The external harm in this context is defined as internet-based cyber threats. The formal model in Chapter 6 focuses solely on system functionality, emphasising availability as a security property without additional security controls. To address system complexity, this chapter employs the second refinement to integrate the security controls as non-functional requirements into the formal model established in Chapter 6.

It was crucial to model the operational failures first, as developed in the first refinement in Chapter 6, before modeling the security failure. Now, this chapter adds the conditions whereby failure is identified as a security breach. This refinement introduces the non-functional requirements and corresponding security controls, strating by integrates the controls that address the abnormality to the model, as shown in Figure 7-1. Then, it incorporates the security measures that were confirmed by the field experts in the previous stages of this research. This refinement includes fundamental categorised as imperatives (MUST), facilitating the development of a security formal model governing the information transmission and the interaction between SAS and SAD within the IoT-enabled Smart Grid framework.

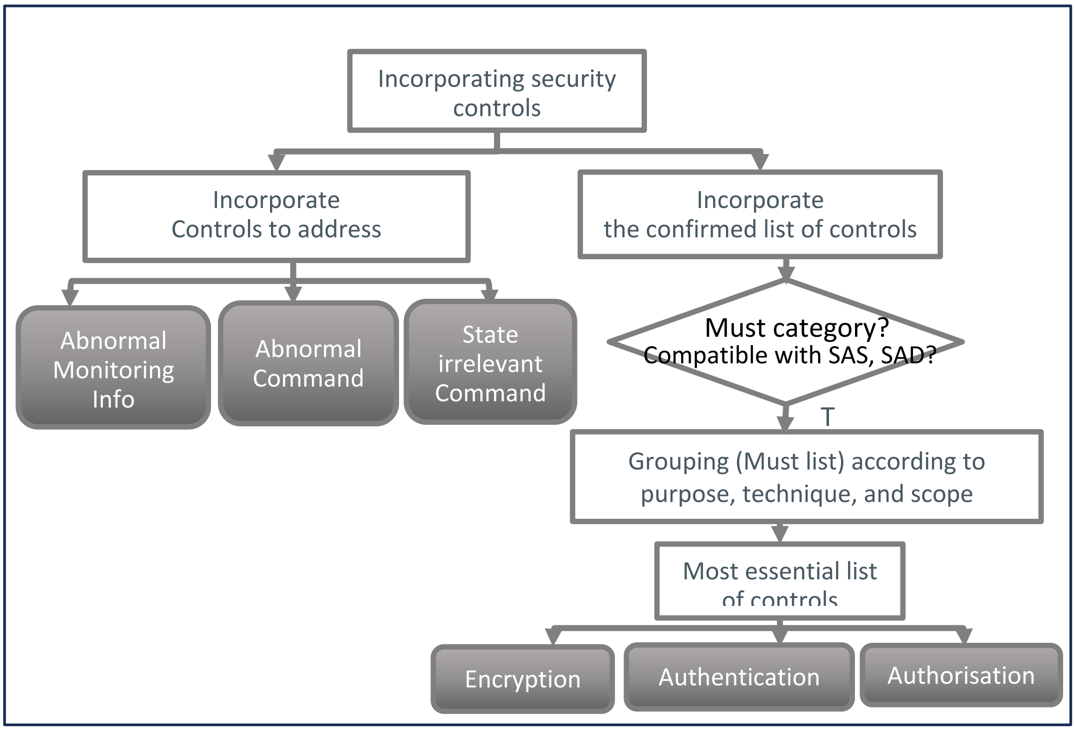

Incorporating security controls
Incorporate Controls to address
Incorporate the confirmed list of controls
Abnormal Monitoring Info
Abnormal Command
State irrelevant Command
Must category? Compatible with SAS, SAD?
T
Grouping (Must list) according to purpose, technique, and scope
Most essential list of controls
Encryption
Authentication
Authorisation

Figure 7-1: Security Formalisation process

7.1 Security Failures and Addressing Abnormality

As shown in section 6.3 in Chapter 6, the list of operational failures was defined and their corresponding conditions were modelled. Consequently, distinguishing between a typical operational failure and a security threat becomes possible in the event of a command failure. Within the scope of this research, any failure that is not inspected as an operational failure in the first refinement is interpreted as indicative of a potential security failure. If the device is not responding, it indicates a failure.

The system then examines this failure. If it is not caused by an operational failure, it is considered as a security failure. Consequently, actions are taken such as converting SAD to Compromised, isolating or blocking SAD, or sending a security alert.

This subsection addresses the following security concerns related to abnormal behaviours of the system and categorises them as security failures, indicating a potential security breach:

- **Abnormal monitoring information**: not belonging to the valid types of monitoring information defined in the system
- **Abnormal commands:** not belonging to the valid types of commands defined in the system
- **State irrelevant commands:** where SAD receives commands irrelevant to its state. For example, it is abnormal for operational SAD to receive an Operate command.

Therefore in this chapter, the second refinement incorporates anomaly management into the formal model as listed in the following hierarchy in Figure 7-2.

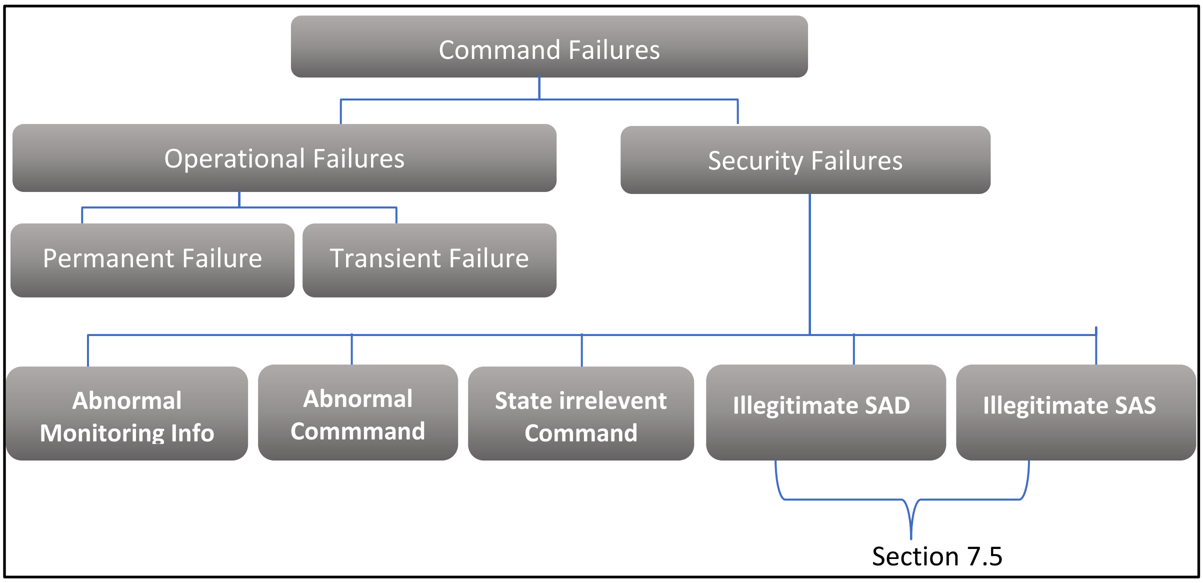


Figure 7-2: Command failures, showing the security failures

7.1.1 System Requirements for Addressing Abnormality

Table 7-1 provides the controls that could address the system abnormality described:

Table 7-1: System requirements for addressing abnormality

Serial No.	System Requirements
REQ 1	If SAS receives abnormal monitoring information from operational SAD, then SAS converts SAD to compromised SAD and sends a Patching command
REQ 2	If SAS receives abnormal monitoring information from on_upgrade SAD, then SAS converts SAD to compromised SAD and sends a Patching command
REQ 3	If standby SAD fails to execute the Operate command by sending abnormal monitoring information, then SAS converts SAD to compromised SAD and sends a Patching command
REQ 4	If operational SAD fails to execute the Release command by sending abnormal monitoring information, then SAS converts SAD to compromised SAD and sends a Patching command.
REQ 5	If unoperational SAD fails to execute the Release command by sending abnormal monitoring information, then SAS converts SAD to compromised SAD and sends a Patching command
REQ 6	If execution of the Patching command succeeds; that is, SAS receives Normal as monitoring information, then SAS converts SAD to operational state and sends Normal command
REQ 7	If execution of the Patching command fails; that is, SAS does not receive Normal as monitoring information, then SAD remains in the compromised state and SAS sends a Patching command. Resending of the Paching command is repeated three times.
REQ 8	If execution of the Patching command fails more than three times; that is, SAS does not receive Normal as monitoring information, then SAS sends the isolate command to compromised SAD for manual check
REQ 9	If SAS identifies an illegitimate SAD, then an attack is detected and SAS blocks this access point
REQ 10	If SAD receives an Abnormal command, then SAD sends a Security Alert as monitoring information to SAS
REQ 11	If operational SAD receives an Operate command, then SAD sends a Security Alert as monitoring information to SAS
REQ 12	If standby SAD receives a Release command, then SAD sends a Security Alert as monitoring information to SAS
REQ 13	If SAS receives a Security Alert in the monitoring information, then SAS sends the previous command and blocks the communicated access point
REQ 14	If SAS receives Security Alert in the monitoring information more than three times, then SAS sends the isolate command to SAD for manual check

7.1.2 UML State Machine for Addressing Abnormality

Section 6.5 explains that the abstract formal model contains a high-level representation of the security failures indicated as “Other Failures”. Now these Abstract events are refined to add more details of security failures. The UML state machine of Other Failures in Figure 6-17 is refined to give more detail on security checks in the following UML state machine in Figure 7-2.

According to the system requirements in Table 7-1, compromised and manualChecked states of SAD are added to the unoperational states in Figure 7-3.

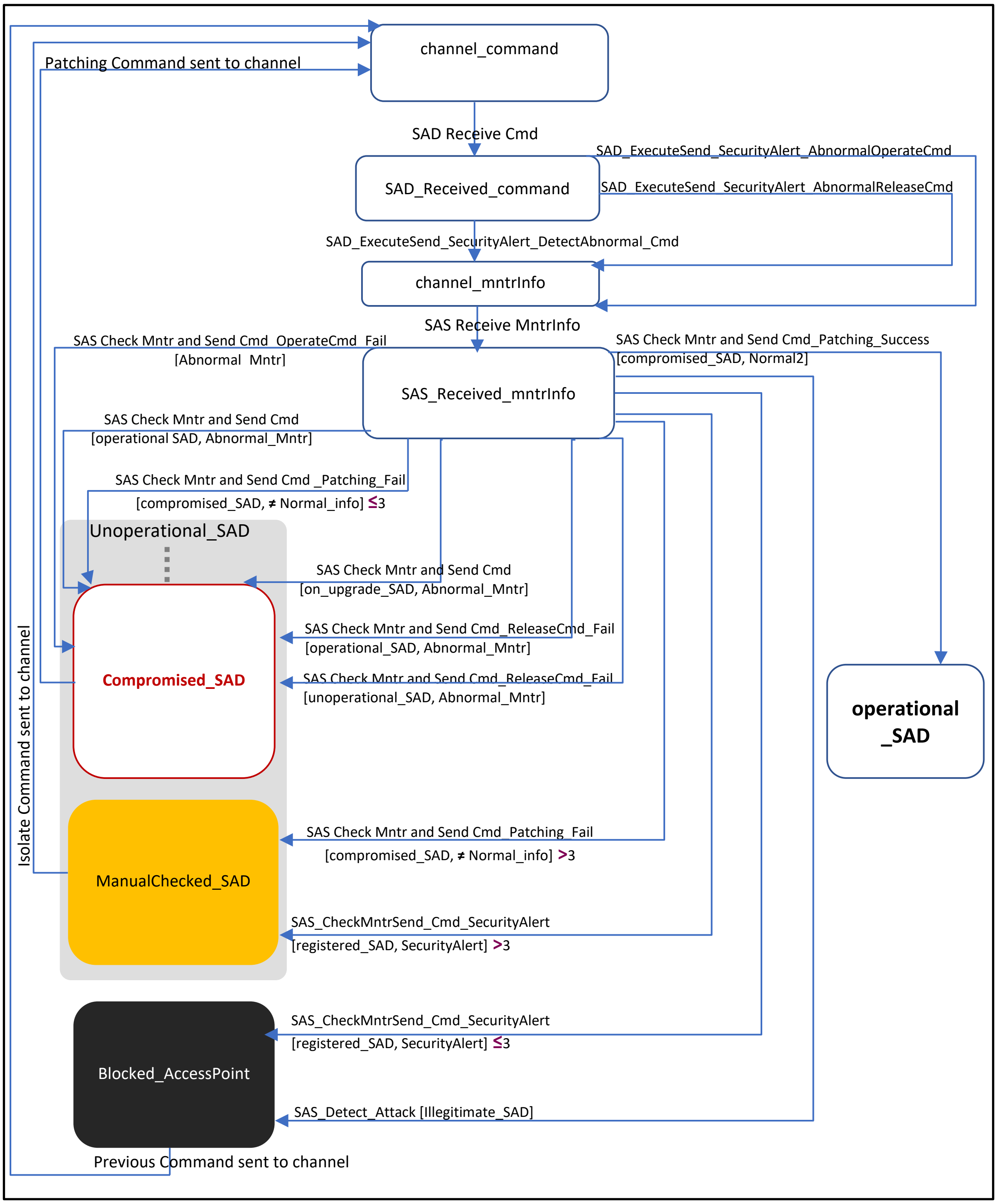

channel_command
Patching Command sent to channel
SAD Receive Cmd
SAD_ExecuteSend_SecurityAlert_AbnormalOperateCmd
SAD_Received_command
SAD_ExecuteSend_SecurityAlert_AbnormalReleaseCmd
SAD_ExecuteSend_SecurityAlert_DetectAbnormal_Cmd
channel_mntrInfo
SAS Receive MntrInfo
SAS Check Mntr and Send Cmd_OperateCmd_Fail
[Abnormal_Mntr]
SAS Check Mntr and Send Cmd_Patching_Success
[compromised_SAD, Normal2]
SAS_Received_mntrInfo
SAS Check Mntr and Send Cmd
[operational SAD, Abnormal_Mntr]
SAS Check Mntr and Send Cmd _Patching_Fail
[compromised_SAD, ≠ Normal_info] ≤3
Unoperational_SAD
SAS Check Mntr and Send Cmd
[on_upgrade_SAD, Abnormal_Mntr]
SAS Check Mntr and Send Cmd_ReleaseCmd_Fail
[operational_SAD, Abnormal_Mntr]
SAS Check Mntr and Send Cmd_ReleaseCmd_Fail
[unoperational_SAD, Abnormal_Mntr]
Compromised_SAD
operational _SAD
Isolate Command sent to channel
SAS Check Mntr and Send Cmd_Patching_Fail
[compromised_SAD, ≠ Normal_info] >3
ManualChecked_SAD
SAS_CheckMntrSend_Cmd_SecurityAlert
[registered_SAD, SecurityAlert] >3
SAS_CheckMntrSend_Cmd_SecurityAlert
[registered_SAD, SecurityAlert] ≤3
Blocked_AccessPoint
SAS_Detect_Attack [Illegitimate_SAD]
Previous Command sent to channel

Figure 7-3: UML State Machine for Adressing Abnormality

The following Table 7-2 maps the abstract events to the refinement events that address the abnormality conditions. The description of these events can be found easily in Table 7-1 using the corresponding serial number. For example, REQ 1 can be used to find the description of the event (SAS Check Mntr and Send Cmd [operational SAD, Abnormal_Mntr]) in Table 7-1. The naming convention for the refinement events is defined in Figure 6-15.

Table 7-2: Mapping the abstract events to the refinement events for addressing the abnormality

Abstract Event	**Event name**	**Serial No.**
SAS Check Mntr and Send Cmd_OperateCmd [OtherFailures]	- SAS Check Mntr and Send Cmd_OperateCmd_Fail [Abnormal_Mntr]	REQ 3
SAS Check Mntr and Send Cmd [operational SAD, OtherFailures]	- SAS Check Mntr and Send Cmd [operational SAD, Abnormal_Mntr]	REQ 1
SAS Check Mntr and Send Cmd_CorrectiveCmd_Fail	- SAS Check Mntr and Send Cmd _Patching_Fail [compromised_SAD, ≠ Normal_info] ≤3	REQ 7
	- SAS Check Mntr and Send Cmd_Patching_Fail [compromised_SAD, ≠ Normal_info] >3	REQ 8
SAS Check Mntr and Send Cmd [unoperational SAD, OtherFailures]	- SAS Check Mntr and Send Cmd [on_upgrade_SAD, Abnormal_Mntr]	REQ 2
SAS Check Mntr and Send Cmd [unoperational SAD, Normal 2]	- SAS Check Mntr and Send Cmd_Patching_Success [compromised_SAD, Normal2]	REQ 6
SAS Check Mntr and Send Cmd_OP_ReleaseCmd [OtherFailures]	- SAS Check Mntr and Send Cmd_ReleaseCmd_Fail [operational SAD, Abnormal_Mntr]	REQ 4
SAS Check Mntr and Send Cmd_UNOP_ReleaseCmd [OtherFailures]	- SAS Check Mntr and Send Cmd_ReleaseCmd_Fail [unoperational SAD, Abnormal_Mntr]	REQ 5
SAS_CheckMntrSend_Cmd_Failure	- SAS_Detect_Attack [Illegitimate_SAD]	REQ 9
	- SAS_CheckMntrSend_Cmd_SecurityAlert - [registered_SAD, SecurityAlert] ≤3	
	- SAS_CheckMntrSend_Cmd_SecurityAlert - [registered_SAD, SecurityAlert] >3	

7.1.3 Formal Model for Addressing Abnormality

This section describes the event-B notations that formalise the system requirements stated in section 7.1.1 which is required to address the following three categories of abnormality:

- **Abnormal monitoring information:**

The abnormal monitoring information is detected on the SAS side. The invariant (@Abnormal_mntr) describes the system behaviour when receiving Abnormal Monitoring Information:

@Abnormal_mntr: $\forall$ x·(x $\in$ channel_mntrInfo $\land$ mntr_cntnt(x) $\notin$ MNTR_TYPE $\Rightarrow$ mntrid(x) $\in$ compromised_SAD)

This invariant states that if SAS receives any abnormal monitoring information, this implies that SAS considers this SAD as compromised. Also, it defines abnormal information as any information of a type that does not belong to the valid MNTR_TYPE, where MNTR_TYPE is defined by the following axioms previously defined in the contexts C0 and C1:

@Valid_Mntr: **partition** (MNTR_TYPE, {Normal_info}, Permanent_failure, Transient_failure, OTHER_MNTR_TYPE)

@Valid_Mntr2: **partition** (Transient_failure, {Overheat}, {Malfunction}, {SW_outdated}, {Finishing_prev_cmd})

@Valid_Mntr3: **partition** (Permanent_failure, {Life_expire}, {Memory_failure}, {Persist_malfunction})

This invariant (@Abnormal_mntr) is maintained by the following events (Code 7-1) and more specifically the following guards:

@grd5: mntr_cntnt(mntr) ≠ Normal_info
@grd6: mntr_cntnt(mntr) ∉ Transient_failure
@grd7: mntr_cntnt(mntr) ∉ Permanent_failure

```
event SAS_CheckMntrSend_Cmd_OP_DetectAbnormal_Mntr extends SAS_CheckMntrSend_Cmd_OP_OtherFailures
where //correspond to REQ-1 in Table 7-1
   @grd5: mntr_cntnt(mntr) ≠ Normal_info
   @grd6: mntr_cntnt(mntr) ∉ Transient_failure
   @grd7: mntr_cntnt(mntr) ∉ Permanent_failure
   @grd8: mntrid(mntr) = sad
   @grd9: cmd_cntnt(cmd) = corCmd
   @grd10: cmdid(cmd) = sad
   @grd11: sad ∉ Release_executed_SAD
   @grd12: sad ∈ AllCmd_executed_SAD
   @grd15: corCmd = Patching_Cmd
   @grd16: counter(sad) ≤ 3
then
   @act6: compromised_SAD ≔  compromised_SAD ∪ {sad}
End

event SAS_CheckMntrSend_Cmd_Upgrade_DetectAbnormal_Mntr extends
SAS_CheckMntrSend_Cmd_UNOP_Upgrade_OtherFailures //correspond to REQ-2 in Table 7-1
where
   @grd5: mntr_cntnt(mntr) ≠ Normal_info
   @grd6: mntr_cntnt(mntr) ∉ Transient_failure
   @grd7: mntr_cntnt(mntr) ∉ Permanent_failure
   @grd8: mntrid(mntr) = sad
   @grd9: cmd_cntnt(cmd) = corCmd
   @grd10: cmdid(cmd) = sad
   @grd11: sad ∈ AllCmd_executed_SAD
   @grd12: corCmd = Patching_Cmd
   @grd13: counter (sad) ≤ 3
then
   @act12: compromised_SAD ≔  compromised_SAD ∪ {sad}
end
```

Code 7-1: SAS detects abnormal monitoring information

As shown in Code 7-1, the event (**event** SAS_CheckMntrSend_Cmd_OP_DetectAbnormal_Mntr) converts the operational SAD to compromised SAD and sends a Patching command if the received monitoring information is abnormal information that does not belong to Normal information, Transient failure, or Permanent failure.

This formalisation is also applied in any case of transient operational failure modelled in the abstract model and first refinement; to include the potential security failure that may occur, ensuring the model accounts for these failure conditions and behaves accordingly. For instance, SAS may receive abnormal monitoring information from on_upgrade SAD as a failed execution result of previous Patching command. Consequently, SAS converts the on_upgarde SAD to compromised and sends a Patching command, as shown in the event (**event** SAS_CheckMntrSend_Cmd_Upgrade_DetectAbnormal_Mntr) in Code 7-1. Owing to the fact that all the transient operational failures occur on the mechanic machinery except for the softwate outdated failure which occurs on the IoT sensors/actuators, as illustrated in Figure 7-4 Therefore, on_upgrade SAD could encounter a security failure as it is a software-related failure.

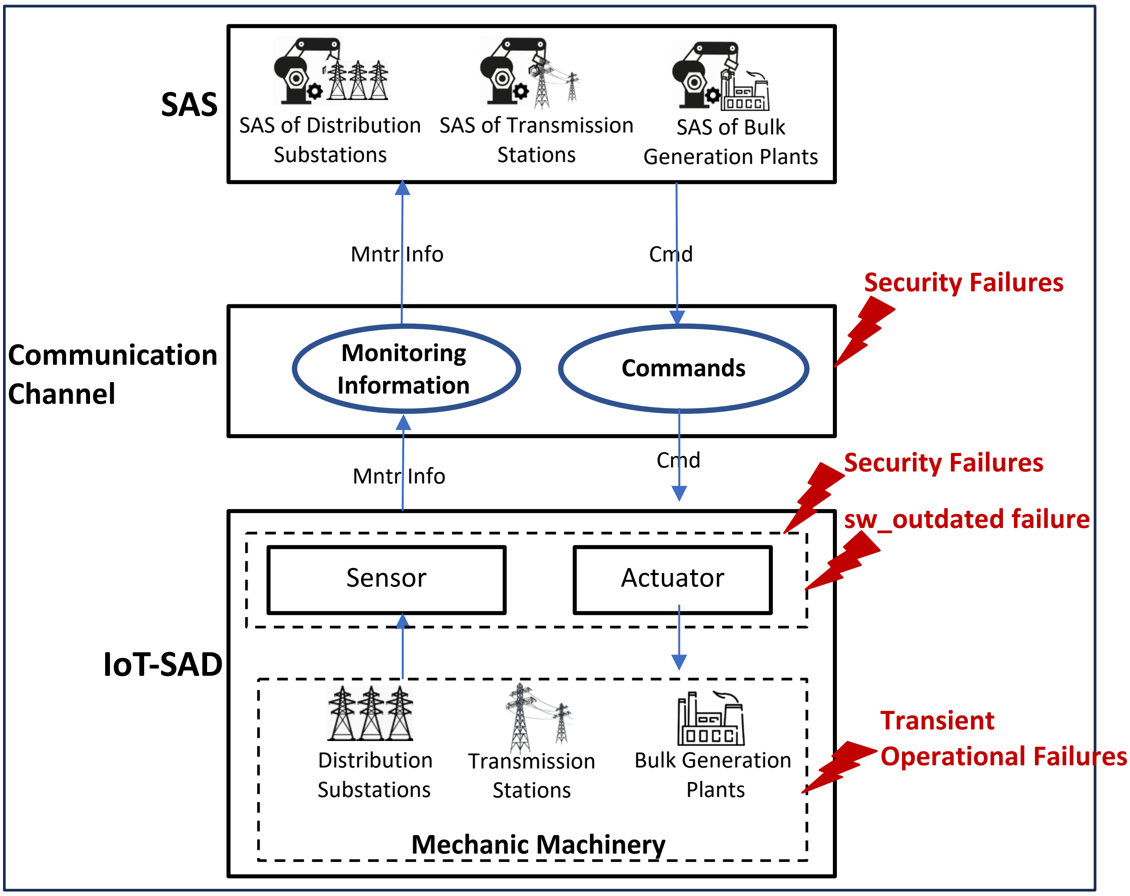


Figure 7-4: Machinery-related failures and software-related failures

Likewise, the Operate command may fail due to a security failure. So, SAS receives abnormal monitoring information as the execution result of the Operate command. The same applies to a Release command that fails to be executed and abnormal monitoring information is received instead. As shown in Code 7-2, there are two events for Release command. One is sent for operational_SAD during overloading. The second is when the Release command is sent to an unoperational_SAD to convert it to standby after recovering from a Transient failure, and back to send Normal information as an execution result of a Corrective command. Both events are refined to include the security failure of a Release command.

```
event SAS_CheckMntrSend_Cmd_OperateCmd_Fail_Abnormal_Mntr extends
SAS_CheckMntrSend_Cmd_OperateCmd_OtherFailures //correspond to REQ-3 in Table 7-1
where
   @grd6: mntr_cntnt(mntr) ∉ Transient_failure
   @grd7: mntr_cntnt(mntr) ∉ Permanent_failure
   @grd8: mntr_cntnt(mntr) ≠ Normal_info
   @grd9: mntrid(mntr) = sad
   @grd10: cmd_cntnt(cmd) = corCmd
   @grd11: cmdid(cmd) = sad
   @grd12: sad ∈ Operate_executed_SAD
   @grd13: corCmd = Patching_Cmd
   @grd14: counter (sad) ≤ 3
then
   @act10: compromised_SAD ≔  compromised_SAD ∪ {sad}
End
event SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Fail_Abnormal_Mntr extends
SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_OtherFailures //correspond to REQ-4 in Table 7-1
where
   @grd5: mntr_cntnt(mntr) ∉ Transient_failure
   @grd6: mntr_cntnt(mntr) ∉ Permanent_failure
   @grd7: mntr_cntnt(mntr) ≠ Normal_info
   @grd8: mntrid(mntr) = sad
   @grd9: cmdid(cmd) = sad
   @grd10: cmd_cntnt(cmd) = corCmd
   @grd11: sad ∈ Release_executed_SAD
   @grd12: corCmd = Patching_Cmd
   @grd13: counter (sad) ≤ 3
then
  @act6: compromised_SAD ≔ compromised_SAD ∪ {sad}
end
event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Fail_Abnormal_Mntr extends
SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_OtherFailures //correspond to REQ-5 in Table 7-1
where
   @grd5: mntr_cntnt(mntr) ∉ Transient_failure
   @grd6: mntr_cntnt(mntr) ∉ Permanent_failure
   @grd7: mntr_cntnt(mntr) ≠ Normal_info
   @grd8: mntrid(mntr) = sad
   @grd9: cmdid(cmd) = sad
   @grd10: cmd_cntnt(cmd) = corCmd
   @grd11: sad ∈ Release_executed_SAD
   @grd12: sad ∈ compromised_SAD
   @grd13: corCmd = Patching_Cmd
End
```

Code 7-2: Failure of the Operate and Release commands due to abnormal monitoring information

The compromised_SAD is a new set variable defined in this second refinement by the partition invariant:

@inv83: **partition** (OTHER_UNOPERATIONAL_SAD, compromised_SAD, OTHER_UNOPERATIONAL_SAD2)

Where OTHER_UNOPERATIONAL_SAD is defined by the following invariant in the first refinement:

@inv66: **partition** (unoperational_SAD, rolling_blackout_SAD, on_repair_SAD, on_upgrade_SAD, busy_SAD, OTHER_UNOPERATIONAL_SAD)

A Patching command may fail or succeed, as shown in Code 7-3. If it fails, SAS sends the Patching command and SAD remains in the compromised state. According to the system requirements, if a compromised SAD fails to execute a Patching command more than three times, SAS sends an isolate command to convert the compromised SAD to the ManualChecked_SAD for manual check by engineers and resets the counter to zero (0). So, each SAD has a counter that is notated by the

```
event SAS_CheckMntrSend_Cmd_compromised_Patching_Success extends
SAS_CheckMntrSend_Cmd_UNOP_Normal2_OtherUNOP //correspond to REQ-6 in Table 7-1
where
    @grd12: sad ∈ compromised_SAD
    @grd13: sad ∉ ManualChecked_SAD
then
    @act8: compromised_SAD ≔ compromised_SAD \ {sad}
end

event SAS_CheckMntrSend_Cmd_compromised_Patching_Fail_AbnormalMntr extends
SAS_CheckMntrSend_Cmd_Corrective_Fail //correspond to REQ-7 in Table 7-1
where
    @grd7: mntr_cntnt(mntr) ≠ Normal_info
    @grd8: sad ∈ compromised_SAD
    @grd9: cmd_cntnt(cmd) = Patching_Cmd
    @grd10: counter (sad) < 3
    @grd11: sad ∈ AllCmd_executed_SAD
then
    @act5: counter(sad) ≔  counter(sad) + 1
end

event SAS_CheckMntrSend_Cmd_compromised_Patching_Fail_AbnormalMntr_MaxReached extends
SAS_CheckMntrSend_Cmd_Corrective_Fail //correspond to REQ-7 in Table 7-1
where
    @grd7: mntr_cntnt(mntr) ≠ Normal_info
    @grd8: sad ∈ compromised_SAD
    @grd9: cmd_cntnt(cmd) = Patching_Cmd
    @grd10: counter (sad) = 3
    @grd11: sad ∈ AllCmd_executed_SAD
then
    @act5: counter(sad) ≔ 0
end

event SAS_CheckMntrSend_Cmd_compromised_Patching_Fail_AbnormalMntr_MaxExceeded extends
SAS_CheckMntrSend_Cmd_Corrective_Fail //correspond to REQ-8 in Table 7-1
where
    @grd7: mntr_cntnt(mntr) ≠ Normal_info
    @grd8: sad ∈ compromised_SAD
    @grd9: cmd_cntnt(cmd) = isolate_cmd
    @grd10: counter (sad) > 3
    @grd11: sad ∈ AllCmd_executed_SAD
then
    @act5: ManualChecked_SAD ≔ ManualChecked_SAD ∪ {sad}
    @act8: counter(sad) ≔ 0
end
```

Code 7-3: Success and failure of exceution of a Patching command

invariant: @inv85: counter ∈ SAD → ℕ

The axiom (@axm52: **partition** (OTHER_CORRECTIVE_CMD, {isolate_cmd})) defines the Isolate command, where OTHER_CORRECTIVE_CMD is defined in the context (**C1)** by the partition axiom:

@Valid_Command2: **partition** (Corrective_Cmd, {Acknowledge_powerON}, {Troubleshooting_Cmd}, {Patching_Cmd}, {Acknowledge_finishPrev}, OTHER_CORRECTIVE_CMD)

The same concept of the counter is applied if the on_repair SAD fails to execute the Troubleshooting command more than three times. The SAS converts the on_repair_SAD to ManualChecked_SAD, as shown in Code 7-4. Therefore, ManualChecked_SAD is defined by the invariant:

@inv86: ManualChecked_SAD ⊆ on_repair_SAD ∪ compromised_SAD

The following invariants set out the condition where on_repair_SAD and compromised_SAD are converted to ManualChecked_SAD:

@inv88: ∀ x· ((mntrid(x) ∈ on_repair_SAD) ∧ (counter(mntrid(x))>3)) ⇒ mntrid(x) ∈ ManualChecked_SAD

@inv89: ∀ x· ((mntrid(x) ∈ compromised_SAD) ∧ (counter(mntrid(x))>3)) ⇒ mntrid(x) ∈ ManualChecked_SAD.

```
event SAS_CheckMntrSend_Cmd_Repair_Troubleshooting_Fail_counter extends
SAS_CheckMntrSend_Cmd_Repair_Troubleshooting_Fail
where
   @grd17: counter (sad) < 3
then
   @act7: counter(sad) ≔  counter(sad) + 1
end

event SAS_CheckMntrSend_Cmd_Repair_Troubleshooting_Fail_MaxReached extends
SAS_CheckMntrSend_Cmd_Repair_Troubleshooting_Fail
where
   @grd17: counter (sad) = 3
then
   @act7: counter(sad) ≔ 0
end

event SAS_CheckMntrSend_Cmd_Repair_Troubleshooting_Fail_MaxExceeded extends
SAS_CheckMntrSend_Cmd_UNOP_Repair_OtherFailures
where
   @grd5: mntr_cntnt(mntr) = Malfunction
   @grd7: sad ∈ on_repair_SAD
   @grd8: cmd_cntnt(cmd) = isolate_cmd
   @grd9: counter (sad) > 3
   @grd10: sad ∈ AllCmd_executed_SAD
   @grd11: sad ∉ ManualChecked_SAD
then
   @act8: ManualChecked_SAD ≔ ManualChecked_SAD ∪ {sad}
   @act9: counter(sad) ≔ 0
end
```

Code 7-4: Refinement events for the fail execution of the Troubleshooting command representing the counter concept

It is worth pointing out that compromised_SAD is a valid registered_SAD that sends abnormal monitoring information due to a security threat, yet there is a case where SAS receives monitoring information from an invalid and illegitimate SAD that must be modelled in which this access point is detected as an attack. The event (**event** SAS_Detect_Attack) detects the attack in Code 7-5. In this event, SAS stops communicating or sending any command to that SAD and adds that SAD to the Blocked_AccessPoint set. This is maintained by the invariant:

@illegitimate_SAD: ∀ x· ((x ∈ channel_mntrInfo) ∧ (mntrid(x) ∉ valid_SAD) ∧ (mntr_cntnt(x) ∉ MNTR_TYPE)) ⇒ mntrid(x) ∈ Blocked_AccessPoint

The (Blocked_AccessPoint) is defined by the invariant:

@inv84: Blocked_AccessPoint ⊆ OTHER_AccessPoint

Where OTHER_AccessPoint is defined in the context (C0) by the axiom:

@axm4: **partition** (AccessPoint, SAS, SAD, OTHER_AccessPoint)

```
event SAS_Detect_Attack extends SAS_CheckMntrSend_Cmd_Failure
any //correspond to REQ-9 in Table 7-1
   sad
where
   @grd3: sad ∉ registered_SAD ∨ sad ∉ valid_SAD
   @grd7: mntrid(mntr) = sad
   @grd8: sad ∉ Blocked_AccessPoint
   @grd9: sad ∈ OTHER_AccessPoint
then
   @act4: Blocked_AccessPoint ≔  Blocked_AccessPoint ∪ {sad}
end
```

Code 7-5: SAS detects an attack and identifies an invalid SAD

- **Abnormal commands:**

The Abnormal command is detected on the SAD side. The invariant (@Abnormal_Cmd) describes the system behaviour when SAD receives Abnormal command:

@Abnormal_Cmd: ∀ x,y· ((x ∈ channel_mntrInfo) ∧ (y ∈ channel_command) ∧ (cmd_cntnt(y) ∉ CMD_TYPE)) ⇒ mntr_cntnt(x) = Security_Alert

This invariant states that if SAD receives any Abnormal command, this implies that SAD sends a Security Alert to SAS. Also, it defines the Abnormal command as any command of a type that does not belong to the valid CMD_TYPE, where CMD_TYPE is defined by the following axioms previously defined in the contexts C0 and C1:

@Valid_Command: **partition** (CMD_TYPE, {Normal_Cmd}, {Operate_Cmd}, {Release_Cmd}, Corrective_Cmd)

@Valid_Command2: **partition** (Corrective_Cmd, {Acknowledge_powerON}, {Troubleshooting_Cmd}, {Patching_Cmd}, {Acknowledge_finishPrev}, OTHER_CORRECTIVE_CMD)

This invariant(@Abnormal_Cmd) is maintained by the following event and guards in Code 7-6:

@grd8: cmd_cntnt(cmd) ≠ Normal_Cmd

@grd9: cmd_cntnt(cmd) ≠ Operate_Cmd

@grd10: cmd_cntnt(cmd) ≠ Release_Cmd

@grd11: cmd_cntnt(cmd) ∉ Corrective_Cmd

```
event SAD_ExecuteSend_SecurityAlert_DetectAbnormal_Cmd extends SAD_ExecuteSend_OtherMntrType
where //correspond to REQ-10 in Table 7-1
   @grd8: cmd_cntnt(cmd) ≠ Normal_Cmd
   @grd9: cmd_cntnt(cmd) ≠ Operate_Cmd
   @grd10: cmd_cntnt(cmd) ≠ Release_Cmd
   @grd11: cmd_cntnt(cmd) ∉ Corrective_Cmd
   @grd12: cmdid(cmd) = sad
   @grd13: mntr_cntnt(mntr) = Security_Alert
   @grd14: mntrid(mntr) = sad
End
```

Code 7-6: SAD detects Abnormal commands

The following events in Code 7-7 describe how SAS responds if a Security Alert is received as monitoring information from SAD. According to this research, receiving commands from an invalid SAS or an invalid command from a valid SAS is considered a security breach. Therefore, SAS sends the previous command and adds the interacting party or the invalid SAS to the Blocked_AccessPoint, as shown in the event (**event** SAS_CheckMntrSend_Cmd_SecurityAlert). The system keeps a record of each command have been sent in the set variable previous_cmd. If SAD reports a Security Alert more than three times, then SAS adds that SAD to the ManualChecked set of SADs and sends the isolate command as shown in the event (**event** SAS_CheckMntrSend_Cmd_SecurityAlert_MaxExceeded).

```
event SAS_CheckMntrSend_Cmd_SecurityAlert extends SAS_CheckMntrSend_Cmd_Failure
any //correspond to REQ-13 in Table 7-1
   sad sas
where
   @grd5: sad ∈ registered_SAD
   @grd6: sas ∈ OTHER_AccessPoint
   @grd7: mntr_cntnt(mntr) = Security_Alert
   @grd9: mntrid(mntr) = sad
   @grd10: cmd ∈ previous_cmd
   @grd11: counter (sad) < 3
then
   @act4: counter(sad) ≔  counter(sad) + 1
   @act5: Blocked_AccessPoint ≔  Blocked_AccessPoint ∪ {sas}
end
event SAS_CheckMntrSend_Cmd_SecurityAlert_MaxReached extends SAS_CheckMntrSend_Cmd_Failure
any //correspond to REQ-13 in Table 7-1
   sad sas
where
   @grd5: sad ∈ registered_SAD
   @grd6:sas ∈  OTHER_AccessPoint
   @grd7: mntr_cntnt(mntr) = Security_Alert
   @grd9: mntrid(mntr) = sad
   @grd10: cmd ∈ previous_cmd
   @grd11: counter (sad) = 3
then
    @act4: counter(sad) ≔ 0
    @act5: Blocked_AccessPoint ≔  Blocked_AccessPoint ∪ {sas}
end
event SAS_CheckMntrSend_Cmd_SecurityAlert_MaxExceeded extends SAS_CheckMntrSend_Cmd_Failure
any //correspond to REQ-14 in Table 7-1
   sad
where
   @grd5: sad ∈ registered_SAD
   @grd6: mntr_cntnt(mntr) = Security_Alert
   @grd9: mntrid(mntr) = sad
   @grd10: counter (sad) > 3
   @grd11: cmd_cntnt(cmd) = isolate_cmd
   @grd12: sad ∈ AllCmd_executed_SAD
then
   @act4: ManualChecked_SAD ≔ ManualChecked_SAD ∪ {sad}
   @act6: counter(sad) ≔ 0
end
```

Code 7-7: SAS responds to a Security Alert in the monitoring information

These events and guards in Code 7-7 preserve the system invariant @inv87:

@inv87: $\forall$ x· ((mntr_cntnt(x) = Security_Alert) $\land$ (counter(mntrid(x))>3)) $\Rightarrow$ mntrid(x) $\in$ ManualChecked_SAD.

- **State irrelevant commands:**

According to the system requirements, it is abnormal for operational SADs to receive an Operate command, likewise standby SADs are not supposed to receive a Release command. So, the received command is irrelevant to the SAD's state. The following events in Code 7-8 address this abnormality where SAD sends a Security Alert to SAS.

```
event SAD_ExecuteSend_SecurityAlert_AbnormalReleaseCmd extends SAD_ExecuteSend_OtherMntrType
where //correspond to REQ-11 in Table 7-1
   @grd8: cmd_cntnt(cmd) = Release_Cmd
   @grd9: cmdid(cmd) = sad
   @grd10: mntr_cntnt(mntr) = Security_Alert
   @grd11: mntrid(mntr) = sad
   @grd12: sad ∈ standby_SAD
end

event SAD_ExecuteSend_SecurityAlert_AbnormalOperateCmd extends SAD_ExecuteSend_OtherMntrType
where //correspond to REQ-12 in Table 7-1
   @grd8: cmd_cntnt(cmd) = Operate_Cmd
   @grd9: cmdid(cmd) = sad
   @grd10: mntr_cntnt(mntr) = Security_Alert
   @grd11: mntrid(mntr) = sad
   @grd12: sad ∈ operational_SAD
end
```

Code 7-8: State irrelevant commands

7.2 Classification of the Security Requirements and Controls

After incorporating the security controls that address the abnormality in the previous section 7.1, this section continues incorporating more security controls that were identified through the literature review and NIST guidelines, and were confirmed by 12 field experts as documented in Table 5-1 and Figure 5-1.

This section starts by classifying the confirmed controls in order to conclude the most essential list of security controls that will be modelled formally using Event-B in the second refinement. The following classification method is followed to establish these list of controls:

Step 1: Applying the MoSCoW method to analyse and classify the security requirements to conclude the list with the "Must" category, which is compatible with SAS and SAD access points.

Step 2: Grouping of the "Must" list according to multiple criteria, including:

- Purpose/technique

- Compatibility with the research's scope and feasibility of Event-B in the context of this research, considering time limitations.

Step 1: Applying the MoSCoW method

The MoSCoW is a prioritisation technique for software requirements, and it was adopted to structure and rank the system requirements of this research (Fitsilis *et al.*, 2010; Achimugu *et al.*, 2014; Wiegers, 2021). This method serves to allocate requirements to the four priority categories: "MUST have", "SHOULD have", "COULD", and "WON'T have", of which "MUST have" has the highest and "COULD" has the lowest priority.

MoSCoW was applied according to the expert reviews conducted. Expert reviews' scripts were reviewed to assign each security control requirement to the status of M, S, C, or W, indicating Must, Should, Could, or Won't, respectively. The classification considers the compatibility with the SAS and SAD access points. The result of this method is shown in Table 7-2.

Table 7-3: MoSCoW classification analysis for the security controls and requirements

Security Control	**Security Requirement**	**Classification Status**
1. The system has to hash the information	Authentication	M
2. The access point has to have the Physically Unclonable Functions (PUF) which are lightweight hardware-based authentication	Authentication	C
3. The system has to hash the information using the MAC-attached cryptographic function or the HORS signature.	Authentication	M
4. The system has to exchange digital certificates using Secure Socket Layer (SSL) certificates to authenticate data transmitted between servers, systems, applications, and end-users.	Authentication	M
5. The system has to have a multi-actor authentication mechanism	Authentication	W
6. The system has to enforce automatic lockouts, whichrepresents a limit on the number of consecutive invalid login attempts by a user in a given time. Both time and number of tries are organisation-defined	Authentication	W
7. The system has to have a secure session management for multiple users' sessions for each request.	Authentication	S
8. The system has to have an anti-spoofing algorithm	Authentication	S
9. The system has to encrypt the data using attribute-based encryption	Authorisation	M
10. The system has to encrypt the data using attribute certificates	Authorisation	M
11. The system has to encrypt the data using attribute-based access control system based on XACML (Extensible Access Control Markup Language)	Authorisation	M

12. The system has to encrypt the data using Role-Based Access Control (RBAC)	Authorisation	M
13. The system has (allow/block) listing	Authorisation	S
14. Privileged Access Management (PAM)	Authorisation	S
15. Principle of Least Privilege (POLP)	Authorisation	S
16. The system has to use symmetric and asymmetric algorithms and Public Key Infrastructure certificates (PKIs)	Confidentiality	M
17. Anonymisation	Privacy	C
18. Trusted aggregator	Privacy	C
19. Encryption	Privacy	M
20. Cryptographic hashing functions and session keys	Integrity	M
21. Digital watermarking	Integrity	S
22. Adaptive cumulative sum algorithm	Integrity	C
23. Secure Phasor Measurement Units (PMUs) installation	Integrity	S
24. Load profiling algorithms	Integrity	S
25. Timestamps	Integrity	S
26. Sequence numbers	Integrity	C
27. Query sanitisation	Integrity	C
28. Nonces	Integrity	C
29. Use multiple alternate frequency channels according to a hard-coded sequence	Availability	S
30. Anomaly Intrusion Detection Systems (IDS)	Availability	S
31. Specification-based IDS	Availability	C
32. Intrusion Prevention Systems (IPS)	Availability	S
33. Quality of Services (QoS)	Availability	C
34. Load balancing	Availability	S
35. Operating system-independent applications	Availability	C
36. Redundancy	Availability	S
37. Web Application Firewall (WAF)	Availability	S
38. Anti-DDOS algorithm	Availability	S
39. Segregation, Segmentation, Data Diode isolation, DMZ, and Air gap	Availability	S
40. Mutual Inspection technique	Non-repudiation	C
41. Unique keys and digital signatures	Non-repudiation	M
42. Transaction log	Non-repudiation	S
43. Patch management for flaw remediation	Common1	S
44. Firewalls	Common2	S
45. Endpoint for Detection and Response (EDR)	Common3	C

The concluded list of security controls and requirements in the “Must” category is shown in Table 7-3:

Table 7-4: Security controls and requirements in the "Must" category

Security Control	**Security Requirement**	**Classification Status**
1. The system has to hash the information.	Authentication	M
2. The system has to hash the information using the MAC-attached cryptographic function or the HORS signature.	Authentication	M
3. The system has to exchange digital certificates using Secure Sockets Layer (SSL) certificates) to authenticate data transmitted between servers, systems, applications, and end-users.	Authentication	M
4. The system has to encrypt the data using attribute-based encryption	Authorisation	M
5. The system has to encrypt the data using the attribute certificates	Authorisation	M
6. The system has to encrypt the data using the attribute-based access control system based on XACML (Extensible Access Control Markup Language)	Authorisation	M
7. The System has to encrypt the data using the Role-Based Access Control (RBAC)	Authorisation	M
8. The system has to use Symmetric and asymmetric algorithms and Public Key Infrastructure (PKI) certificates	Confidentiality	M
9. Encryption	Privacy	M
10. Cryptographic hashing functions and session keys	Integrity	M
11. Unique keys and digital signatures	Non-repudiation	M

Step 2: Grouping the “Must” list by multiple criteria

It is clear from Table 7-2 that many cryptographic mechanisms and signature certificates have a common purpose and can be grouped under the heading of encryption techniques. According to the findings of the expert reviews in Chapter 5, encryption is widely adopted, especially in IoT frameworks, and basic encryption should be sufficient as a security control to achieve many security requirements such as Authentication, Authorisation, Confidentiality, Privacy, Integrity, and Non-Repudiation.

It is noted in Table 7-2 that one control could serve and achieve many security requirements. Many controls are mapped to Authentication and Authorisation, indicating that these two security requirements are essential implementations.

Encryption, Authentication and Authorisation measures are feasibly modelled in Event-B, and it is compatible with the targeted system, namely the interaction between SAS and SAD.

Therefore, the security requirements and controls are summarised and grouped in Table 7-5:

Table 7-5: Selected security requirements and controls of the second refinement

Security Controls	**Classification Status**	**Security Requirements**
1. Encryption	M	Authentication Authorisation Confidentiality Privacy Integrity Non-Repudiation
2. Authentication measures	M	Authentication
3. Authorisation measures	M	Authentication

Those selected security requirements and controls are later modelled formally using Event-B to introduce the most essential security controls in the second refinement of the formal model. The "Should" and "Could" categories shown in Appendix D (Tables D-1 and D-2) represent advanced security controls for further refinements in future work.

7.3 Incorporate Encryption in the Formal Model and Attack Modelling

This section involves the notation of attacks, and uses formal modelling to prove that encryption mitigates them. In Chapter 3, STRIDE threat analysis was conducted to identify internet-based threats systematically and map them to the security controls and requirements. As seen in the findings in Chapter 5, the mapping matrices were confirmed by field experts for each access point of the system. This section formalises two types of identified threats as a sample to represent the malicious actor behaviour described in the next subsection.

Next, this refinement incorporates symmetric key encryption as a basic cryptographic security control. Further refinement can develop more advanced encryption, such as asymmetric encryption or the Advanced Encryption Standard (AES).

7.3.1 Vulnerability and Attack Scenario

As presented in Figure 7-5, the communication channel formulated in the current formal model is insecure. Any security violation can disclose the monitoring information and commands transmitted via this channel. At this stage, any attack can access critical information (monitoring information/commands) in the channel as it is in plain text and no key is used to encode it.

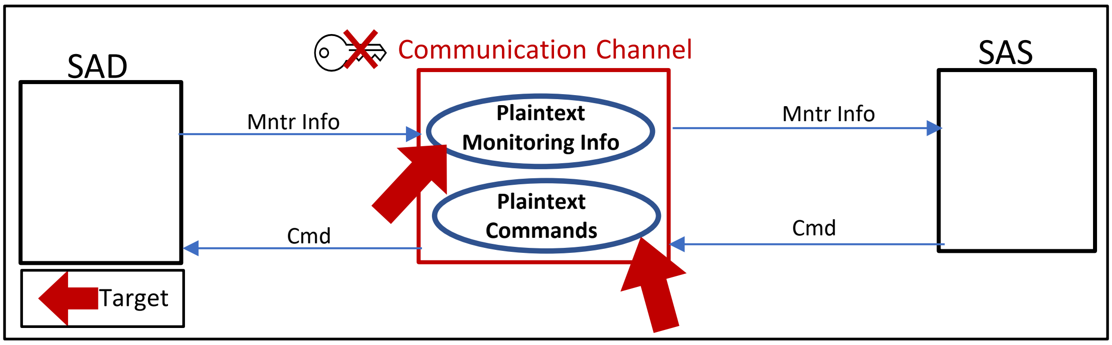


Figure 7-5: Potential targets of cyber attacks

The channel is susceptible to both passive and active types of attack. Accordingly, two types of attacks are formalised as samples: Eavesdropping and Data Tampering. An attacker may hack the registered SAD. Then, that attacker can gain access and intercept the commands channel and monitoring channel to build knowledge about transmitted information. The risk of counterfeit monitoring information can lead to incorrect decisions or state estimation due to false measurements, and may put SAD out of synchronisation.

Nevertheless, the attacker is aware that the command channel controls SADs. So, the attacker can manipulate the information in the commands channel transmitted between SAD and SAS, causing faulty commands that cause damage and dangerous controlling actions in the stations. One example of such cases is changing the command from a supposed troubleshooting to another type.

7.3.2 System Requirements for Encryption

At this level of the second refinement, encryption is introduced as a security control. An attacker may not be able to disclose the monitoring information and commands, even if they can access the information flow, as shown in Figure 7-6. Before sending information, each command and monitoring information is encrypted using the key. Then on the receiving end, command and monitoring information is decrypted using the same key to be processed.

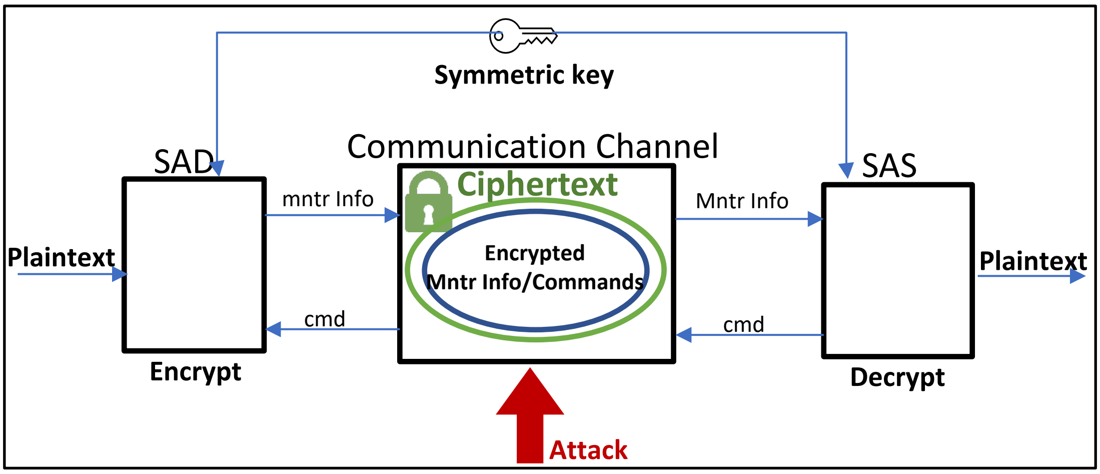


Figure 7-6: Symmetric key encryption for the information flow in the communication channel

7.3.3 Formal Model of Encryption and Attack

- **Modelling assumption**

The encryption is formalised on the basis of the following assumptions:

- The symmetric key is accessed only by SAS and SAD, and the attacker does not know that key. However, if an attacker compromises the key, they are able to decrypt the encrypted monitoring information and commands easily, disclose the original information and generate false monitoring information and commands. It is therefore crucial to ensure the secrecy of this key
- The attacker may compromise (impersonate) the registered_SAD and access the information in the channel, but information is not usable because it is encrypted
- The devised encryption is robust and cannot be cracked or broken by an attacker
- Quantum computing is incapable of decrypting monitoring information/commands that are protected by the developed encryption control.

- **Formal model of attack**

An attack is an abstraction of any unauthorised party interacting with the system in a way that might undermine the system’s purpose. According to the attack scenario described above, the attacker intends to eavesdrop and tamper with the commands. First, the attacker_knowledge variable is defined in the model machine to represent what the attacker knows, and this variable mirrors the attacker's capability. This is formalised by the invariant:

@inv92: attacker_knowledge $\subseteq$ channel_command

This invariant ensures that the attacker's knowledge is limited to the data in the command channel (channel_command).

In Code 7-9, the event (**event** ATK_Eavesdrop) captures the attack actions in an Eavesdropping attack where the attack builds attacker knowledge by adding the unencrypted commands to this variable. This refinement formalises the encrypted commands, as shown in the next section in the formalisation of the mitigation control. In the event (**event** ATK_Tamper), the attacker changes the unencrypted command’s value to another command in the action (unencrypted_commands(m) ≔ cmd).

```
event ATK_Eavesdrop
any m
where
   @grd1: m ∈ dom(unencrypted_commands)
then
   @act1: attacker_knowledge ≔ attacker_knowledge ∪ {unencrypted_commands(m)}
end

event ATK_Tamper
any m cmd
where
   @grd1: m ∈ dom(unencrypted_commands)
   @grd2: cmd ∈ COMMAND
then
   @act1: unencrypted_commands(m) ≔ cmd
end
```

Code 7-9: Attack modelling

- **Formal model of encryption control**

The context (**C2**) of the formal model formulates the encryption function by the axioms:

@axm55: Encryption ∈ COMMAND × KEY ↣ CYPHER_COMMAND
@axm56: Key ∈ KEY

where CYPHER_COMMAND and KEY are both carrier sets. Encryption is a carried function that relates the Cartesian product of COMMAND and KEY to the CYPHER_COMMAND KEY. So, it uses the key on the COMMAND to convert it to CYPHER_COMMAND, which is the ciphertext of the original command that is the encrypted command.

In Machine (**M2**), the following invariants in Code 7-10 describe the critical security properties of the system:

```
invariants
@inv90: encrypted_commands ∈ MESSAGE ⇸ CYPHER_COMMAND
@inv91: unencrypted_commands ∈ MESSAGE ⇸ channel_command
@inv92: attacker_knowledge ⊆ channel_command
@inv93: channel_command ∩ attacker_knowledge = ∅ //Security property: CONFIDENTIALITY of Cmd
@inv94: dom(encrypted_commands) ∩ dom(unencrypted_commands) = ∅
//The messages have either encrypted Cmds or unencrypted Cmds, it cannot have encrypted Cmds
and unencrypted Cmds at same time
@inv95: channel_command ∩ ran(unencrypted_commands) = ∅
//Channel is conatin no unencrypted Cmds
```

Code 7-10: Encryption invariants

The invariant (@inv93: channel_command ∩ attacker_knowledge = ∅) states that the intersection between the commands in the Channel and the attacker knowledge is empty (∅). This invariant is the first security property that represents the Confidentiality of the Command. The second security property is designed by the invariant:

> (@inv95: channel_command ∩ **ran**(unencrypted_commands) = ∅), which means that the Channel contains no unencrypted Commands.

The encrypted commands are defined in the invariant:

> @inv90: encrypted_commands ∈ MESSAGE ⇸ CYPHER_COMMAND

This is a partial function that relates the MESSAGE to the CYPHER_COMMAND. Similarly, the unencrypted commands are defined by the invariant (@inv91) that relates the MESSAGE to the original command (channel_command). Where MESSAGE is defined in the context (C2) as a carrier set, MESSAGE is a record that could have many properties. One of these properties is CYPHER_COMMAND.

The invariant (@inv94: **dom**(encrypted_commands) ∩ **dom**(unencrypted_commands) = ∅) states that MESSAGES have either encrypted commands or unencrypted commands, and cannot have encrypted commands and unencrypted commands at the same time.

In terms of the events, each event that involves sending a command is extended in this refinement to send the encrypted command. For example, the event of sending an Operate command is refined now to send the encrypted Operate command, as shown in the following Code 7-11:

```
event SAS_Send_OperateCmd extends SAS_Send_OperateCmd
any
m
where
   @grd8: m ∉ dom(encrypted_commands)
   @grd9: m ∉ dom(unencrypted_commands)
then
@act3: encrypted_commands(m) ≔ Encryption(cmd ↦ Key)
end
```

Code 7-11: Event of sending an encrypted Operate command

This event adds the following action (@act3: encrypted_commands(m) ≔ Encryption(cmd ↦ Key)) which encrypts the command parameter (cmd) using the key and assigns that to the encrypted_commands of the message that is the parameter (m). Given that @inv90: encrypted_commands ∈ MESSAGE ⇸ CYPHER_COMMAND.

(cmd) is a parameter of the COMMAND carrier set, and (m) is a parameter of the MESSAGES carrier set and does not belong to the domain of the encrypted_commands which are the messages encrypted before (@grd8: m ∉ **dom**(encrypted_commands)).

The same approach of encrypting the (Operate) Command is applicable when sending all other types of commands.

On the other side, SAD checks the received command, as shown in code Code 7-12:

1- If it is encrypted @grd3: m ∈ **dom**(encrypted_commands). Then, it will be decrypted according to @grd4: encrypted_commands(m) = Encryption(cmd ↦ Key) since Command is the decryption of the Encrypted Command using the key. Then, the command will be processed and added to the received set of commands by SAD (SAD_received_command) and the buffer (channel _command) will be removed.

2- If it is unencrypted as stated by guards (@grd3) and (@grd4): @grd3: m ∈ dom(unencrypted_commands) and @grd4: unencrypted_commands(m) = cmd. Then, it will be discarded @act1: channel_command ≔ channel_command \ {cmd}.

The events (**event** SAD_Receive_EncryptedCmd) and (**event** SAD_Receive_Unencrypted) will be triggered in receiving all types of Commands throughout the system.

```
event SAD_Receive_EncryptedCmd extends SAD_Receive_Cmd //Checking when receiving the Command if
it is encrypted, it will be processed
any
   m
where
   @grd3: m ∈ dom(encrypted_commands)
   @grd4: encrypted_commands(m) = Encryption(cmd ↦ Key) //Command is the decryption of the
Encrypted Command x Key
end

event SAD_Receive_Unencrypted //Checking when receiving the Command if it is unencrypted, it will
be discarded
any m sad cmd
where
   @grd1: sad ∈ registered_SAD
   @grd2: cmd ∈ channel_command
   @grd3: m ∈ dom(unencrypted_commands)
   @grd4: unencrypted_commands(m) = cmd
then
   @act1: channel_command ≔ channel_command \ {cmd}
end
```

Code 7-12: Checking that the command is encrypted on receiving

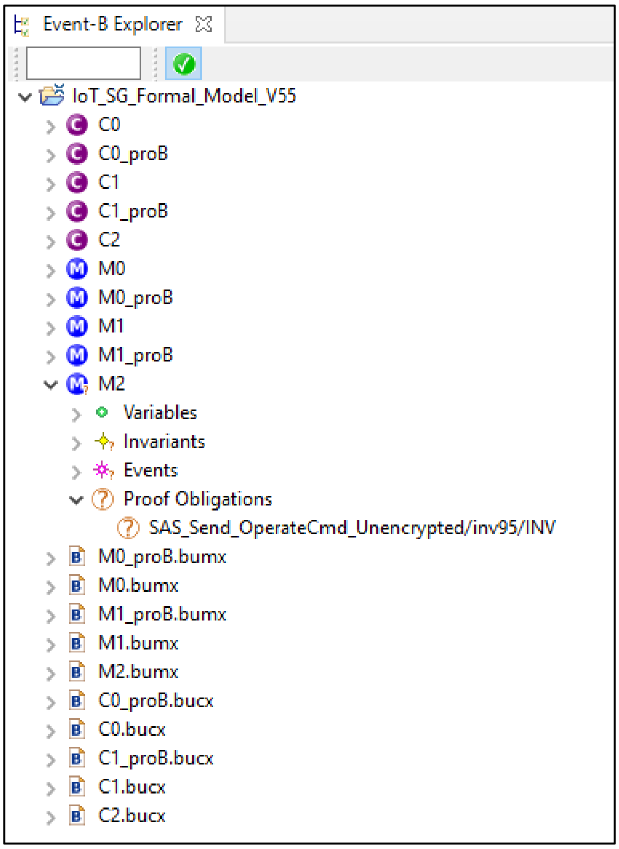
The same approach of encrypting, sending, and receiving the Commands applies to Monitoring Information.

If an event is added to send an unencrypted Command as shown in Code 7-13, then a proof obligation results on the invariant (@inv95) which means that the Channel contains no unencrypted Commands: @inv95: channel_command ∩ ran(unencrypted_commands) = ∅

```
event SAS_Send_OperateCmd_Unencrypted extends SAS_Send_OperateCmd
any
m
where
    @grd8: m ∉ dom(encrypted_commands)
    @grd9: m ∉ dom(unencrypted_commands)
then
    @act3: unencrypted_commands(m) ≔ cmd
end
```

Code 7-13:Sending an unencrypted command

The resulting proof obligation cannot be discharged, as shown in Figure 7-7. The event (**event** SAS_Send_OperateCmd_Unencrypted) is added and the obligation is included for demonstration and explanation purposes. As a result, this section proves that the designated encryption control can prevent or mitigate the modelled attacks in Code 7-9.



```
IoT_SG_Formal_Model_V55
  C0
  C0_proB
  C1
  C1_proB
  C2
  M0
  M0_proB
  M1
  M1_proB
  M2
    Variables
    Invariants
    Events
    Proof Obligations
      SAS_Send_OperateCmd_Unencrypted/inv95/INV
  M0_proB.bumx
  M0.bumx
  M1_proB.bumx
  M1.bumx
  M2.bumx
  C0_proB.bucx
  C0.bucx
  C1_proB.bucx
  C1.bucx
  C2.bucx
```

Figure 7-7: Proof obligation on sending an unencrypted command

7.4 Incorporate Authentication in the Formal Model

This section demonstrates that an Authentication control is fulfilled by the built-in features and checks that were modelled in the abstract formal model and the first refinement in Chapter 6. Those features were not discussed earlier in Chapter 6, as the focus was on the functional aspects. Now, this section highlights the security aspects that are maintained by designated guards and events.

7.4.1 Vulnerability and Misbehaviour

Authentication-related vulnerabilities may allow attackers to gain access to system information and functionality. An attacker can send incorrect monitoring information, which causes false corrective commands to be sent that may result in faults or, even worse, dangerous and conflicting actions or damage. Furthermore, inaccurate monitoring information about power may cause wasteful blackouts.

The Authentication vulnerabilities expose additional attack surfaces to further exploitation. For instance, bypassing Authentication not only exposes the system to Spoofing/Impersonation threats but makes it susceptible to every identified cybersecurity threats listed in this research, including Eavesdropping/Traffic analysis/MITM, Replay attacks, Data tampering, Denial Of Service (DOS)/Jamming channel, Malware injection, and False data injection.

7.4.2 System Requirements of Authentication

As the first step, each SAD has to be registered and assigned to a unique identification notated as registered_SAD that represents the IP/Mac address in the real system implementation. Also, Legitimate_SAS acts as a unique identity for SAS. The following Table 7-5 provides the controls that could mitigate the described system vulnerability above:

Table 7-6: System requirements of Authentication

Serial No.	System Requirement
REQ 1	SAS verifies the commissioned SAD, which sends the admission request on the valid list of SADs
REQ 2	The registration process involves authentication using unique identification, which is registered SAD
REQ 3	The retired SAD cannot register again
REQ 4	SAS cannot accept double registration

7.4.3 Formal Model of Authentication

This subsection represents the Event-B notations that devise the requirements stated in section 7.4.2.

As shown in Code 7-14, at first SAD could be either a valid set of SAD or a retired set of SAD devices. The valid set of SADs represents the list of valid manufacturing ID numbers for SADs. Then, the valid SADs could be registered SADs or unregistered SADs:

@inv4: **partition** (valid_SAD, registered_SAD, unregistered_SAD).

```
@inv1: partition (SAD, valid_SAD, retired_SAD)
@Legitimate_SAD: registered_SAD ⊆ valid_SAD
@inv3: unregistered_SAD ⊆ valid_SAD
@inv4: partition (valid_SAD, registered_SAD, unregistered_SAD)
@inv5: operational_SAD ⊆ registered_SAD
@inv6: unoperational_SAD ⊆ registered_SAD
@inv7: standby_SAD ⊆ registered_SAD
@inv9: partition (registered_SAD, operational_SAD, unoperational_SAD, standby_SAD)
```

Code 7-14: Authentication-related invariants of the registration process

This invariant (@inv4) is preserved in the registration event (**event** SAS_RegSAD_Success) specifically by the guard (@grd1: sad ∈ unregistered_SAD) that verifies that SAD belongs to the valid set to be registered successfully. This step is shown in Code 7-15 and fulfils (REQ 1).

Mainly, the registration process is considered a security layer in the model that involves authentication. If valid_SAD is successfully registered, then it is granted a unique ID (registered_SAD) for the legitimate SADs. @act1: registered_SAD ≔ registered_SAD ∪ {sad}, which fulfils (REQ 2).

The event (**event** SAS_RegSAD_Success) besides the guard (@grd1), and the action (@act1) are preserving the invariant that defines the Legitimate_SAD:

@Legitimate_SAD: registered_SAD ⊆ valid_SAD.

```
event SAS_RegSAD_Success //correspond to REQ-2 in Table 7-5
any
   sad
where
   @grd1: sad ∈ unregistered_SAD //correspond to REQ-1 in Table 7-5
   @grd2: sad ∈ SAS_rcv_admRequest
   @grd3: sad ∉ channel_AdmRequest
then
   @act1: registered_SAD ≔ registered_SAD ∪ {sad}
   @act2: standby_SAD ≔ standby_SAD ∪ {sad}
   @act3: SAS_rcv_admRequest ≔  SAS_rcv_admRequest ∖ {sad}
   @act4: unregistered_SAD ≔ unregistered_SAD ∖ {sad}
end
event SAS_RegSAD_Fail //correspond to REQ-3, REQ-4 in Table 7-5
any
   sad
where
   @no-double-registered: sad ∈ registered_SAD ∨ sad ∉ valid_SAD
   @grd3: sad ∈ SAS_rcv_admRequest
then
   @act2: SAS_rcv_admRequest ≔  SAS_rcv_admRequest ∖ {sad}
end
```

Code 7-15: Highlighting Authentication in the registration process

REQ 3 is fulfilled by the addition of invariants to the machine element of the model to constrain variables: invariant (@inv1) indicates that the variables of valid SADs and retired SADs may not overlap: @inv1: **partition** (SAD, valid_SAD, retired_SAD) this partition means that valid_SAD ∩ retired_SAD = ∅ This partition also implies that registered SADs and retired SADs do not overlap, because the registered SAD ⊆ valid_SAD.

REQ 4 is fulfilled by the addition of guards to the registration event to narrow the circumstances in which it may occur. The model restricts the registration event (**event** SAS_RegSAD_Success) through the guard below to ensure that retired SADs may not be re-registered:

@grd1: sad ∈ unregistered_SAD.

REQ 4, too, is fulfilled by the addition of guards to the registration event to prevent double registration for SADs and also allows only a valid list of SADs to be registered, These are maintained by the following guard in the event (**event** SAS_RegSAD_Fail):

@no-double-registered: sad ∈ registered_SAD ∨ sad ∉ valid_SAD

7.5 Incorporating Authorisation into the Formal Model

This section demonstrates that Authorisation and access control are fulfilled by the built-in features and checks modelled in the abstract formal model and the first refinement in Chapter 6.

In addition, this section addresses some security failures in the second refinement related to illegitimate SAD and illegitimate SAS, as represented in the hierarchy in Figure 7-1.

7.5.1 Vulnerability and Misbehaviour

Authorisation vulnerabilities may allow attackers to access system resources, including equipment/access points including SAD and SAS, and information flow. Authorisation vulnerabilities are mostly associated with privilege escalation due to insufficient privilege management and verification within the system.

Unauthorised SAD or SAS can gain the rights to send and receive commands and monitoring information. Authorisation vulnerabilities allow SADs in incorrect states to access functions they should not have access to. Nevertheless, SAD's privileges can be escalated to SAS privileges, which eventually leads to control of the entire grid.

7.5.2 System Requirements of Authorisation

To mitigate the unwanted behaviour described above, critical guards are added for checking purposes according to the requirements in Table 7-6:

Table 7-7: System requirements of Authorisation

Serial No.	System Requirement
REQ 1	SAS verifies that SAD is registered_SAD to be granted privileges of sending/receiving commands and monitoring information
REQ 2	SAS checks that the SAD belongs to the correct state of SAD corresponding to sending the correct type of monitoring information and receiving the correct type of commands
REQ 3	If SAS identifies an illegitimate SAD, then an attack is detected and SAS blocks this access point in the Blocked_AccessPoint set
REQ 4	If SAD receives a command from an illegitimate SAS, then SAD sends a Security Alert
REQ 5	If SAS receives a Security Alert in the monitoring information, then SAS sends the previous command and blocks the communicated access point
REQ 6	If SAS receives a Security Alert in the monitoring information more than three times, then SAS sends the isolate command to SAD for manual checks

7.5.1 Formal Model of Authorisation

Mutual authorisation is required for any interaction (sending/receiving) between SAS and SAD. In each event of sending or receiving, SAS checks that the SAD belongs to registered_SAD. That is to verify the ID of the interacted SAD. Also on the SAD side, SAD checks that the SAS is legitimate.

For example, REQ 1 is fulfilled in the event (**event** SAS_CheckMntrSend_Cmd_OP_TransFailure) in Code 7-16 by the addition of the guard (@grd1: sad ∈ operational_SAD). This guard implies that the SAD belongs to the registered_SAD due to the following invariant:

@inv9: **partition** (registered_SAD, operational_SAD, unoperational_SAD, standby_SAD)

So, SAS checks that the SAD belongs to the registered set of SADs, before sending the Corrective command.

The same approach is applied to all events that send commands.

```
event SAS_CheckMntrSend_Cmd_OP_TransFailure
any
   sad
   cmd
   mntr
   corCmd
where
   @grd1: sad ∈ operational_SAD //correspond to REQ-1 in Table 7-6
   @grd2: cmd ∈ COMMAND
   @grd3: corCmd ∈ Corrective_Cmd
   @grd6: mntr ∈ SAS_received_mntrInfo
   @grd7: mntr_cntnt(mntr) ∈ Transient_failure
   @grd8: mntrid(mntr) = sad
   @grd9: cmd_cntnt(cmd) = corCmd
   @grd10: cmdid(cmd) = sad
   @grd11: sad ∉ Release_executed_SAD
   @grd12: sad ∈ AllCmd_executed_SAD
then
   @act1: unoperational_SAD ≔ unoperational_SAD ∪ {sad}
   @act2: operational_SAD ≔ operational_SAD ∖ {sad}
   @act3: channel_command ≔ channel_command ∪ {cmd}
   @act4: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∖ mntr}
   @act7: AllCmd_executed_SAD ≔ AllCmd_executed_SAD ∖ {sad}
   @act8: previous_cmd ≔ previous_cmd ∪ {cmd}
end
```

Code 7-16: Sending a corrective command to the operational SAD with transient failure

Nevertheless, REQ 2 is fulfilled through the addition of the guard that checks that the SAD belongs to the correct state of registered SADs, whether it is standby_SAD, operational_SAD, or unoperational_SAD. For instance, sending the Operate command includes a guard to check that SAD belongs to the standby_SAD as it is the state of SAD that SAD is supposed to be in before it is operated, as shown in Code 7-17.

```
event SAS_Send_OperateCmd
any
   sad
   cmd
where
   @grd1: sad ∈ standby_SAD //correspond to REQ-2 in Table 7-6
   @grd2: cmd ∈ COMMAND
   @grd3: cmd ∉ channel_command
   @grd4: fault = {Overload}
   @grd5: sad ∉ Operate_executed_SAD
   @grd6: cmd_cntnt(cmd) = Operate_Cmd
   @grd7: cmdid(cmd) = sad
then
   @act1: channel_command ≔ channel_command ∪ {cmd}
   @act2: previous_cmd ≔ previous_cmd ∪ {cmd}
end
```

Code 7-17: Sending the Operate command to the correct state of SAD (standby SAD)

REQ 3 is fulfilled by the invariant below and the events and guards that were discussed before in attack detection in Code 7-5:

@illegitimate_SAD: ∀ x· ((x ∈ channel_mntrInfo) ∧ (mntrid(x) ∉ valid_SAD)) ⇒ mntrid(x) ∈ Blocked_AccessPoint

In terms of REQ 4, REQ 5, and REQ 6, these requirements are related to addressing the illegitimate SAS. As presented in Code 7-18, SAD checks that SAS is legitimate to receive the command successfully by the guard (@grd3: sas ∈ {Legitimate_SAS}). Otherwise, the command is not received and a Security Alert is reported by SAD.

Therefore, the event (**event** SAD_Receive_Cmd) is refined in the second refinement to have a successful and failed receiving process, where legitimate SAS is defined by the axiom (@axm54: {Legitimate_SAS} ⊆ SAS) in the Context (C2).

```
event SAD_Receive_Cmd_Success extends SAD_Receive_Cmd
any //correspond to REQ-4 in Table 7-6
   sas
where
   @grd3: sas ∈ {Legitimate_SAS}
end
event SAD_Receive_Cmd_Fail extends SAD_Receive_Cmd_Failure
any //correspond to REQ-4 in Table 7-6
   sas
   sad
where
   @grd1: sad ∈ registered_SAD
   @grd3: sas ∉ {Legitimate_SAS}
   @grd5: mntr_cntnt(mntr) = Security_Alert
   @grd6: mntrid(mntr) = sad
end
```

Code 7-18: Refinement events of receiving commands by SAD

The response of SAS for the reported Security Alert is modelled and discussed earlier, in Code 7-7, and it is maintained by the invariant:

@inv87: ∀ x· ((mntr_cntnt(x) = Security_Alert) ∧ (counter(mntrid(x))>3)) ⇒ mntrid(x) ∈ ManualChecked_SAD

7.6 Conclusion of the Formal Verification and Validation of the Model

The Rodin platform provides a supportive environment for both modelling and proving, through theorem proving and model checking. As a result of using the Rodin platform, the proposed Event-B model was proved to be correct and consistent. Table 7-8 and Figure 7-8 present an overview of the proof statistics provided by Rodin. These statistics measure the Proof Obligations (PO) generated and discharged by the Rodin prover and the POs that are interactively proved. The complete development of the (SAD-SAS) interaction results in 837 POs, all of which (100%) are proved automatically by Rodin. The number of POs in the system abstraction that captures the functionality is relatively close to the first refinement, while the POs of the second refinement are greater than the other refinements. This is because the main components of the security properties have been established, therefore many invariants are introduced in that layer to guarantee the correctness of these components.

Table 7-8: Proof statistics using the Rodin platform

Event-B Component	**Total**	**Auto**	**Manual**	**Undischarged**
Total	836	528	308	0
M0	177	167	10	0
M1	183	161	22	0
M2	476	200	276	0
C0	0	0	0	0
C1	0	0	0	0
C2	0	0	0	0

The main aim is to present a generic model that can be used to develop further formal models for the information flow between other access points in the IoT-enabled Smart Grid, and implement further security controls. Each refinement aids in constructing a finer design of the targeted system.

In theorem proving, different POs are generated by Rodin during the development of the targeted system. The Rodin's proof POs were used to verify that different machines adheres to its invariants and correctly models the system scenarios, ensuring that the system actions are consistent with the system requirements and are without ambiguity.

An example of a PO is demonstrated by invariant preservation. The INV PO ensures that each invariant is preserved by each event. The INV PO is introduced on the invariant (@inv89):

```
@inv89: ∀ x· ((mntrid(x) ∈ compromised_SAD) ∧ (counter(mntrid(x))>3)) ⇒ mntrid(x) ∈ ManualChecked_SAD
```

As shown in the following Code 7-19, the obligation was introduced in the event (SAS_CheckMntrSend_Cmd_compromised_Patching_Fail_AbnormalMntr) because, after executing this event three times, the maximum was reached and the counter was incremented, while according to the (@inv89) it should be reset to (0).

To discharge this obligation, a third event was added to model the case when the maximum is reached, as presented in Code 7-20:

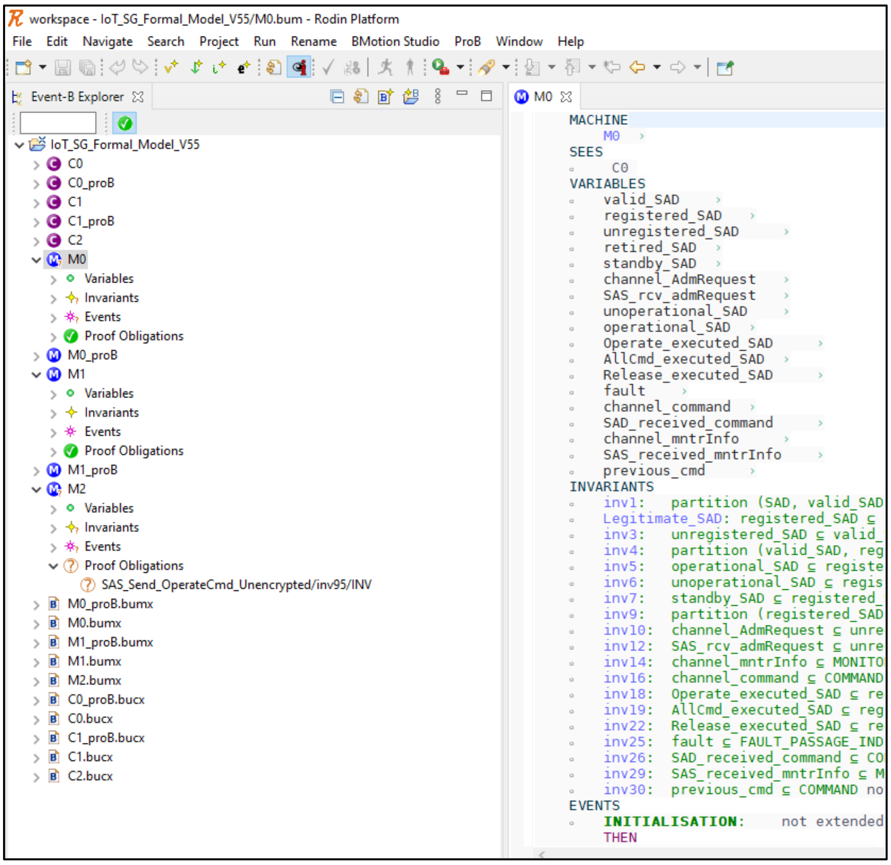
```
event SAS_CheckMntrSend_Cmd_compromised_Patching_Fail_AbnormalMntr extends
SAS_CheckMntrSend_Cmd_Corrective_Fail
where
   @grd7: mntr_cntnt(mntr) ≠ Normal_info
   @grd8: sad ∈ compromised_SAD
    @grd9: cmd_cntnt(cmd) = Patching_Cmd
    @grd10: counter (sad) ≤ 3
    @grd11: sad ∈ AllCmd_executed_SAD
 then
   @act4: counter(sad) ≔  counter(sad) + 1
end
event SAS_CheckMntrSend_Cmd_compromised_Patching_Fail_AbnormalMntr_MaxExceeded
extends SAS_CheckMntrSend_Cmd_Corrective_Fail
where
    @grd7: mntr_cntnt(mntr) ≠ Normal_info
    @grd8: sad ∈ compromised_SAD
    @grd9: cmd_cntnt(cmd) = isolate_cmd
    @grd10: counter (sad) > 3
    @grd11: sad ∈ AllCmd_executed_SAD
then
    @act4: ManualChecked_SAD ≔ ManualChecked_SAD ∪ {sad}
   @act8: counter(sad) ≔ 0
end
```

Code 7-20: Event before discharging the obligation

```
event SAS_CheckMntrSend_Cmd_compromised_Patching_Fail_AbnormalMntr extends
SAS_CheckMntrSend_Cmd_Corrective_Fail
where
   @grd7: mntr_cntnt(mntr) ≠ Normal_info
   @grd8: sad ∈ compromised_SAD
   @grd9: cmd_cntnt(cmd) = Patching_Cmd
   @grd10: counter (sad) < 3
   @grd11: sad ∈ AllCmd_executed_SAD
then
   @act4: counter(sad) ≔  counter(sad) + 1
end
event SAS_CheckMntrSend_Cmd_compromised_Patching_Fail_AbnormalMntr_MaxReached extends
SAS_CheckMntrSend_Cmd_Corrective_Fail
where
   @grd7: mntr_cntnt(mntr) ≠ Normal_info
   @grd8: sad ∈ compromised_SAD
   @grd9: cmd_cntnt(cmd) = Patching_Cmd
   @grd10: counter (sad) = 3
   @grd11: sad ∈ AllCmd_executed_SAD
then
   @act4: counter(sad) ≔ 0
end
event SAS_CheckMntrSend_Cmd_compromised_Patching_Fail_AbnormalMntr_MaxExceeded
extends SAS_CheckMntrSend_Cmd_Corrective_Fail
where
   @grd7: mntr_cntnt(mntr) ≠ Normal_info
   @grd8: sad ∈ compromised_SAD
   @grd9: cmd_cntnt(cmd) = isolate_cmd
   @grd10: counter (sad) > 3
   @grd11: sad ∈ AllCmd_executed_SAD
then
   @act4: ManualChecked_SAD ≔ ManualChecked_SAD ∪ {sad}
   @act8: counter(sad) ≔ 0
end
```

Code 7-19: Discharge the obligation by addition of an event

In model checking, the model is validated using ProB. ProB is an animator and a model checker for Event-B. ProB allows fully automatic exploration of Event-B models and can be used to check systematically a specification for a range of errors. This model was analysed using ProB to ensure that the model is deadlock-free. Each new event added to the refinements was verified to ensure that it did not introduce a deadlock and that the events met the expected behaviour and scenario. The model was animated using ProB, and it follows and fulfils the expected state machine.



```
MACHINE
    M0
SEES
    C0
VARIABLES
    valid_SAD
    registered_SAD
    unregistered_SAD
    retired_SAD
    standby_SAD
    channel_AdmRequest
    SAS_rcv_admRequest
    unoperational_SAD
    operational_SAD
    Operate_executed_SAD
    AllCmd_executed_SAD
    Release_executed_SAD
    fault
    channel_command
    SAD_received_command
    channel_mntrInfo
    SAS_received_mntrInfo
    previous_cmd
INVARIANTS
    inv1:   partition (SAD, valid_SAD
    Legitimate_SAD: registered_SAD ⊆
    inv3:   unregistered_SAD ⊆ valid_
    inv4:   partition (valid_SAD, reg
    inv5:   operational_SAD ⊆ registe
    inv6:   unoperational_SAD ⊆ regis
    inv7:   standby_SAD ⊆ registered_
    inv9:   partition (registered_SAD
    inv10:  channel_AdmRequest ⊆ unre
    inv12:  SAS_rcv_admRequest ⊆ unre
    inv14:  channel_mntrInfo ⊆ MONITO
    inv16:  channel_command ⊆ COMMAND
    inv18:  Operate_executed_SAD ⊆ re
    inv19:  AllCmd_executed_SAD ⊆ reg
    inv22:  Release_executed_SAD ⊆ re
    inv25:  fault ⊆ FAULT_PASSAGE_IND
    inv26:  SAD_received_command ⊆ CO
    inv29:  SAS_received_mntrInfo ⊆ M
    inv30:  previous_cmd ⊆ COMMAND no
EVENTS
    INITIALISATION:    not extended
    THEN
```

Figure 7-8: Proof obligations status from the Rodin platform

7.7 Summary

This chapter demonstrates that the developed model maintains the appropriate security controls to mitigate the internet-based threats to the information flow between SAS and SAD in the IoT-enabled Smart Grid. This is particularly as a formal template to allow field experts and engineers to verify any changes made so that they do not compromise the security of the information flow of the targeted system. This template could be reused by the field experts in mitigating any new threats on additional access points or maintaining new security controls. Formal modelling helps to control the complexity and assure the consistency of the system requirements. Three levels of refinements are presented in the last two chapters by the CONTEXT and MACHINE: **C0, C1, and C2**, as well as **M0, M1, and M2**.

Chapter 8 **Conclusions and Future Work**

The current research examines the requirements and controls that secure the information systems against security threats to the exchange of data between all access points in the IoT-enabled Smart Grid. This chapter presents the research undertaken thus far in answering the research questions and the objectives (a to e), and presents the conclusions revealed in the findings of the expert reviews. Then, it describes the results of the verification and validation process for the Information Security Model through formal methods. Finally, the future work of this research is outlined.

8.1 Conclusions

The primary Research Question (RQ) addressed in this research is "**What is an appropriate model for a secure information system for an IoT-enabled Smart Grid in the Saudi energy sector?**". In order to answer this, the RQ was divided into three subquestions and five objectives. All these objectives (a to e) have been answered, as can be seen below in Table 8-1, which shows the details and objectives of the methods used with linking them to research contributions and output.

First, security incidents were presented. Chapter 2 showed the significance of security and threat modelling at the system design stage of IoT-enabled Smart Grids. Although many studies have proposed models and solutions to maintain information security in Smart Grids, the IoT-enabled Smart Grid is still challenged, and security threats persist. The research conducted a comprehensive review of the literature on IoT-enabled SGs and identified the primary security challenges, requirements, and controls associated with these systems. Key studies and standards, such as those from IEEE, US Department of Energy, and NIST, provided foundational insights into the definitions, architectures, and security concerns of SGs. Further, in section 2.9., the research surveyed the main challenges that the energy sector faces in securing the bi-directional flow of information. It demonstrated that the IoT-enabled Smart Grid has several connected entities, alongside increasing demand for electric energy, resulting in many significant challenges for the IoT-enabled SG. Security was shown to be the main challenge. Previous studies (Bekara 2014; Ghasempour 2019) highlighted challenges such as interoperability, scalability, heterogeneity, and security vulnerabilities in IoT-enabled SGs. This research corroborates these findings and further refines them by identifying specific threats and controls through a detailed threat modeling process.

Table 8-1: Summary of methods used for the research aims and objectives, showing research contributions and output

Research Aims	Contribution	Objective	Method	Research output
1. To investigate the main challenges of the IoT-enabled Smart Grid.	**Contribution 1** • Conducting a detailed Literature review on challenges, access points, threats, and controls • Conducting Threat Analysis to conclude the controls that support the security requirements against the internet-based threats	a. To survey the main challenges that the electric energy sector faces in securing the bi-directional flow of information.	Literature review. **(Chapter 2)**	Conference paper2 **zenodo.org, 2022**
2. To develop a security model for the information flow around the IoT-enabled SG.		b. To review the security requirements that IoT-enabled Smart Grid needs to fulfil.	Literature review and international standards for Smart Grids. **(Chapter 3)**	Conference paper1 **SciTePress, 2022**

		c. To analyse the main IoT cyber threats to information security in IoT-enabled Smart Grids in order to identify the security controls that mitigate and detect potential IoT cybersecurity threats in the electric energy sector.	Threat modelling, analysis, and literature review. **(Chapter 3)**	Conference paper3 **IEEE, 2023**
	Contribution 2 • Developing a security model for the information flow around an IoT-enabled SG.	d. To confirm the developed Information Security Model for an IoT-enabled Smart Grid, including access points, security requirements, threats, and controls.	Semi-structured Interviews with security and electricity experts. **(Chapter 5)**	Journal Paper **Elsevier, 2023**
3. To validate and verify the security model for the IoT-enabled SG.	**Contribution 3** • Utilising a unique approach combining threat analysis, expert reviews, and formal methods to develop a formal model for an IoT-enabled Smart Grid (SG). This model can serve as a foundation for further analysis of other access points and for implement new security controls.	e. 1. To demonstrate the usefulness of the proposed model to conduct a more detailed and tangible analysis.	Event-B formal method and Rodin toolkit. (**Chapter 6**)	Submitted for publication **Elsevier, 2024**

	Contribution 4 • Producing a formal generic template, to establish the foundation for security checks, allowing field experts and engineers to verify any change made, ensuring they do not compromise the security of information flow within the IoT-enabled Smart Grid.	e. 2. To verify and validate the effectiveness of the mitigation security controls to address the identified security concerns.	Event-B formal method and Rodin toolkit. **(Chapter 7)**	Submitted for publication **Energies, Special Issue of Smart Grid Cybersecurity, 2024**

Second, the research modelled the system, and this model was used to explore the security requirements. The identification of seven core security requirements (Authentication, Authorisation, Confidentiality, Privacy, Integrity, Availability, and Non-repudiation) aligns with international standards and the existing literature (NIST 2014), ensuring that the proposed model is comprehensive and adheres to established guidelines.

Third, the security threats were analysed. Threat modelling showed that the nine main common internet-based threats confronting the electric energy sector are: Spoofing/Impersonation, Eavesdropping/Traffic analysis/Man-In-The-Middle (MITM), Replay attack, Data tampering, Denial-of-service/jamming channel, Malware injection, Phishing, SQL injections, and False data injection. In addition, in section 3.2.4 this research identified 38 security controls and countermeasures that may mitigate and detect potential IoT cybersecurity threats in the IoT-enabled electricity sector. The security controls were then mapped to the requirements. Furthermore, seven access points and their interconnections were described in section 3.2.2, and identified threats and controls were assigned to each access point. As a result, in section 3.4 the Information Security Model for an IoT-enabled Smart Grid was developed. This research presents a detailed and empirically validated security model for the information flow in an IoT-enabled Smart Grid, specifically tailored to the Saudi energy sector. Unlike previous models that often remained theoretical or generalised, this model addresses practical and context-specific security concerns.

Fourth, research was conducted to confirm the information security model, including all access points, security requirements, threats, and controls. To achieve this, qualitative methods were employed alongside the literature review using a mixed-method approach. The triangulation technique was adopted to examine the aspects of the findings and to increase their level of accuracy and reliability. The interviews were semi-structured and characterised by open-ended questions. The interviews involved 14 experts in the fields of security and electricity. The interviews confirmed the developed information security model and examined whether there were any other access points, security requirements, threats, and controls that had not been mentioned or that required modification. This resulted in five controls being omitted, while two controls were split, nine controls added, and one control was re-assigned in the developed model, creating a common control list. This resulted in a confirmed information security model for the IoT-enabled Smart Grid that contains seven access points, seven security requirements, nine threats, and 45 security controls. During the expert interviews, discrepancies emerged regarding the prioritisation and effectiveness of certain security controls. While some experts emphasised the importance of specific controls, others questioned their practicality or cost-effectiveness. This variance highlights the subjective nature of security assessments and the need for adaptable security models that can be tailored to different operational or organisational contexts.

Identifying 38 specific security controls mapped to seven access points and nine common threats provides a granular understanding of the security landscape. This detailed mapping is a unique contribution, offering a practical guide for implementing security measures. By incorporating semi-structured interviews with 14 experts from the fields of security and electricity, the research provided practical insights that ensured the model’s relevance and applicability. This empirical validation is a unique aspect that many prior studies lack, which often relied solely on theoretical frameworks. The model was recommended by the field experts as it contributes to information security in such machinery stations and OT networks that represent a vital interest in the recent initiatives and direction of Saudi energy.

Fifth, formal methods were adopted to formulate the security model. Starting with formalising the system functionality and basic aspects of Availability, the abstract formal model and first refinement were developed. This was followed by formalising the required security controls. The use of Event-B formal methods to model and validate security controls is a novel approach within this context. This rigorous methodology provides a structured framework for ensuring system consistency and reliability, setting this research apart from more conventional approaches that might not offer the same level of assurance. The resulting constructed formal security model established the basic understanding model that can be reused as a base model for the IoT-enabled Smart Grid, contributing to generating and building other frameworks for other access points or components or other controls, as the field of formal methods for security is continually evolving. The creation of a formal generic template for security checks allows field experts and engineers to verify changes without compromising the security of information flow. This tool is particularly valuable for ongoing security management and has potential applications beyond the initial scope of this research.

The findings from this research have significant implications for policymaking, industry practices, and academic theory. For policymakers, the detailed security requirements and threat modeling insights offer tools to develop robust regulatory frameworks that ensure comprehensive security in IoT-enabled Smart Grids. These frameworks can include guidelines for secure IoT implementation, mandating regular security audits and adherence to best practices.

In industry, the validated security model and practical tools such as security check templates can help practitioners enhance their security measures. By adopting the identified security controls and integrating formal methods like Event-B, companies can ensure robust protection against cyber threats and maintain ongoing security management.

Theoretically, this research provides a robust foundation for future academic studies. It encourages the development of dynamic security models that evolve with emerging threats and highlights the value of interdisciplinary research. By integrating insights from cybersecurity, electrical engineering, data science, and policy studies, future research can develop comprehensive security solutions for IoT-enabled Smart Grids.

8.2 Contributions of the Research

Security threats will always arise in information systems, since attacks are becoming more advanced, yet encouraging the construction of systems that are secure by design accommodates the security requirements earlier in the development process. This research is, therefore, able to provide both breadth by combining the fields of IoT, SG and information security, and depth by focusing on a detailed security model. This work contains the following useful contributions to the research community:

1- A detailed literature review was conducted on both industrial standards and academic publications that identified the main access points of an IoT-enabled Smart Grid. Then, threat analysis was undertaken on the information flow around an IoT-enabled Smart Grid. The gap analysis showed that few studies comprehensively address security controls against the internet-based threats to the information flow around the IoT-enabled Smart Grid, either generally or specifically in KSA’s electricity sector.

2- A security model for the information flow around an IoT-enabled Smart Grid was developed. The model identified the security controls required to mitigate internet-based threats, taking a comprehensive, structured approach to fill in the missing details in the NIST conceptual model. This model was then confirmed by interviewing experts in Saudi Arabia on the IoT-enabled Smart Grid.

3- The aforementioned points (1) and (2) were published in the journal paper.

4- A unique approach combining threat analysis, expert reviews, and formal methods was utilised to develop a formal model for an IoT-enabled Smart Grid (SG). This model can serve as a foundation for further analysis of other access points and for implement new security controls.

5- A formal generic template was produced, to establish the foundation for security checks, allowing field experts and engineers to verify any change made, ensuring they do not compromise the security of information flow within the IoT-enabled Smart Grid.

6- The aforementioned points (4) and (5) were submitted for publication as the second and third journal papers, one of which being published in a Special Issue of Smart Grid Cybersecurity.

8.3 Discussion of Challenges and Limitations

In conducting this research to develop the Information security model of the IoT-enabled Smart Grid several challenges and limitations were encountered. Challenges are related to Technical Challenges, Human and Organizational Challenges, and Methodological Challenges. Addressing these challenges is vital to understand the context and implications of our findings fully:

Technical Challenges

Complexity and Scalability the inherent complexity and scalability issues of IoT-enabled Smart Grids presented significant challenge. To address this, threat modeling and formal methods abstraction were employed to systematically analyse and model the security requirements and controls. This approach helped manage the complexity and provide scalable solutions.

Heterogeneity and Interoperability the diverse nature of devices and systems within the IoT-enabled Smart Grid posed interoperability issues. Different devices and systems often use various protocols and standards, complicating the integration and security management. To overcome this challenge, an extensive literature review was conducted and aligned the final security model with international standards to ensure compatibility and interoperability across different devices and systems.

Human and Organizational Challenges

Expert Knowledge and Bias the reliance on expert opinions through semi-structured interviews introduced potential biases. Experts' subjective perspectives could influence the validation of the security model. To address this issue, triangulation techniques was employed to corroborate findings from multiple sources and reduce the impact of individual biases. Furthermore, a diverse group of experts with long years of experience were consulted to ensure a broad range of insights and perspectives from their practical experience.

Implementation and Acceptance ensuring the acceptance and implementation of the proposed security model within the industry was challenging due to existing practices and resistance to change. To mitigate this challenge, an early engagement with industry stakeholders was followed in the research process to understand their needs and concerns, incorporating their feedback into the model development. This collaborative approach facilitated a smoother acceptance and potential implementation.

Methodological Challenges

Formal Methods Limitations due to the abstraction factors of formal reasoning and the ability of the research tool of Event-B, the system specifications are subject to change and new perspectives. Therefore, the formal modelling is not unified. To mitigate this challenge, the formal models were iteratively refined, incorporating feedback from real-world applications and expert reviews to ensure they remained relevant and practical.

In addition, due to the subjective factors of security, the specifications may often encounter change. Therefore, security is not static. The subjective factor has been reduced as far as possible but, as in all human analysis, it is hard to remove it entirely. Because security is related to how it is perceived, there is no objective security. Thus, it is most likely that the same scenario, if analysed by another researcher, would lead to slightly different specifications.

Generalisability the specific focus on the Saudi energy sector may limit the generalisability of the findings to other regions or sectors. While our model is tailored to the Saudi context, it is designed with flexibility to be adapted to other regions or sectors. Future research could explore these adaptations, ensuring broader applicability.

Impact on Generalizability and Applicability

The challenges discussed above highlight areas where the research findings might face limitations. For instance, the complexity and scalability issues addressed through formal methods might require further refinement when applied to different scales or more diverse IoT environments. The potential biases from expert opinions necessitate a cautious interpretation of validation results, ensuring they are cross-verified with empirical data. Lastly, while the model's adaptability is a strength, the initial regional focus means additional contextualisation is needed for broader applicability.

In conclusion, acknowledging and addressing these challenges not only provides a clearer understanding of the research context but also strengthens the robustness and credibility of our findings. This discussion underscores the importance of continuous refinement and validation of the proposed security model, ensuring it remains relevant and effective in diverse real-world applications.

8.4 Future Work

In this research the focus is internet-based threats and their mitigation controls and security requirements. However, detailed information related to individual access points or safety, such as the requirements related to emergency lighting, fire protection, temperature and humidity controls, power equipment and power cabling, and lockout/tagout, are beyond the scope of cybersecurity and so are not covered. These requirements should be addressed by each organisation under its local and organisational regulations and policies. Equipment failures, employee errors, and natural disasters are out of scope.

Future work could be implemented to include more advanced multi-stage threats, such as Advanced Persistent Threat (APT), which is composed of many other basic threat behaviours.

The next stage will verify and validate more complex security controls by augmenting further refinements in formal methods and reasoning; this is in order to extend the security controls to include the advanced set with “should” and “could” categories. This future refinement represents upscaling protection and raises security to the medium and high level. Many future research directions from this research could be undertaken to examine the internal processes of specific access points or components in the IoT-enabled Smart Grid, such as research on SCADA modelling alone, AMI, or smart meters separately.

More work needs to be undertaken in the area of formally modelling other security threats and mitigation controls.

This model could be applied in other countries worldwide, but would require further research to investigate the customised requirements of each region.

References

Abrial, J.R. (2010) ‘Modeling in event-b: System and software engineering’, Modeling in Event-B: System and Software Engineering, 9780521895, pp. 1–586. doi:10.1017/CBO9781139195881.

Abrial, J.R. and Hallerstede, S. (2007) ‘Refinement, decomposition, and instantiation of discrete models: Application to event-B’, Fundamenta Informaticae, 77(1–2), pp. 1–28.

Achimugu, P., Selamat, A., Ibrahim, R. and Mahrin, M.N.R. (2014) ‘A systematic literature review of software requirements prioritization research’, Information and Software Technology, 56(6), pp. 568–585. doi:10.1016/j.infsof.2014.02.001.

Al-Ali, A.R. and Aburukba, R. (2015) ‘Advanced role of internet of things in the smart grid technology’, Journal of Computer and Communications, pp. 229–233. doi:http://dx.doi.org/10.4236/jcc.2015.35029 Role.

Al-Omar, B., Al-Ali, A.R., Ahmed, R. and Landolsi, T. (2012) ‘The Role of Information and Communication Technologies in’, Journal of Emerging Trends in Computing and Information Sciences, 3(5), pp. 707–716. Available at: http://search.proquest.com/docview/854286083?accountid=28962%5Cnhttp://sfx.metabib.ch/sfx_locater??url_ver=Z39.88-2004&rft_val_fmt=info:ofi/fmt:kev:mtx:dissertation&genre=dissertations+&+theses&sid=ProQ:ProQuest+Dissertations+&+Theses+A&I&atitle=&title.

Al-Turjman, F. and Abujubbeh, M. (2019) ‘IoT-enabled smart grid via SM: An overview’, Future Generation Computer Systems, 96, pp. 579–590. doi:10.1016/j.future.2019.02.012.

Aloul, F., Al-Ali, A.R., Al-Dalky, R., Al-Mardini, M. and El-Hajj, W. (2012) ‘Smart Grid Security: Threats, Vulnerabilities and Solutions’, International Journal of Smart Grid and Clean Energy, 00, pp. 1–6. doi:10.12720/sgce.1.1.1-6.

Aloul, F.A. (2012) ‘The Need for Effective Information Security Awareness’, Journal of Advances in Information Technology, 3(3), pp. 176–183. doi:10.4304/jait.3.3.176-183.

Anwar, A. and Mahmood, A. (2014) ‘Cyber security of smart grid infrastructure’, The State of the Art in Intrusion Prevention and Detection, (January), pp. 449–472. doi:10.1201/b16390-9.

Arasteh, H., Hosseinnezhad, V., Loia, V., Tommasetti, A., Troisi, O., Shafie-Khah, M. and Siano, P. (2016) ‘Iot-based smart cities: A survey’, EEEIC 2016 - International Conference on Environment and Electrical Engineering, pp. 1–6. doi:10.1109/EEEIC.2016.7555867.

Aravinthan, V., Namboodiri, V., Sunku, S. and Jewell, W. (2011) ‘Wireless AMI application and security for controlled home area networks’, IEEE Power and Energy Society General Meeting, pp. 1–8. doi:10.1109/PES.2011.6038996.

Armoogum, S. and Bassoo, V. (2018) Privacy of energy consumption data of a household in a smart grid, Smart Power Distribution Systems: Control, Communication, and Optimization. Elsevier Inc. doi:10.1016/B978-0-12-812154-2.00008-0.

Atlam, H.F. and Wills, G.B. (2019) Technical aspects of blockchain and IoT. 1st edn, Advances in Computers. 1st edn. Elsevier Inc. doi:10.1016/bs.adcom.2018.10.006.

Banerjee, A., Chitnis, U., Jadhav, S., Bhawalkar, J. and Chaudhury, S. (2009) ‘Hypothesis testing, type I and type II errors’, Industrial Psychiatry Journal, 18(2), p. 127. doi:10.4103/0972-6748.62274.

Barker, E., Chen, L., Roginsky, A., Vassilev, A., Davis, R. and Simon, S. (2019) NIST Special Publication 800-56B Revision 2 - Recommendation for pair-wise key establishment using integer factorization cryptography, NIST Special Publication. Available at: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Br2.pdf.

Bedi, G., Venayagamoorthy, G.K., Singh, R., Brooks, R.R. and Wang, K.C. (2018) ‘Review of Internet of Things (IoT) in Electric Power and Energy Systems’, IEEE Internet of Things Journal, 5(2), pp. 847–870. doi:10.1109/JIOT.2018.2802704.

Bekara, C. (2014) ‘Security issues and challenges for the IoT-based smart grid’, Procedia Computer Science, 34, pp. 532–537. doi:10.1016/j.procs.2014.07.064.

Bekara, C., Luckenbach, T. and Bekara, K. (2012) ‘A Privacy Preserving and Secure Authentication Protocol for the Advanced Metering Infrastructure with Non-Repudiation Service’, Proc. of ENERGY, (c), pp. 60–68.

Benmalek, M., Challal, Y. and Derhab, A. (2019) ‘Authentication for Smart Grid AMI Systems: Threat Models, Solutions, and Challenges’, Proceedings - 2019 IEEE 28th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE 2019, pp. 208–213. doi:10.1109/WETICE.2019.00052.

Benoit, J. (2011) An Introduction to Cryptography as Applied to the Smart Grid, Cooper Power Systems.

Bhattacherjee, A. (2012) Social Science Research: Principles, Methods, and Practices, Textbooks Collection. Available at: http://scholarcommons.usf.edu/oa_textbookshttp://scholarcommons.usf.edu/oa_textbooks/3 (Accessed: 15 November 2020).

Bhattarai, S., Ge, L. and Yu, W. (2012) ‘A novel architecture against false data injection attacks in smart grid’, IEEE International Conference on Communications, pp. 907–911. doi:10.1109/ICC.2012.6364511.

Boehm, B.W. and Papaccio, P.N. (1988) ‘Understanding and Controlling Software Costs’, IEEE Transactions on Software Engineering, 14(10), pp. 1462–1477. doi:10.1109/32.6191.

Bowen, G.A. (2008) ‘Naturalistic inquiry and the saturation concept: A research note’, Qualitative Research, 8(1), pp. 137–152. doi:10.1177/1468794107085301.

Butler, M. (2013) ‘Mastering system analysis and design through abstraction and refinement’, Engineering Dependable Software Systems, 34, pp. 49–78. doi:10.3233/978-1-61499-207-3-49.

Butler, M., Leuschel, M., Presti, S. Lo and Turner, P. (2004) ‘The Use of Formal Methods in the Analysis of Trust (Position Paper)’, in Second International Conference on Trust Management, pp. 333–339. Available at: https://eprints.soton.ac.uk/258770/.

Calderaro, V., Hadjicostis, C.N., Piccolo, A. and Siano, P. (2011) ‘Failure identification in smart grids based on Petri net modeling’, IEEE Transactions on Industrial Electronics, 58(10), pp. 4613–4623. doi:10.1109/TIE.2011.2109335.

Caldwell, T. (2018) ‘Plugging IT/OT vulnerabilities – part 1’, Network Security, 2018(8), pp. 9–14. doi:10.1016/S1353-4858(18)30078-3.

Camachi, B.E., Ichim, L. and Popescu, D. (2018) ‘Cyber security of smart grid infrastructure’, SACI 2018 - IEEE 12th International Symposium on Applied Computational Intelligence and Informatics, Proceedings, pp. 303–308. doi:10.1109/SACI.2018.8441017.

Canetti, R. and Krawczyk, H. (2002) ‘Universally composable notions of key exchange and secure channels’, in International Conference on the Theory and Applications of Cryptographic Techniques, pp. 337–351. doi:10.1007/3-540-46035-7_22.

Cawthra, J., Hodges, B., Kuruvilla, J., Littlefield, K., Bob, N., Peloquin, C., Wang, S., Williams, R. and Zheng, K. (2018) Securing Picture Archiving and Communication System (Pacs). Available at: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.1800-24.pdf.

CEN/CENELEC/ETSI (2012) CEN-CENELEC-ETSI Smart Grid Coordination Group: Smart Grid Information Security. Available at: ftp://ftp.cen.eu/EN/EuropeanStandardization/HotTopics/SmartGrids/Security.pdf.

Cichonski, P., Millar, T., Grance, T. and Scarfone, K. (2012) Computer Security Incident Handling Guide : Recommendations of the National Institute of Standards and Technology, NIST Special Publication. Available at: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-61r2.pdf.

CIS (2021) ‘The SolarWinds Cyber-Attack: What You Need to Know’, pp. 1–10. Available at: https://www.cisecurity.org/solarwinds/ (Accessed: 11 August 2021).

CISA (2005) Cyber Threat Source Descriptions | CISA. Available at: https://us-cert.cisa.gov/ics/content/cyber-threat-source-descriptions (Accessed: 14 May 2020).

CISA (2009) ‘Understanding Digital Signatures | CISA’. Available at: https://us-cert.cisa.gov/ncas/tips/ST04-018 (Accessed: 10 August 2020).

CISA (2013) Black Box Security Testing Tools, The US-CERT website archive. Available at: https://www.cisa.gov/uscert/bsi/articles/tools/black-box-testing/black-box-security-testing-toolsg (Accessed: 27 October 2022).

Cisco (2013) The Internet of Everything Global Public Sector Economic Analysis. Available at: https://www.cisco.com/c/dam/en_us/about/business-insights/docs/ioe-value-at-stake-public-sector-analysis-faq.pdf (Accessed: 26 January 2020).

Cisco (2017) What Is a Cyberattack? - Most Common Types - Cisco. Available at: https://www.cisco.com/c/en/us/products/security/common-cyberattacks.html (Accessed: 21 February 2020).

Cisco (2020) What Is a Firewall? - Cisco. Available at: https://www.cisco.com/c/en/us/products/security/firewalls/what-is-a-firewall.html (Accessed: 13 June 2020).

Clarke, E.M., Wing, J.M., et al. (1996) ‘Formal methods: State of the art and future directions’, ACM Computing Surveys, 28(4), pp. 626–643. doi:10.1145/242223.242257.

Cohen, L., Manion, L. and Morrison, K. (2013) Research Methods in Education, Taylor & Francis. doi:10.4324/9781315456539.

Colak, I. (2016) ‘Introduction to smart grid’, 2016 International Smart Grid Workshop and Certificate Program, ISGWCP 2016, pp. 1–5. doi:10.1109/ISGWCP.2016.7548265.

Connor, P. and Fitch-Roy, O. (2019) The Socio-Economic Challenges of Smart Grids, Pathways to a Smarter Power System. Elsevier Ltd. doi:10.1016/B978-0-08-102592-5.00014-4.

Creswell, J.W. and Creswell, J.D. (2017) Research design: Qualitative, quantitative, and mixed methods approaches. Sage publications.

Dagle, J.E. (2012) ‘Cyber-physical system security of smart grids’, 2012 IEEE PES Innovative Smart Grid Technologies, ISGT 2012, pp. 1–2. doi:10.1109/ISGT.2012.6175607.

Dahlqvist, F., Patel, M., Rajko, A. and Shulmanm, J. (2019) Growing opportunities in the Internet of Things | McKinsey. Available at: https://www.mckinsey.com/industries/private-equity-and-principal-investors/our-insights/growing-opportunities-in-the-internet-of-things (Accessed: 16 February 2020).

Dalipi, F. and Yayilgan, S.Y. (2016) ‘Security and privacy considerations for IoT application on smart grids: Survey and research challenges’, Proceedings - 2016 4th International Conference on Future Internet of Things and Cloud Workshops, W-FiCloud 2016, pp. 63–68. doi:10.1109/W-FiCloud.2016.28.

Das, A.K. and Zeadally, S. (2019) Data Security in the Smart Grid Environment, Pathways to a Smarter Power System. Elsevier Ltd. doi:10.1016/B978-0-08-102592-5.00013-2.

DeBlasio, R. and Tom, C. (2008) ‘Standards for the smart grid’, 2008 IEEE Energy 2030 Conference, ENERGY 2008, (November), pp. 1–7. doi:10.1109/ENERGY.2008.4780988.

Díaz Redondo, R.P., Fernández-Vilas, A. and Dos Reis, G.F. (2020) ‘Security aspects in smart meters: Analysis and prevention’, Sensors (Switzerland), 20(14), pp. 1–19. doi:10.3390/s20143977.

Ding, J., Qammar, A., Zhang, Z., Karim, A. and Ning, H. (2022) ‘Cyber Threats to Smart Grids: Review, Taxonomy, Potential Solutions, and Future Directions’, Energies, 15(18), pp. 1–37. doi:10.3390/en15186799.

DLMS User Association (2019) ‘Green Book Edition 9 DLMS/COSEM Architecture and Protocols’, pp.

1–142.

Dolev, D. and Yao, A.C. (1983) ‘On the Security of Public Key Protocols’, IEEE Transactions on Information Theory, 29(2), pp. 198–208. doi:10.1109/TIT.1983.1056650.

Egozcue, E., Rodríguez, D.H., Ortiz, J.A., Villar, V.F. and Tarrafeta, L. (2012) Annex II. Smart Grid Security.

EPRI (2005) EPRI | SmartGrid Resource Center. Available at: https://smartgrid.epri.com/ (Accessed: 7 January 2020).

ETP (2006) Smart Grids European Technology Platform-EARPA. Available at: https://www.earpa.eu/earpa/39/etp_smartgrids.html.

Fadhel, N., Lombardi, F., Aniello, L., Margheri, A. and Sassone, V. (2019) ‘Towards a semantic modelling for threat analysis of IoT applications: A case study on transactive energy’, IET Conference Publications, 2019(CP756). doi:10.1049/cp.2019.0147.

Farha, F., Ning, H., yang, shunkun, xu, J., Zhang, W. and Choo, K.-K.R. (2020) ‘Timestamp Scheme to Mitigate Replay Attacks in Secure ZigBee Networks’, IEEE Transactions on Mobile Computing, 1233(c), pp. 1–1. doi:10.1109/tmc.2020.3006905.

Farrell, S. and Housley, R. (2002) ‘An Internet Attribute Certificate Profile for Authorization”, RFC 3281’. Available at: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.431.6587 (Accessed: 9 September 2021).

Fink, A. (2003) The survey handbook. Sage publications. doi:https://dx.doi.org/10.4135/9781412986328.n3.

Fitsilis, P., Gerogiannis, V., Anthopoulos, L. and Savvas, I. (2010) ‘Supporting the requirements prioritization process using social network analysis techniques’, Proceedings of the Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE, (October 2014), pp. 110–115. doi:10.1109/WETICE.2010.24.

Flick, T. and Morehouse, J. (2011) ‘Threats and Impacts: Consumers’, pp. 19–33. doi:10.1016/B978-1-59749-570-7.00002-9.

Ganguly, P., Nasipuri, M. and Dutta, S. (2018) ‘A Novel Approach for Detecting and Mitigating the Energy Theft Issues in the Smart Metering Infrastructure’, Technology and Economics of Smart Grids and Sustainable Energy, 3(1). doi:10.1007/s40866-018-0053-x.

Ganguly, P., Nasipuri, M. and Dutta, S. (2019) ‘Challenges of the Existing Security Measures Deployed in the Smart Grid Framework’, Proceedings of 2019 the 7th International Conference on Smart Energy Grid Engineering, SEGE 2019, pp. 1–5. doi:10.1109/SEGE.2019.8859917.

Gegick, M. and Barnum, S. (2005) Least Privilege | CISA, CISA. Available at: https://us-cert.cisa.gov/bsi/articles/knowledge/principles/least-privilege (Accessed: 18 September 2020).

Ghasempour, A. (2016) ‘Optimized Advanced Metering Infrastructure Architecture of Smart Grid based on Total Cost , Energy , and Delay’, 2016 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT), pp. 1–6. doi:10.1109/ISGT.2016.7781250.

Ghasempour, A. (2019) ‘Internet of things in smart grid: Architecture, applications, services, key technologies, and challenges’, Inventions, 4(1). doi:10.3390/inventions4010022.

Giani, A., Bitar, E., Garcia, M., McQueen, M., Khargonekar, P. and Poolla, K. (2013) ‘Smart grid data integrity attacks’, IEEE Transactions on Smart Grid, 4(3), pp. 1244–1253. doi:10.1109/TSG.2013.2245155.

Glenn, C., Sterbentz, D. and Wright, A. (2017) Cyber Threat and Vulnerability Analysis of the U.S. Electric Sector, Inl (Idaho National Laboratory). Available at: https://energy.gov/sites/prod/files/2017/01/f34/Cyber Threat and Vulnerability Analysis of the U.S. Electric Sector.pdf.

Golafshani, N. (2003) ‘Understanding Reliability and Validity in Qualitative Research’, The Qualitative

Report, 8(4), pp. 597–607. doi:10.17763/haer.62.3.8323320856251826.

GOV.UK (2024) Guidance Electric vehicles: costs, charging and infrastructure. Available at: https://www.gov.uk/government/publications/electric-vehicles-costs-charging-and-infrastructure/electric-vehicles-costs-charging-and-infrastructure (Accessed: 20 June 2024).

Grant, J.S. and Davis, L.L. (1997) 'Focus on Quantitative Methods: Selection and Use of Content Experts for Instrument Development', Research in Nursing and Health, 20(3), pp. 269–274. doi:10.1002/(sici)1098-240x(199706)20:3<269::aid-nur9>3.3.co;2-3.

Green, M. (2014) The Application of Black Box Theory to System Development, Incose. Available at: https://www.incose.org/products-and-publications/papers-presentations-library.

Guest, G., Bunce, A. and Johnson, L. (2006) 'How Many Interviews Are Enough?: An Experiment with Data Saturation and Variability', Field Methods, 18(1), pp. 59–82. doi:10.1177/1525822X05279903.

Guion, L.A. (2002) 'Triangulation: Establishing the Validity of Qualitative Studies. Institute of Food and Agricultural Sciences: University of Florida', pp. 1–3. Available at: http://edis.ifas.ufl.edu.

Gunduz, M.Z. and Das, R. (2020) 'Cyber-security on smart grid: Threats and potential solutions', Computer Networks, 169, p. 107094. doi:10.1016/j.comnet.2019.107094.

Gungor, V.C., Lu, B., Member, S., Hancke, G.P. and Member, S. (2010) 'Opportunities and Challenges of Wireless Sensor Networks in Smart Grid', 57(10), pp. 3557–3564.

Gutzmann, K. (2001) 'Access control and session management in the HTTP environment', IEEE Internet Computing, 5(1), pp. 26–35. doi:10.1109/4236.895139.

Hahn, A. and Govindarasu, M. (2011) 'Cyber attack exposure evaluation framework for the smart grid', IEEE Transactions on Smart Grid, 2(4), pp. 835–843. doi:10.1109/TSG.2011.2163829.

Hall, A. (1990) 'Seven Myths of Formal Methods', IEEE Software, 7(5), pp. 11–19. doi:10.1109/52.57887.

He, D., Kumar, N., Chen, J., Lee, C.C., Chilamkurti, N. and Yeo, S.S. (2013) 'Robust anonymous authentication protocol for health-care applications using wireless medical sensor networks', Multimedia Systems, 21(1), pp. 49–60. doi:10.1007/s00530-013-0346-9.

Hemsley, K.E. and Fisher, R.E. (2018) History of Industrial Control System Cyber Incidents, INL/CON-18-44411-Revision-2, Idaho National Lab, Idaho Falls. Available at: https://www.osti.gov/servlets/purl/1505628.

Hoang, T.S. (2013) 'An introduction to the Event-B modelling method', Industrial Deployment of System Engineering Methods, pp. 211–236. doi:10.1007/978-3-642-33170-1.

Hu, J. and Vasilakos, A. V. (2016) 'Energy Big Data Analytics and Security: Challenges and Opportunities', IEEE Transactions on Smart Grid, 7(5), pp. 2423–2436. doi:10.1109/TSG.2016.2563461.

Hussain, S., Meraj, M., Abughalwa, M. and Shikfa, A. (2018) 'Smart Grid Cybersecurity: Standards and Technical Countermeasures', 2018 International Conference on Computer and Applications, ICCA 2018, pp. 136–140. doi:10.1109/COMAPP.2018.8460390.

Hutchins, E., Cloppert, M. and Amin, R. (2011) 'Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains', 6th International Conference on Information Warfare and Security, ICIW 2011, (July 2005), pp. 113–125.

IEEE (2018) About - IEEE Smart Grid. Available at: https://smartgrid.ieee.org/about-ieee-smart-grid (Accessed: 4 December 2019).

IET (2013) What is a Smart Grid? Available at: https://www.theiet.org/media/1251/smart-grids.pdf (Accessed: 7 January 2020).

IoTSF (2020) Best Practice Guidelines 2 – IoT Security Foundation. Available at: https://www.iotsecurityfoundation.org/best-practice-guidelines/ (Accessed: 23 July 2020).

Irwin, S. (2019) ‘SANS Institute Information Security Reading Room Creating a Threat Profile for Your Organization’. Available at: https://www.sans.org/reading-room/whitepapers/threats/creating-threat-profile-organization-35492.

Jelacic, B., Rosic, D., Lendak, I., Stanojevic, M. and Stoja, S. (2018) ‘STRIDE to a secure smart grid in a hybrid cloud’, in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pp. 77–90. doi:10.1007/978-3-319-72817-9_6.

Jeon, Y.-H. (2011) ‘QoS Requirements for the Smart Grid Communications System’, IJCSNS International Journal of Computer Science and Network Security, 11(3), pp. 86–94. Available at: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=6038941.

Jung, M., Kienesberger, G., Granzer, W., Unger, M. and Kastner, W. (2011) ‘Privacy enabled web service access control using SAML and XACML for home automation gateways’, 2011 International Conference for Internet Technology and Secured Transactions, ICITST 2011, (December), pp. 584–591.

Kamto, J., Qian, L., Fuller, J. and Attia, J. (2011) ‘Light-weight key distribution and management for Advanced Metering Infrastructure’, 2011 IEEE GLOBECOM Workshops, GC Wkshps 2011, pp. 1216–1220. doi:10.1109/GLOCOMW.2011.6162375.

Kaur, M. and Kalra, S. (2016) ‘A Review on IOT Based Smart Grid’, International Journal of Energy, Information and Communications, 7(3), pp. 11–22. doi:10.14257/ijeic.2016.7.3.02.

Kessler, G.C. (2019) ‘An Overview of Cryptography (Updated Version’, 1998(January), pp. 1–65. Available at: https://www.garykessler.net/library/crypto.html.

Khan, R., Mclaughlin, K., Laverty, D. and Sezer, S. (2017) ‘STRIDE-based Threat Modeling for Cyber-Physical Systems’, pp. 0–5.

Al Khuffash, K. (2018) Smart grids—Overview and background information, Application of Smart Grid Technologies. Elsevier Inc. doi:10.1016/b978-0-12-803128-5.00001-5.

Kimani, K., Oduol, V. and Langat, K. (2019) ‘Cyber security challenges for IoT-based smart grid networks’, International Journal of Critical Infrastructure Protection, 25, pp. 36–49. doi:10.1016/j.ijcip.2019.01.001.

Knapp, E.D. and Samani, R. (2013) Applied cyber security and the smart grid: implementing security controls into the modern power infrastructure. Amsterdam: Elsevier, Syngress. Available at: https://rb.gy/0zhje6.

Komninos, N., Philippou, E. and Pitsillides, A. (2014) ‘Survey in smart grid and smart home security: Issues, challenges and countermeasures’, IEEE Communications Surveys and Tutorials, 16(4), pp. 1933–1954. doi:10.1109/COMST.2014.2320093.

KSA (2017) Vision 2030 Kingdom of Saudi Arabia. Available at: https://www.vision2030.gov.sa/media/rc0b5oy1/saudi_vision203.pdf.

Kumar Suman, S., Aqib, M. and Kumar Singh, S. (2017) ‘A Security Approach for Smart Grid on Review’, APTIKOM Journal on Computer Science and Information Technologies, 2(1), pp. 12–19. doi:10.11591/aptikom.j.csit.93.

Lázaro, J., Astarloa, A., Rodríguez, M., Bidarte, U. and Jiménez, J. (2021) ‘A survey on vulnerabilities and countermeasures in the communications of the smart grid’, Electronics (Switzerland), 10(16), pp. 1–15. doi:10.3390/electronics10161881.

Lee, E.-K., Oh, S.Y. and Gerla, M. (2011) ‘Frequency quorum rendezvous for fast and resilient key establishment under jamming attack’, ACM SIGMOBILE Mobile Computing and Communications Review, 14(4), pp. 1–3. doi:10.1145/1942268.1942270.

Ling, A.P.A. and Masao, M. (2011) ‘Selection of model in developing information security criteria on smart grid security system’, Proceedings - 9th IEEE International Symposium on Parallel and Distributed Processing with Applications Workshops, ISPAW 2011 - ICASE 2011, SGH 2011, GSDP 2011, pp. 91–98. doi:10.1109/ISPAW.2011.12.

Lozupone, V. (2018) ‘Analyze encryption and public key infrastructure (PKI)’, International Journal of Information Management, 38(1), pp. 42–44. doi:10.1016/j.ijinfomgt.2017.08.004.

Lynn, M.R. (1986) ‘Determination and Quantification Of Content Validity’, Nursing Research, 35(6), pp. 382–386. doi:https://doi.org/10.1097/00006199-198611000-00017.

Machaka, P., McDonald, A., Nelwamondo, F. and Bagula, A. (2016) ‘Using the cumulative sum algorithm against distributed denial of service attacks in internet of things’, Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST, 165, pp. 62–72. doi:10.1007/978-3-319-29236-6_7.

Mahmood, K., Ashraf Chaudhry, S., Naqvi, H., Shon, T. and Farooq Ahmad, H. (2016) ‘A lightweight message authentication scheme for Smart Grid communications in power sector’, Computers and Electrical Engineering, 52, pp. 114–124. doi:10.1016/j.compeleceng.2016.02.017.

Manyika, J., Chui, M., Bisson, P., Woetzel, J., Dobbs, R., Bughin, J. and Aharon, D. (2015) Unlocking the potential of the Internet of Things | McKinsey &amp; Company, McKinsey. Available at: http://www.mckinsey.com/business-functions/digital-mckinsey/our-insights/the-internet-of-things-the-value-of-digitizing-the-physical-world.

Marcelo, T., Gobbi, M., De Berardinis, E., Pannunzio, G. and Palumbo, C. (2013) Smart Metering and Smart Grids Strategy for the Kingdom of Saudi Arabia Strategy, Business Case, and Minimum Functional Requirements. Available at: www.cesi.it.

Marinos, L. (2013) ‘European Union Agency for Network and Information Security Smart Grid Threat Landscape and Good Practice Guide Smart Grid Threat Landscape and Good Practice Guide About ENISA Smart Grid Threat Landscape and Good Practice Guide’, (December).

Marinos, L. and Lourenço, M. (2018) ENISA Threat Landscape Report 2018 15 Top Cyberthreats and Trends, European Union Agency For Network and Information Security. doi:10.2824/622757.

Marshall, B., Cardon, P., Poddar, A. and Fontenot, R. (2013) ‘Does sample size matter in qualitative research?: A review of qualitative interviews in is research’, Journal of Computer Information Systems, 54(1), pp. 11–22. doi:10.1080/08874417.2013.11645667.

Martins, J.F., Pronto, A.G., Delgado-Gomes, V. and Sanduleac, M. (2019) ‘Smart Meters and Advanced Metering Infrastructure’, in Pathways to a Smarter Power System, pp. 89–114. doi:10.1016/b978-0-08-102592-5.00004-1.

Mavridou, A. and Papa, M. (2012) ‘A situational awareness architecture for the smart grid’, Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, 99 LNICST(July 2015), pp. 229–236. doi:10.1007/978-3-642-33448-1_31.

McCary, E. and Xiao, Y. (2015) ‘Smart Grid Attacks and Countermeasures’, EAI Endorsed Transactions on Industrial Networks and Intelligent Systems, 2(2), p. e4. doi:10.4108/inis.2.2.e4.

Mckinsey & Company (2024) Integrating renewable energy sources into grids, Mckinsey Global Institute, San Francisco. Available at: https://www.mckinsey.com/industries/electric-power-and-natural-gas/our-insights/how-grid-operators-can-integrate-the-coming-wave-of-renewable-energy (Accessed: 20 June 2024).

Microsoft (2009) The STRIDE Threat Model | Microsoft Docs, Microsoft Docs. Available at: https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20)?redirectedfrom=MSDN (Accessed: 12 May 2020).

Microsoft (2021) Driver Security Guidance. Available at: https://docs.microsoft.com/en-us/windows-hardware/drivers/driversecurity/threat-modeling-for-drivers#the-stride-approach-to-threat-categorization (Accessed: 13 June 2020).

Mohammadali, A., Haghighi, M.S., Tadayon, M.H. and Mohammadi-Nodooshan, A. (2018) ‘A Novel Identity-Based Key Establishment Method for Advanced Metering Infrastructure in Smart Grid’, IEEE Transactions on Smart Grid, 9(4), pp. 2834–2842. doi:10.1109/TSG.2016.2620939.

Mohassel, R.R., Fung, A., Mohammadi, F. and Raahemifar, K. (2014) ‘A survey on Advanced Metering

Infrastructure', International Journal of Electrical Power and Energy Systems, 63, pp. 473–484. doi:10.1016/j.ijepes.2014.06.025.

Mrabet, Z. El, Kaabouch, N., Ghazi, Hassan El and Ghazi, Hamid El (2018) 'Cyber-security in smart grid: Survey and challenges', Computers and Electrical Engineering, 67, pp. 469–482. doi:10.1016/j.compeleceng.2018.01.015.

Mugunthan, S.R. and Vijayakumar, D.T. (2019) 'REVIEW ON IOT BASED SMART GRID ARCHITECTURE IMPLEMENTATIONS', 01(01), pp. 12–20.

Mui, R. and Frankl, P. (2010) 'Preventing SQL Injection through Automatic Query Sanitization with ASSIST', Electronic Proceedings in Theoretical Computer Science, 35, pp. 27–38. doi:10.4204/eptcs.35.3.

Myagmar, S., Lee, A.J. and Yurcik, W. (2005) 'Threat Modeling as a Basis for Security Requirements', In StorageSS '05: Proceedings of the 2005 ACM workshop on Storage security and survivability, pp. 94–102. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.703.8462&rep=rep1&type=pdf (Accessed: 8 June 2020).

Nabeel, M., Kerr, S., Ding, X. and Bertino, E. (2012) 'Authentication and key management for Advanced Metering Infrastructures utilizing physically unclonable functions', 2012 IEEE 3rd International Conference on Smart Grid Communications, SmartGridComm 2012, pp. 324–329. doi:10.1109/SmartGridComm.2012.6486004.

NEOM (2024) NEOM: an accelerator of human progress. Available at: https://www.neom.com/en-us (Accessed: 18 June 2024).

NIST (2014) NIST Special Publication 1108R3 NIST Framework and Roadmap for Smart Grid Interoperability Standards, Release 3.0, NIST Special Publication. doi:10.6028/NIST.SP.1108r3.

NISTIR 7628 (2014) NISTIR 7628 Guidelines for Smart Grid Cyber Security, Revision 1, NIST. doi:10.6028/NIST.IR.7628r1.

Nozomi Networks (2020) OT / IoT Security Report. Available at: https://www.nozominetworks.com/downloads/US/Nozomi-Networks-OT-IoT-Security-Report-2020-1H.pdf.

Ogburn, M., Turner, C. and Dahal, P. (2013) 'Homomorphic encryption', Procedia Computer Science, 20, pp. 502–509. doi:10.1016/j.procs.2013.09.310.

Otuoze, A.O., Mustafa, M.W. and Larik, R.M. (2018) 'Smart grids security challenges: Classification by sources of threats', Journal of Electrical Systems and Information Technology, 5(3), pp. 468–483. doi:10.1016/j.jesit.2018.01.001.

Pallotti, E. and Mangiatordi, F. (2011) 'Smart grid cyber security requirements', 2011 10th International Conference on Environment and Electrical Engineering, EEEIC.EU 2011 - Conference Proceedings, pp. 1–4. doi:10.1109/EEEIC.2011.5874822.

Paverd, A., Martin, A. and Brown, I. (2015) 'Privacy-enhanced bi-directional communication in the Smart Grid using trusted computing', 2014 IEEE International Conference on Smart Grid Communications, SmartGridComm 2014, pp. 872–877. doi:10.1109/SmartGridComm.2014.7007758.

Penghou, L. (2018) 'Design and implementation of feeder automation', MATEC Web of Conferences, 175, pp. 1–4. doi:10.1051/matecconf/201817503021.

Pour, M.M., Anzalchi, A. and Sarwat, A. (2017) 'A review on cyber security issues and mitigation methods in smart grid systems', Conference Proceedings - IEEE SOUTHEASTCON, pp. 17–20. doi:10.1109/SECON.2017.7925278.

PTI (2011) Smart Grid 101 for Local Governments. Available at: https://www.smartgrid.gov/files/documents/Smart_Grid_101_for_Local_Governments_201112.pdf (Accessed: 23 June 2020).

Rahman, S. (2009) ‘Smart grid expectations: What will make it a reality’, IEEE Power and Energy Magazine, 7(5). doi:10.1109/MPE.2009.933415.

Rawat, D.B. and Bajracharya, C. (2015) ‘Cyber security for smart grid systems: Status, challenges and perspectives’, Conference Proceedings - IEEE SOUTHEASTCON, 2015-June(June). doi:10.1109/SECON.2015.7132891.

Reka, S.S. and Dragicevic, T. (2018) ‘Future e ff ectual role of energy delivery : A comprehensive review of Internet of Things and smart grid’, Renewable and Sustainable Energy Reviews, 91(April), pp. 90–108. doi:10.1016/j.rser.2018.03.089.

Reyzin, L. and Reyzin, N. (2002) ‘Better than BiBa: Short one-time signatures with fast signing and verifying’, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2384, pp. 144–153. doi:10.1007/3-540-45450-0_11.

Risteska Stojkoska, B.L. and Trivodaliev, K. V. (2017) ‘A review of Internet of Things for smart home: Challenges and solutions’, Journal of Cleaner Production, 140, pp. 1454–1464. doi:10.1016/j.jclepro.2016.10.006.

Roth, B.G. and Spafford, E.H. (2011) ‘Implicit buffer overflow protection using memory segregation’, Proceedings of the 2011 6th International Conference on Availability, Reliability and Security, ARES 2011, pp. 175–182. doi:10.1109/ARES.2011.32.

Sagiroglu, S., Terzi, R., Canbay, Y. and Colak, I. (2017) ‘Big data issues in smart grid systems’, 2016 IEEE International Conference on Renewable Energy Research and Applications, ICRERA 2016, 5, pp. 1007–1012. doi:10.1109/ICRERA.2016.7884486.

Sailio, M., Latvala, O. and Szanto, A. (2020) ‘applied sciences Cyber Threat Actors for the Factory of the Future’, Appl. Sci [Preprint].

Sajid, A., Abbas, H. and Saleem, K. (2016) ‘Cloud-Assisted IoT-Based SCADA Systems Security: A Review of the State of the Art and Future Challenges’, IEEE Access, 4, pp. 1375–1384. doi:10.1109/ACCESS.2016.2549047.

Saleem, Y., Crespi, N., Rehmani, M.H. and Copeland, R. (2019) ‘Internet of Things-Aided Smart Grid: Technologies, Architectures, Applications, Prototypes, and Future Research Directions’, IEEE Access, 7, pp. 62962–63003. doi:10.1109/ACCESS.2019.2913984.

Sani, A.S., Yuan, D., Jin, J., Gao, L., Yu, S. and Dong, Z.Y. (2019) ‘Cyber security framework for Internet of Things-based Energy Internet’, Future Generation Computer Systems, 93, pp. 849–859. doi:10.1016/j.future.2018.01.029.

Santos, L., Rabadao, C. and Goncalves, R. (2018) ‘Intrusion detection systems in Internet of Things: A literature review’, Iberian Conference on Information Systems and Technologies, CISTI, 2018-June, pp. 1–7. doi:10.23919/CISTI.2018.8399291.

SASG (2023) SASG– Saudi Arabia Smart Grid 2023. Available at: https://sasg2023.com/en/ (Accessed: 18 June 2024).

Saudi Electricity Company (2019) 40-SDMS-02A. Available at: https://www.se.com.sa/ar-sa/Business_Document/pdf/40-SDMS-02A- CT-CTVT Meter Specifications Rev.9_ Sep-19.pdf.

Scarfone, K. and Mell, P. (2011) Intrusion Detection and Prevention Systems, Securing the Information Infrastructure. doi:10.4018/978-1-59904-379-1.ch012.

Shakerighadi, B., Anvari-Moghaddam, A., Vasquez, J.C. and Guerrero, J.M. (2018) ‘Internet of things for modern energy systems: State-of-the-art, challenges, and open issues’, Energies, 11(5). doi:10.3390/en11051252.

Sharma, S., Singh, S. and Sharma, M. (2008) ‘Performance analysis of load balancing algorithms’, World Academy of Science, engineering and technology, pp. 269–272. Available at: http://masters.donntu.edu.ua/2010/fknt/babkin/library/article11.pdf.

Shirey, R. (2007) Internet Security Glossary, Version 2, IETF. doi:10.17487/RFC4949.

Shostack, A. (2008) ‘Experiences threat modeling at Microsoft’, CEUR Workshop Proceedings, 413, pp. 1–11.

Shostack, A. (2014) Threat Modeling. 1st Editio, Security for Software Engineers. 1st Editio. Wiley. Available at: https://www.perlego.com/book/1003144/threat-modeling-designing-for-security-pdf.

Silva, L.V., Marinho, R., Vivas, J.L. and Brito, A. (2017) ‘Security and privacy preserving data aggregation in cloud computing’, Proceedings of the ACM Symposium on Applied Computing, Part F1280, pp. 1732–1738. doi:10.1145/3019612.3019795.

Singer, J. (2009) Enabling Tomorrow’s Electricity System: Report of the Ontario Smart Grid Forum. Available at: http://www.ieso.ca/en/Learn/Ontario-Power-System/etno/-/media/files/ieso/document-library/smart_grid/Smart_Grid_Forum-Report.pdf.

Sortomme, E., Hindi, M.M., MacPherson, S.D.J. and Venkata, S.S. (2011) ‘Coordinated charging of plug-in hybrid electric vehicles to minimize distribution system losses’, IEEE Transactions on Smart Grid, 2(1), pp. 186–193. doi:10.1109/TSG.2010.2090913.

Spitzner, L. (SANS) (2019) Applying Security Awareness Cyber Kill Chain, SANS. Available at: https://www.sans.org/blog/applying-security-awareness-to-the-cyber-kill-chain/ (Accessed: 5 November 2020).

Stellios, I., Kotzanikolaou, P., Psarakis, M., Alcaraz, C. and Lopez, J. (2018) ‘A survey of iot-enabled cyberattacks: Assessing attack paths to critical infrastructures and services’, IEEE Communications Surveys and Tutorials, 20(4), pp. 3453–3495. doi:10.1109/COMST.2018.2855563.

Stouffer, K., Pillitteri, V., Lightman, S., Abrams, M. and Hahn, A. (2015) Guide to Industrial Control Systems (ICS) Security NIST Special Publication 800-82 Revision 2, NIST Special Publication 800-82 rev 2. doi:http://dx.doi.org/10.6028/NIST.SP.800-82r1.

Suh, G.E. and Devadas, S. (2007) ‘Physical unclonable functions for device authentication and secret key generation’, p. 9. doi:10.1145/1278480.1278484.

Suleiman, H., Alqassem, I., Diabat, A., Arnautovic, E. and Svetinovic, D. (2015) ‘Integrated smart grid systems security threat model’, Information Systems, 53, pp. 147–160. doi:10.1016/j.is.2014.12.002.

Swiderski, F. and Snyder, W. (2004) Threat Modeling. Microsoft Press.

Tazi, K., Abdi, F. and Abbou, M.F. (2016) ‘Review on cyber-physical security of the smart grid: Attacks and defense mechanisms’, Proceedings of 2015 IEEE International Renewable and Sustainable Energy Conference, IRSEC 2015, pp. 1–6. doi:10.1109/IRSEC.2015.7455127.

Teng, J., Gu, W. and Xuan, D. (2012) Defending against physical attacks in wireless sensor networks, Handbook on Securing Cyber-Physical Critical Infrastructure. Elsevier Inc. doi:10.1016/B978-0-12-415815-3.00010-8.

Thomas, S.A. (2000) SSL and TLS Securing the Web. Wiley Computer Publishing. Available at: https://www.academia.edu/download/32169332/SSL_and_TLS_Essentials_-_Securing_the_Web.pdf (Accessed: 19 November 2021).

Thurmond, V.A. (2001) ‘The point of triangulation’, Journal of Nursing Scholarship, 33(3), pp. 253–258. doi:10.1111/j.1547-5069.2001.00253.x.

Tonyali, S., Akkaya, K., Saputro, N., Uluagac, A.S. and Nojoumian, M. (2018) ‘Privacy-preserving protocols for secure and reliable data aggregation in IoT-enabled Smart Metering systems’, Future Generation Computer Systems, 78, pp. 547–557. doi:10.1016/j.future.2017.04.031.

Tonyali, S., Cakmak, O., Akkaya, K., Mahmoud, M.M.E.A. and Guvenc, I. (2016) ‘Secure Data Obfuscation Scheme to Enable Privacy-Preserving State Estimation in Smart Grid AMI Networks’, IEEE Internet of Things Journal, 3(5), pp. 709–719. doi:10.1109/JIOT.2015.2510504.

Tufail, S., Parvez, I., Batool, S. and Sarwat, A. (2021) ‘A survey on cybersecurity challenges, detection, and mitigation techniques for the smart grid’, Energies, 14(18), pp. 1–22. doi:10.3390/en14185894.

U.S. Department of energy (2012) ‘Communications requirements of smart grid technologies’, in Smart Meters and the Smart Grid: Privacy and Cybersecurity Considerations, pp. 105–144. Available at: https://www.energy.gov/sites/default/files/gcprod/documents/Smart_Grid_Communications_Requirements_Report_10-05-2010.pdf.

U.S. Department of energy (2023) Grid Modernization and the Smart Grid, US Department of Energy. Available at: https://www.energy.gov/oe/grid-modernization-and-smart-grid (Accessed: 19 June 2024).

U.S. Department of Energy (2018) Smart Grid System Report 2018: Report to Congress, Https://Www.Energy.Gov/Oe/Information-Center/Library/Reports-and-Other-Documents. Available at: https://www.energy.gov/sites/prod/files/2019/02/f59/Smart Grid System Report November 2018_1.pdf.

US Code (2016) TITLE 44, Public Printing and Documents, US Federal Statute.

US DoD Joint Publication (2013) JP 3-60 Joint Targeting. Available at: https://www.justsecurity.org/wp-content/uploads/2015/06/Joint_Chiefs-Joint_Targeting_20130131.pdf.

US Public Law (2007) Energy independence and security act of 2007, US Government Printing Office. Available at: https://www.gpo.gov/fdsys/pkg/PLAW-110publ140/html/PLAW-110publ140.htm.

Wang, Q., Tai, W., Tang, Y. and Ni, M. (2019) ‘Review of the false data injection attack against the cyber-physical power system’, IET Cyber-Physical Systems: Theory and Applications, 4(2), pp. 101–107. doi:10.1049/iet-cps.2018.5022.

Wang, W. and Lu, Z. (2013) ‘Cyber security in the Smart Grid: Survey and challenges’, Computer Networks, 57(5), pp. 1344–1371. doi:10.1016/j.comnet.2012.12.017.

Wang, X., Xu, Z., Cai, Z. and Wang, T. (2020) ‘Novel Temporal Perturbation-Based Privacy-Preserving Mechanism for Smart Meters’, Mobile Networks and Applications, 25(4), pp. 1548–1562. doi:10.1007/s11036-019-01359-8.

Wang, X. and Yi, P. (2011) ‘Smart Distribution Grid’, Ieee Transactions on Smart Grid, 2(4), pp. 809–818. doi:10.1109/TSG.2011.2167354.

Wei, D., Lu, Y., Jafari, M., Skare, P.M. and Rohde, K. (2011) ‘Protecting smart grid automation systems against cyberattacks’, IEEE Transactions on Smart Grid, 2(4), pp. 782–795. doi:10.1109/TSG.2011.2159999.

Wiegers, K. (2021) Software Development Pearls; Lessons from Fifty Years of Software Experience. Addison-Wesley Professional.

Wing, J.M. (1990) ‘A Specifier’s Introduction to Formal Methods’, Computer, 23(9), pp. 8–22. doi:10.1109/2.58215.

Wing, J.M. (1998) ‘A symbiotic relationship between formal methods and security’, Proceedings - Computer Security, Dependability, and Assurance: From Needs to Solutions, CSDA 1998, 1998-Novem(December), pp. 26–38. doi:10.1109/CSDA.1998.798355.

Xiao, Z., Xiao, Y. and Du, D. (2013) ‘Non-repudiation in neighborhood area networks for smart grid’, IEEE Communications Magazine, 51(1), pp. 18–26. doi:10.1109/MCOM.2013.6400434.

Xu, J., Wang, W., Pei, J., Wang, X., Shi, B. and Fu, A.W.-C. (2006) ‘Utility-based anonymization for privacy preservation with less information loss’, ACM SIGKDD Explorations Newsletter, 8(2), pp. 21–30. doi:10.1145/1233321.1233324.

Yan, J., He, H., Zhong, X. and Tang, Y. (2017) ‘Q-Learning-Based Vulnerability Analysis of Smart Grid Against Sequential Topology Attacks’, IEEE Transactions on Information Forensics and Security, 12(1), pp. 200–210. doi:10.1109/TIFS.2016.2607701.

Yan, Y., Qian, Y., Sharif, H. and Tipper, D. (2012) ‘A survey on cyber security for smart grid communications', IEEE Communications Surveys and Tutorials, 14(4), pp. 998–1010. doi:10.1109/SURV.2012.010912.00035.

Yang, Q. (2019) 'Internet of things application in smart grid: A brief overview of challenges, opportunities, and future trends', in Smart Power Distribution Systems. doi:10.1016/b978-0-12-812154-2.00013-4.

Yilmaz, Y. and Uludag, S. (2017) 'Mitigating IoT-based cyberattacks on the smart grid', Proceedings - 16th IEEE International Conference on Machine Learning and Applications, ICMLA 2017, 2017-Decem, pp. 517–522. doi:10.1109/ICMLA.2017.0-109.

Yu, S., Li, M. and Shi, L. (2014) Trust Establishment in Wireless Body Area Networks, Wearable Sensors: Fundamentals, Implementation and Applications. Elsevier Inc. doi:10.1016/B978-0-12-418662-0.00011-8.

Zeynal, H., Eidiani, M. and Yazdanpanah, D. (2014) 'Intelligent Substation Automation Systems for robust operation of smart grids', 2014 IEEE Innovative Smart Grid Technologies - Asia, ISGT ASIA 2014, (August), pp. 786–790. doi:10.1109/ISGT-Asia.2014.6873893.

Zhang, P. (2010) Industrial control system simulation routines. First Edit, Advanced Industrial Control Technology. First Edit. Peng Zhang. doi:10.1016/b978-1-4377-7807-6.10019-1.

Alotaibi, I., Abido, M. A., Khalid, M., & Savkin, A. V. (2020). A comprehensive review of recent advances in smart grids: A sustainable future with renewable energy resources. Energies, 13(23). https://doi.org/10.3390/en13236269

Arshad, Q. ul A., Khan, W. Z., Azam, F., Khan, M. K., Yu, H., & Zikria, Y. Bin. (2023). Blockchain-based decentralized trust management in IoT: systems, requirements and challenges. *Complex and Intelligent Systems*, *9*(6), 6155–6176. https://doi.org/10.1007/s40747-023-01058-8

Aurangzeb, M., Wang, Y., Iqbal, S., Naveed, A., Ahmed, Z., Alenezi, M., & Shouran, M. (2024). Enhancing cybersecurity in smart grids: Deep black box adversarial attacks and quantum voting ensemble models for blockchain privacy-preserving storage. *Energy Reports*, *11*, 2493–2515. https://doi.org/10.1016/j.egyr.2024.02.010

Bertot, Y., & Cast, P. (2013). *Interactive theorem proving and program development: Coq'Art: the calculus of inductive constructions*. Springer.

Butler, M. (2013). Mastering system analysis and design through abstraction and refinement. *Engineering Dependable Software Systems*, *34*, 49–78. https://doi.org/10.3233/978-1-61499-207-3-49

Cavalieri, S., Cantali, G., & Susinna, A. (2022). Integration of IoT Technologies into the Smart Grid. *Sensors*, *22*(7). https://doi.org/10.3390/s22072475

Dissanayake, N., Jayatilaka, A., Zahedi, M., & Babar, M. A. (2022). Software security patch management - A systematic literature review of challenges, approaches, tools and practices. In *Information and Software Technology* (Vol. 144). Elsevier B.V. https://doi.org/10.1016/j.infsof.2021.106771

Katoen, J.-P. (2010). *Advances in Probabilistic Model Checking* (pp. 25–25). https://doi.org/10.1007/978-3-642-11319-2_5

Khan, R., Mclaughlin, K., Laverty, D., & Sezer, S. (2017). *STRIDE-based Threat Modeling for Cyber-Physical Systems*. 0–5.

Kim, K. H., Kim, K., & Kim, H. K. (2022). STRIDE-based threat modeling and DREAD evaluation for the distributed control system in the oil refinery. *ETRI Journal*, *44*(6), 991–1003. https://doi.org/10.4218/etrij.2021-0181

Klinger, C., Landeg, O., & Murray, V. (2014). Power Outages, Extreme Events and Health: A Systematic Review of the Literature from 2011-2012. *PLoS Currents*, *JAN*. https://doi.org/10.1371/currents.dis.04eb1dc5e73dd1377e05a10e9edde673

Kumar, N. M., Chand, A. A., Malvoni, M., Prasad, K. A., Mamun, K. A., Islam, F. R., & Chopra, S. S. (2020). Distributed energy resources and the application of ai, iot, and blockchain in smart grids. In *Energies* (Vol. 13, Issue 21). MDPI AG. https://doi.org/10.3390/en13215739

Marcelo, T., Gobbi, M., De Berardinis, E., Pannunzio, G., & Palumbo, C. (2013). *Smart Metering and Smart Grids Strategy for the Kingdom of Saudi Arabia Strategy, Business Case, and Minimum Functional Requirements* (Issue June). www.cesi.it

Mell, P., Bergeron, T., & Henning, D. (2005). *Creating a Patch and Vulnerability Management Program Recommendations NIST*.

Myagmar, S., Lee, A. J., & Yurcik, W. (2005). Threat Modeling as a Basis for Security Requirements. *In StorageSS '05: Proceedings of the 2005 ACM Workshop on Storage Security and Survivability*, 94–102. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.703.8462&rep=rep1&type=pdf

NASA.gov. (2008). *System Failure Case Studies*. https://sma.nasa.gov/docs/default-source/safety-messages/safetymessage-2008-03-01-northeastblackoutof2003.pdf

Parker, D. (2023). *Multi-Agent Verification and Control with Probabilistic Model Checking*. http://arxiv.org/abs/2308.02829

Yaseen, A. (2023). *The Role of Machine Learning in Network Anomaly Detection for Cybersecurity*. https://orcid.org/0009-0002-8950-0767

Appendix A Expert Review Questionnaire

This part shows the expert review questionnaire with all the documents that were emailed to the experts.

Expert Review Questionnaire

Dear Participant,

Based on your experience in the field of Smart Grid, and for us to collect information about exchanging data securely between energy sectors, you have been selected to participate in this interview. Please refer to the Participant Information sheet first, then if you decide to take part in this research, please refer to the consent form. By signing the consent form, you agree that you are willing to answer the questions in this interview. Terms are defined in the glossary appended to this pack of interview questions. I appreciate your time and attention.

Yours sincerely,

Abeer Akkad

Introduction

An IoT-enabled Smart Grid involves many challenges and issues. It is argued that the main issue of the IoT-enabled Smart Grid is security. It is essential to develop a highly secure information system to prevent an attacker from modifying the consumption data. Compromising the security of smart meters will mislead estimations, and an incorrect consumption estimation would lead to large financial losses. Security is also important to protect the privacy of consumers and the utility. This has led to a question regarding an appropriate model for a secure information system for an IoT-enabled Smart Grid in the Saudi energy sector and the security requirements, threats, and controls to secure the information flow in the IoT-enabled Smart Grid in Saudi Arabia.

………………… Questions start …………………

Section A: Access points **

In the current research, the following are the main access points of the IoT-enabled Smart Grids, under each access point, there are several security threats from the STRIDE analysis undertaken in this research. The access points are highlighted by red arrows in the following Figure A-1:

A.1. ***Anything to add:*** Is there any missing access point?

A.2. ***Anything to change or remove:*** Would you change/remove something concerning the access points?

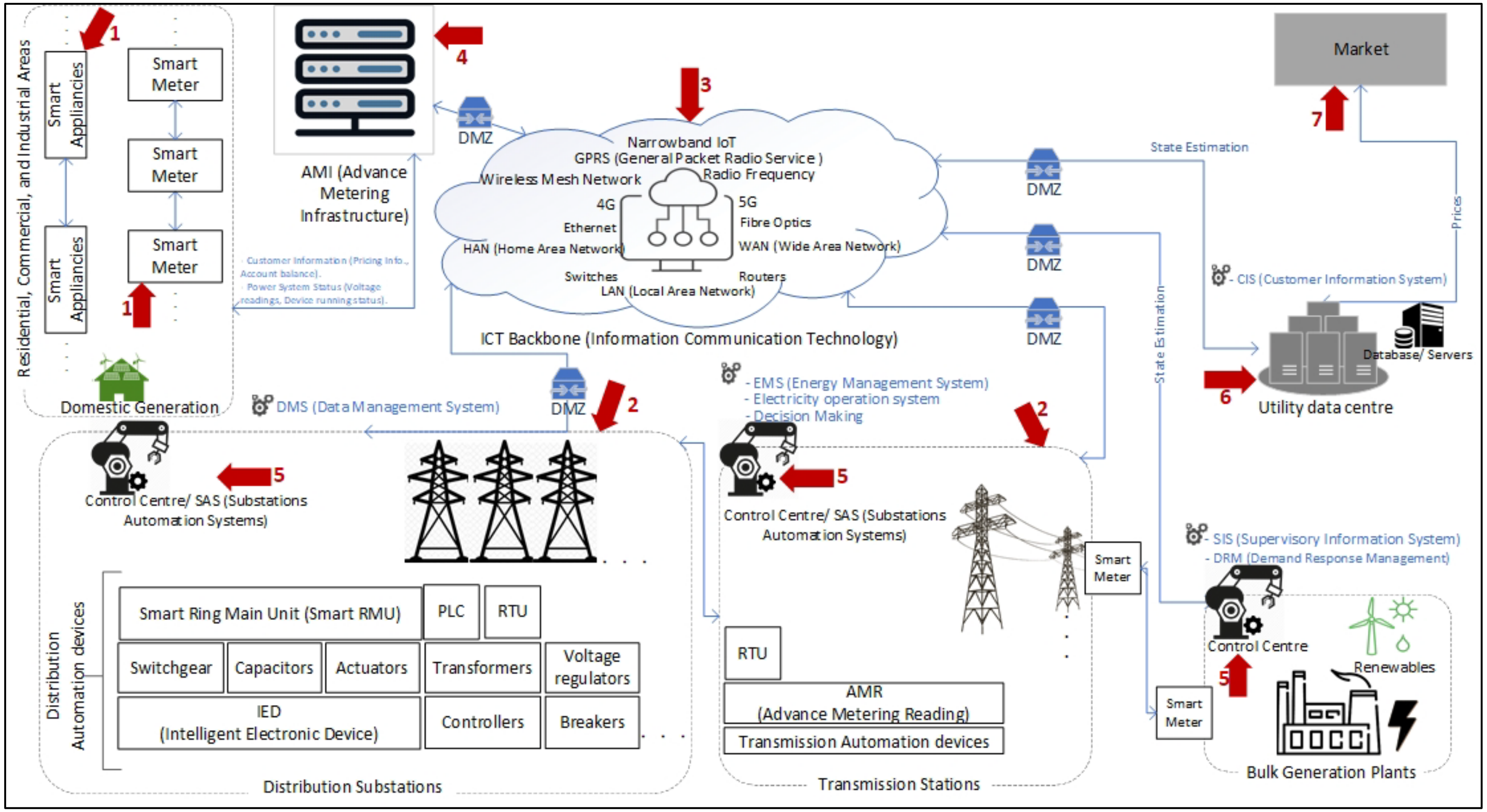

Residential, Commercial, and Industrial Areas
Smart Appliancies
Smart Appliancies
1
1
Smart Meter
Smart Meter
Smart Meter
Domestic Generation
4
AMI (Advance Metering Infrastructure)
- Customer Information (Pricing Info., Account balance).
- Power System Status (Voltage readings, Device running status).
DMZ
3
Narrowband IoT
GPRS (General Packet Radio Service)
Radio Frequency
Wireless Mesh Network
4G
5G
Ethernet
Fibre Optics
HAN (Home Area Network)
WAN (Wide Area Network)
Switches
Routers
LAN (Local Area Network)
ICT Backbone (Information Communication Technology)
DMZ
DMZ
DMZ
Market
7
State Estimation
Prices
CIS (Customer Information System)
Database/ Servers
6
Utility data centre
State Estimation
DMS (Data Management System)
DMZ
2
- EMS (Energy Management System)
- Electricity operation system
- Decision Making
2
5
Control Centre/ SAS (Substations Automation Systems)
5
Control Centre/ SAS (Substations Automation Systems)
Distribution Automation devices
Smart Ring Main Unit (Smart RMU)
PLC
RTU
Switchgear
Capacitors
Actuators
Transformers
Voltage regulators
IED (Intelligent Electronic Device)
Controllers
Breakers
Distribution Substations
RTU
AMR (Advance Metering Reading)
Transmission Automation devices
Transmission Stations
Smart Meter
SIS (Supervisory Information System)
- DRM (Demand Response Management)
Control Centre
5
Renewables
Smart Meter
Bulk Generation Plants

Figure A-1: The Access Points of the IoT-enabled Smart Grids

** 1-Smart Meters and Smart Appliances. 2-Distribution Substations, Transmission Stations, and Smart automation devices for transmission and distribution. 3- Information Communication Technology. 4-Advance Metering Infrastructure. 5- SCADA (Supervisory Control and Data Acquisition). 6-Utility data centre. 7-Market.

Section B: Controls and Requirements

The following master Table A-1 suggests the security controls that secure the information flow and serve the security requirement of the IoT-enabled Smart Grids:

B.1. ***Anything to add:*** Is there something that should be added to the security controls for the corresponding requirement? Why?

B.2. ***Anything to change or Remove:*** Would you change/remove something concerning the security controls? Why?

Table A-1: security requirements for IoT-enabled Smart Grids and corresponding security controls

Security requirement	**Security control**	**Code**
Authentication (**Aun**)	1. Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators	Aun1
	2. Physically Unclonable Functions (PUF)	Aun2
	3. MAC-attached, and HORS-signed messages	Aun3
	4. Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)	Aun4
	5. Multi-factor authentication mechanism	Aun5
	6. Automatic lockouts	Aun6
Authorisation (**Aur**)	7. Attribute-Based Encryption	Aur1
	8. Attribute Certificates	Aur2
	9. Attribute-Based Access Control System based on XACML (Extensible Access Control Markup Language)	Aur3
	10. Role-Based Access Control and allow/block listing	Aur4
Confidentiality (**C**)	11. Symmetric and asymmetric algorithms and Public Key Infrastructure certificate (PKI)	C1
Privacy (**P**)	12. Anonymisation	P1
	13. Trusted aggregators	P2
	14. Homomorphic encryption	P3
	15. Perturbation models	P4
	16. Verifiable computation models, and zero-knowledge proof systems	P5
	17. Data obfuscation techniques	P6
Integrity (**In**)	18. Cryptographic hashing functions and session keys	In1
	19. Digital watermarking	In2
	20. Automated patch management for flaw remediation	In3
	21. Adaptive cumulative sum algorithm	In4
	22. Secure Phasor Measurement Units (PMUs) installation	In5
	23. Load profiling algorithms	In6
	24. Timestamps	In7
	25. Sequence numbers	In8
	26. Query sanitisation	In9
	27. Nonces	In10
Availability (**Av**)	28. Use multiple alternate frequency channels according to a hardcoded sequence	Av1
	29. Frequency quorum rendezvous between connected nodes	Av2

Security requirement	Security control	Code
	30. Anomaly Intrusion Detection Systems (IDS)	Av3
	31. Specification-based IDS	Av4
	32. Intrusion Prevention Systems (IPS)	Av5
	33. Quality of Services (QoS)	Av6
	34. Load balancing	Av7
	35. Operating system-independent Applications	Av8
Non-repudiation (**N**)	36. Mutual Inspection technique	N1
	37. Unique keys and digital signatures	N2
	38. Transaction log	N3

Section C: Access Points and Threats

For each row of each table (Table A-2 to Table A-8),

C.1. ***Anything to add:*** Is there something that should be added to the threats for the corresponding access point? Why? (see column A,B).

C.2. ***Anything to change or Remove:*** Would you change/remove something concerning the threats? Why? (see column A,B).

C.3. ***Anything to add:*** Is there something that should be added to the controls for the corresponding access point? Why? (see column A,D).

C.4. ***Anything to change or Remove:*** Would you change/remove something concerning the controls? Why? (see column A,D).

Table A-2: Smart Meters and Smart Appliances matrix

A	B	C	D
Access Point	**Internet-based Security Threats from the STRIDE Analysis**	**Security Requirement**	**Security Control**
1. Smart Meters and Smart Appliances	Spoofing	Authentication Authorisation	Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
			Aun2: Physically Unclonable Functions (PUF)
			Aun3: MAC-attached, and HORS-signed messages
			Aun6: Automatic lockouts
			Aur2: Attribute Certificates
			Aur3: Attribute-Based Access Control System based on XACML
			Aur4: Role-Based Access Control and block listing
	Eavesdropping/ Traffic Analysis/ Man In The Middle (MITM)	Confidentiality	Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
			Aun2: Physically Unclonable Functions (PUF)
			Aun3: MAC-attached, and HORS-signed messages
			Aun4: Secure Sockets layer Certificates (SSL Certificates)
			Aun6: Automatic lockouts
			P1: Anonymisation
			P2: Trusted aggregators
			P5: Verifiable computation models, and zero-knowledge proof systems
	Replay Attack	Integrity	In1: Cryptographic hashing functions and Session keys
			In3: Automated patch management for flaw remediation
			In6: Load profiling algorithms
			In7: Timestamps
	Data Tampering	Integrity	In8: Sequence numbers
			In10: Nonces

A	B	C	D
Access Point	**Internet-based Security Threats from the STRIDE Analysis**	**Security Requirement**	**Security Control**
	Denial Of Service	Availability	Av1: Use multiple alternate frequency channels according to a hardcoded sequence
			Av3: Anomaly Intrusion Detection Systems (IDS)
			Av4: Specification-based IDS
	Malware injection	Availability Integrity	Av5: Intrusion Prevention Systems (IPS)
			Av6: Quality of Services (QoS)

Table A-3: Transmission Stations, Distribution Substations, Smart automation devices for transmission and distribution matrix

A	B	C	D
Access Point	**Internet-based Security Threats from the STRIDE analysis**	**Security Requirement**	**Security Control**
2. Transmission Stations, Distribution Substations, and Smart automation devices for transmission and distribution (Switches, Sensors, Actuators, Transformers, Voltage regulator, Capacitors).	Spoofing	Authentication Authorisation	Aun2: Physically Unclonable Functions (PUF)
			Aun4: Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)
			Aur2: Attribute Certificates
			Aur3: Attribute-Based Access Control System based on XACML
			Aur4: Role-Based Access Control and allow listing
	Eavesdropping/ Traffic Analysis/ Man In The Middle (MITM)	Confidentiality	Aun2: Physically Unclonable Functions (PUF)
			Aun4: Secure Sockets layer Certificates (SSL Certificates)
			P1: Anonymisation
			P2: Trusted aggregators
			P4: Perturbation models
			P6: Data obfuscation techniques
	Replay Attack	Integrity	In3: Automated patch management for flaw remediation
			In5: Secure Phasor Measurement Units (PMUs) installation
			In6: Load profiling algorithms
			In7: Timestamps
	Data Tampering	Integrity	In8: Sequence numbers
			In10: Nonces
	Denial Of Service	Availability	Av1: Use multiple alternate frequency channels according to a hardcoded sequence
			Av3: Anomaly Intrusion Detection Systems (IDS)
			Av4: Specification-based IDS
			Av5: Intrusion Prevention Systems (IPS)
	Malware injection	Availability Integrity	Av6: Quality of Services (QoS)

A	B	C	D
Access Point	Internet-based Security Threats from the STRIDE analysis	Security Requirement	Security Control
	False data injection	Integrity Non-Repudiation	N1: Mutual Inspection technique
			N3: transaction log

Table A-4: Generation Plant and Information Communication Technology (ICT) Systems matrix

A	B	C	D
Access point	**Internet-based Security Threats from the STRIDE analysis**	**Security Requirement**	**Security Control**
3. Generation Plant and Information Communication Technology (ICT) Systems	Spoofing	Authentication Authorisation	Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
			Aun2: Physically Unclonable Functions (PUF)
			Aun3: MAC-attached, and HORS-signed messages
			Aun4: Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)
			Aun5: Multi-factor authentication mechanism
			Aun6: Automatic lockouts
			Aur1: Attribute-Based Encryption
			Aur2: Attribute Certificates
			Aur3: Attribute-Based Access Control System based on XACML
			Aur4: Role-Based Access Control and allow listing
	Eavesdropping/ Traffic Analysis/ Man In The Middle (MITM)	Confidentiality	C1: Symmetric and asymmetric algorithms and Public Key Infrastructure certificate (PKI)
			Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
			Aun2: Physically Unclonable Functions (PUF)
			Aun3: MAC-attached, and HORS-signed messages
			Aun4: Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)
			Aun5: Multi-factor authentication mechanism
			Aun6: Automatic lockouts
			P1: Anonymisation
			P2: Trusted aggregators

A	B	C	D
Access point	**Internet-based Security Threats from the STRIDE analysis**	**Security Requirement**	**Security Control**
			P3: Homomorphic encryption
			P4: Perturbation models
			P5: Verifiable computation models, and zero-knowledge proof systems
			P6: Data obfuscation techniques
	Replay Attack	Integrity	In1: Cryptographic hashing functions and Session keys
	Data Tampering	Integrity	In3: Automated patch management for flaw remediation
			In4: Adaptive cumulative sum algorithm
			In5: Secure Phasor Measurement Units (PMUs) installation
			In6: Load profiling algorithms
			In7: Timestamps
			In8: Sequence numbers
			In10: Nonces
	Denial Of Service	Availability	Av1: Use multiple alternate frequency channels according to a hardcoded sequence
			Av2: Frequency quorum rendezvous between connected nodes
			Av3: Anomaly Intrusion Detection Systems (IDS)
			Av4: Specification-based IDS
			Av5: Intrusion Prevention Systems (IPS)
			Av6: Quality of Services (QoS)
			Av7: Load balancing
	Malware injection	Availability Integrity	Av8: Operating system-independent Applications
	False data injection	Integrity Non-Repudiation	N1: Mutual Inspection technique
			N2: Unique keys and digital signatures
			N3: Transaction log

Table A-5: Advanced Metering Infrastructure (AMI) matrix

A	B	C	D
Access point	**Internet-based Security Threats from the STRIDE analysis**	**Security Requirement**	**Security Control**
4. Advanced Metering Infrastructure (AMI)	Spoofing	Authentication Authorisation	Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
			Aun2: Physically Unclonable Functions (PUF)
			Aun3: MAC-attached, and HORS-signed messages
			Aun4: Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)
			Aun6: Automatic lockouts
			Aur1: Attribute-Based Encryption
			Aur3: Attribute-Based Access Control System based on XACML
			Aur4: Role-Based Access Control and allow listing
	Eavesdropping/ Traffic Analysis/ Man In The Middle (MITM)	Confidentiality	C1: Symmetric and asymmetric algorithms and Public Key Infrastructure certificate (PKI)
			Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
			Aun2: Physically Unclonable Functions (PUF)
			Aun3: MAC-attached, and HORS-signed messages
			Aun4: Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)
			Aun6: Automatic lockouts
			P1: Anonymisation
			P2: Trusted aggregators
			P3: Homomorphic encryption
			P4: Perturbation models
			P5: Verifiable computation models, and zero-knowledge proof systems
			P6: Data obfuscation techniques

A	B	C	D
Access point	**Internet-based Security Threats from the STRIDE analysis**	**Security Requirement**	**Security Control**
	Replay Attack	Integrity	In1: Cryptographic hashing functions and Session keys
	Data Tampering	Integrity	In3: Automated patch management for flaw remediation
			In4: Adaptive cumulative sum algorithm
			In7: Timestamps
			In8: Sequence numbers
			In9: Query sanitisation
	SQL injection	Integrity	In10: Nonces
	Denial Of Service	Availability	Av1: Use multiple alternate frequency channels according to a hardcoded sequence
	Malware injection	Availability Integrity	Av2: Frequency quorum rendezvous between connected nodes
			Av3: Anomaly Intrusion Detection Systems (IDS)
			Av4: Specification-based IDS
			Av5: Intrusion Prevention Systems (IPS)
			Av6: Quality of Services (QoS)
			Av7: Load balancing
			Av8: Operating system-independent Applications
	False data injection	Integrity Non-Repudiation	N1: Mutual Inspection technique
			N2: Unique keys and digital signatures
			N3: Transaction log

Table A-6: SCADA Systems matrix

A	B	C	D
Access point	**Internet-based Security Threats from the STRIDE analysis**	**Security Requirement**	**Security Control**
5. SCADA/ SAS control centre	Spoofing	Authentication Authorisation	Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
			Aun2: Physically Unclonable Functions (PUF)
			Aun3: MAC-attached, and HORS-signed messages
			Aun6: Automatic lockouts
			Aur1: Attribute-Based Encryption
			Aur3: Attribute-Based Access Control System based on XACML
	Eavesdropping/ Traffic Analysis/ Man In The Middle (MITM)	Confidentiality	Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
			Aun2: Physically Unclonable Functions (PUF)
			Aun3: MAC-attached, and HORS-signed messages
			Aun6: Automatic lockouts
			P2: Trusted aggregators
			P3:Homomorphic encryption
	Replay Attack	Integrity	In3: Automated patch management for flaw remediation
	Data Tampering	Integrity	In6: Load profiling algorithms
			In7: Timestamps
			In8: Sequence numbers
			In10: Nonces
			In12: Comparing with baseline configuration
	Denial Of Service	Availability	Av2: Frequency quorum rendezvous between connected nodes
	Malware injection	Availability	Av3: Anomaly Intrusion Detection Systems (IDS)
			Av5: Intrusion Prevention Systems (IPS)

A	B	C	D
Access point	**Internet-based Security Threats from the STRIDE analysis**	**Security Requirement**	**Security Control**
		Integrity	Av6: Quality of Services (QoS)
	False data injection	Integrity Non-Repudiation	N3: Transaction log

Table A-7: Utility data centre matrix

A	B	C	D
Access point	**Internet-based Security Threats from the STRIDE analysis**	**Security Requirement**	**Security Control**
6. Utility data centre	Spoofing	Authentication Authorisation	Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
			Aun3: MAC-attached, and HORS-signed messages
			Aun4: Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)
			Aun5: Multi-factor authentication mechanism
			Aun6: Automatic lockouts
			Aur1: Attribute-Based Encryption
			Aur2: Attribute Certificates
			Aur3: Attribute-Based Access Control System based on XACML (Extensible Access Control Markup Language)
			Aur4: Role-Based Access Control and allow listing
	Eavesdropping/ Traffic Analysis/ Man In The Middle (MITM)	Confidentiality	C1: Symmetric and asymmetric algorithms and Public Key Infrastructure certificate (PKI)
			Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
			Aun3: MAC-attached, and HORS-signed messages
			Aun4: Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)
			Aun5: Multi-factor authentication mechanism
			Aun6: Automatic lockouts
			P1: Anonymisation
			P2: Trusted aggregators
			P3: Homomorphic encryption
			P4: Perturbation models

A	B	C	D
Access point	**Internet-based Security Threats from the STRIDE analysis**	**Security Requirement**	**Security Control**
			P5: Verifiable computation models, and zero-knowledge proof systems
	Phishing	Confidentiality	P6: Data obfuscation techniques
	Replay Attack	Integrity	In1: Cryptographic hashing functions and Session keys
	Data Tampering	Integrity	In2: Digital watermarking
			In3: Automated patch management for flaw remediation
			In4: Adaptive cumulative sum algorithm
			In6: Load profiling algorithms
			In7: Timestamps
			In8: Sequence numbers
			In9: Query sanitisation
			In10: Nonces
	SQL injection	Integrity	
	Denial Of Service	Availability	Av3: Anomaly Intrusion Detection Systems (IDS)
	Malware injection	Availability Integrity	Av4: Specification-based IDS
			Av5: Intrusion Prevention Systems (IPS)
			Av6: Quality of Services (QoS)
			Av7: Load balancing
			Av8: Operating system-independent Applications
	False data injection	Integrity Non-Repudiation	N1: Mutual Inspection technique
			N2: Unique keys and digital signatures
			N3: Transaction log

Table A-8: Market matrix

A	B	C	D
Access point	**Internet-based Security Threats from the STRIDE analysis**	**Security Requirement**	**Security Control**
7. Market	Spoofing	Authentication Authorisation	Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
			Aun3: MAC-attached, and HORS-signed messages
			Aun4: Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)
			Aun5: Multi-factor authentication mechanism
			Aun6: Automatic lockouts
			Aur1: Attribute-Based Encryption
			Aur2: Attribute Certificates
			Aur3: Attribute-Based Access Control System based on XACML (Extensible Access Control Markup Language)
			Aur4: Role-Based Access Control and allow listing
	Eavesdropping/ Traffic Analysis/ Man In The Middle (MITM)	Confidentiality	C1: Symmetric and asymmetric algorithms and Public Key Infrastructure certificate (PKI)
			Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
			Aun3: MAC-attached, and HORS-signed messages
			Aun4: Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)
			Aun5: Multi-factor authentication mechanism
			Aun6: Automatic lockouts
			P1: Anonymisation
			P2: Trusted aggregators
			P3: Homomorphic encryption
			P4: Perturbation models

A	B	C	D
Access point	**Internet-based Security Threats from the STRIDE analysis**	**Security Requirement**	**Security Control**
			P5: Verifiable computation models, and zero-knowledge proof systems
	Phishing	Confidentiality	P6: Data obfuscation techniques
	Replay Attack	Integrity	In1: Cryptographic hashing functions and Session keys
	Data Tampering	Integrity	In2: Digital watermarking
			In3: Automated patch management for flaw remediation
			In4: Adaptive cumulative sum algorithm
			In6: Load profiling algorithms
	SQL injection	Integrity	In7: Timestamps
			In8: Sequence numbers
			In9: Query sanitisation
			In10: Nonces
	Denial Of Service	Availability	Av3: Anomaly Intrusion Detection Systems (IDS)
	Malware injection	Availability Integrity	Av4: Specification-based IDS
			Av5: Intrusion Prevention Systems (IPS)
			Av6: Quality of Services (QoS)
			Av7: Load balancing
			Av8: Operating system-independent Applications
	False data injection	Integrity Non-Repudiation	N1: Mutual Inspection technique
			N2: Unique keys and digital signatures
			N3: Transaction log

………………… **End of questions** …………………

Appendix B The Modified Access Points Matrices

This part shows the modified matrices of each access point according to findings analysed from the expert review.

Table B-1: Smart meters and smart appliances matrix

A	B	Common controls	C	D
Access Point	**Internet-Based Security Threats from the STRIDE Analysis**		**Security Requirement**	**Security Control**
1. Smart Meters and Smart Appliances	Spoofing	Common1: Patch management for flaw remediation	Authentication Authorisation	Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
				Aun2: Physically Unclonable Functions (PUF)
				Aun3: MAC-attached, and HORS-signed messages
				Aun6: Automatic lockouts
				Aun7: Secure Session Management
				Aun8: Anti-Spoofing algorithm
				Aur2: Attribute Certificates
				Aur3: Attribute-Based Access Control System based on XACML
				Aur4: Role-Based Access Control and block listing
				Aur7: Principle of Least Privilege (POLP)

A	B	Common controls	C	D
Access Point	**Internet-Based Security Threats from the STRIDE Analysis**		**Security Requirement**	**Security Control**
	Eavesdropping/ Traffic Analysis/ Man In The Middle (MITM)		Confidentiality	Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
				Aun2: Physically Unclonable Functions (PUF)
				Aun3: MAC-attached, and HORS-signed messages
				Aun7: Secure Session Management
				Aun8: Anti-Spoofing algorithm
				Aun6: Automatic lockouts
				P1: Anonymisation
				P2: Trusted aggregators
	Replay Attack		Integrity	In1: Cryptographic hashing functions and Session keys
				In5: Load profiling algorithms
				In6: Timestamps
				In7: Sequence numbers
	Data Tampering		Integrity	In9: Nonces
	Denial Of Service		Availability	Av1: Use multiple alternate frequency channels according to a hardcoded sequence
				Av2: Anomaly Intrusion Detection Systems (IDS)
				Av3: Specification-based IDS
				Av4: Intrusion Prevention Systems (IPS)

A	B	Common controls	C	D
Access Point	**Internet-Based Security Threats from the STRIDE Analysis**		**Security Requirement**	**Security Control**
				Av5: Quality of Services (QoS)
				Av9: Web Application Firewall (WAF)
	Malware injection		Availability	Av10: Anti-DDOS algorithm
			Integrity	Av11: Segregation, Segmentation, Data Diode isolation, DMZ, and Air gap
				Common3: Endpoint for Detection and Response EDR

Table B-2: Transmission stations, distribution substations, smart automation devices for transmission and distribution matrix

A	B	Common Controls	C	D
Access Point	Internet-Based Security Threats from the STRIDE Analysis		Security Requirement	Security Control
2. Transmission Stations, Distribution Substations, and Smart automation devices for transmission and distribution (Switches, Sensors, Actuators, Transformers, Voltage	Spoofing	Common1: Patch management for flaw remediation	Authentication	Aun2: Physically Unclonable Functions (PUF)
			Authorisation	Aun4: Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)
				Aun8: Anti-Spoofing algorithm
				Aur2: Attribute Certificates
				Aur3: Attribute-Based Access Control System based on XACML
				Aur4: Role-Based Access Control
				Aur5: allow listing
				Aur6: Secure Session Management
				Aur7: Anti-Spoofing algorithm
	Eavesdropping/ Traffic Analysis/ Man In The Middle (MITM)		Confidentiality	Aun2: Physically Unclonable Functions (PUF)
				Aun4: Secure Sockets layer Certificates (SSL Certificates)
				P1: Anonymisation
				P2: Trusted aggregators
	Replay Attack		Integrity	In4: Secure Phasor Measurement Units (PMUs) installation
				In5: Load profiling algorithms
				In6: Timestamps

A	B	Common Controls	C	D
Access Point	**Internet-Based Security Threats from the STRIDE Analysis**		**Security Requirement**	**Security Control**
regulator, Capacitors)				In7: Sequence numbers
	Data Tampering		Integrity	In9: Nonces
	Denial Of Service		Availability	Av1: Use multiple alternate frequency channels according to a hardcoded sequence
				Av2: Anomaly Intrusion Detection Systems (IDS)
				Av3: Specification-based IDS
				Av4: Intrusion Prevention Systems (IPS)
				Av5: Quality of Services (QoS)
				Av8: Redundancy
				Av9: Web Application Firewall (WAF)
				Av10: Anti-DDOS algorithm
	Malware injection		Availability Integrity	Av11: Segregation, Segmentation, Data Diode isolation, DMZ, and Air gap
				Common3: Endpoint for Detection and Response EDR
	False data injection		Integrity Non-Repudiation	N1: Mutual Inspection technique
				N3: transaction log

Table B-3: Generation Plant and Information Communication Technology (ICT) Systems matrix

A	B		C	D
Access Point	**Internet-Based Security Threats from the STRIDE Analysis**	**Common Controls**	**Security Requirement**	**Security Control**
3. Generation Plant and Information Communication Technology (ICT) Systems	Spoofing	Common1: Patch management for flaw remediation	Authentication Authorisation	Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
				Aun2: Physically Unclonable Functions (PUF)
				Aun3: MAC-attached, and HORS-signed messages
				Aun4: Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)
				Aun5: Multi-factor authentication mechanism
				Aun6: Automatic lockouts
				Aun7: Secure Session Management
				Aun8: Anti-Spoofing algorithm
				Aur1: Attribute-Based Encryption
				Aur2: Attribute Certificates
				Aur3: Attribute-Based Access Control System based on XACML
				Aur4: Role-Based Access Control
				Aur5: allow listing
				Aur6: Privileged Access Management (PAM)
				Aur7: Principle of Least Privilege (POLP)

A	B		C	D
Access Point	**Internet-Based Security Threats from the STRIDE Analysis**	**Common Controls**	**Security Requirement**	**Security Control**
	Eavesdropping/ Traffic Analysis/ Man In The Middle (MITM)		Confidentiality	C1: Symmetric and asymmetric algorithms and Public Key Infrastructure certificate (PKI)
				Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
				Aun2: Physically Unclonable Functions (PUF)
				Aun3: MAC-attached, and HORS-signed messages
				Aun4: Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)
				Aun5: Multi-factor authentication mechanism
				Aun6: Automatic lockouts
				Aun7: Secure Session Management
				P1: Anonymisation
				P2: Trusted aggregators
				P3: Encryption
	Replay Attack		Integrity	In1: Cryptographic hashing functions and session keys
	Data Tampering		Integrity	In3: Adaptive cumulative sum algorithm
				In4: Secure Phasor Measurement Units (PMUs) installation
				In5: Load profiling algorithms
				In6: Timestamps

A	B		C	D
Access Point	**Internet-Based Security Threats from the STRIDE Analysis**	**Common Controls**	**Security Requirement**	**Security Control**
				In7: Sequence numbers
				In9: Nonces
	Denial Of Service		Availability	Av1: Use multiple alternate frequency channels according to a hardcoded sequence
				Av2: Anomaly Intrusion Detection Systems (IDS)
				Av3: Specification-based IDS
				Av4: Intrusion Prevention Systems (IPS)
				Av5: Quality of Services (QoS)
				Av6: Load balancing
				Av7: Operating system-independent Applications
				Av8: Redundancy
				Av9: Web Application Firewall (WAF)
				Av10: Anti-DDOS algorithm
	Malware injection		Availability	Av11: Segregation, Segmentation, Data Diode isolation, DMZ, and Air gap
			Integrity	Common3: Endpoint for Detection and Response EDR
	False data injection		Integrity	N1: Mutual Inspection technique
			Non-Repudiation	N2: Unique keys and digital signatures
				N3: Transaction log

Table B-4: Advanced Metering Infrastructure (AMI) matrix

A	B		C	D
Access Point	**Internet-Based Security Threats from the STRIDE Analysis**	**Common Controls**	**Security Requirement**	**Security Control**
4. Advanced Metering Infra-structure (AMI)	Spoofing	Common1: Patch management for flaw remediation	Authentication Authorisation	Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
				Aun2: Physically Unclonable Functions (PUF)
				Aun3: MAC-attached, and HORS-signed messages
				Aun4: Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)
				Aun6: Automatic lockouts
				Aun8: Anti-Spoofing algorithm
				Aur1: Attribute-Based Encryption
				Aur3: Attribute-Based Access Control System based on XACML
				Aur4: Role-Based Access Control
				Aur5: allow listing
				Aur6: Privileged Access Management (PAM)
				Aur7: Principle of Least Privilege (POLP)
	Eavesdropping/ Traffic Analysis/		Confidentiality	C1: Symmetric and asymmetric algorithms and Public Key Infrastructure certificate (PKI)

A	B		C	D
Access Point	**Internet-Based Security Threats from the STRIDE Analysis**	**Common Controls**	**Security Requirement**	**Security Control**
	Man In The Middle (MITM)			Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
				Aun2: Physically Unclonable Functions (PUF)
				Aun3: MAC-attached, and HORS-signed messages
				Aun4: Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)
				Aun6: Automatic lockouts
				P1: Anonymisation
				P2: Trusted aggregators
				P3: Encryption
	Replay Attack		Integrity	In1: Cryptographic hashing functions and session keys
	Data Tampering		Integrity	In3: Adaptive cumulative sum algorithm
				In6: Timestamps
				In7: Sequence numbers
				In8: Query sanitisation
				In9: Nonces
	SQL injection		Integrity	

A	B		C	D
Access Point	**Internet-Based Security Threats from the STRIDE Analysis**	**Common Controls**	**Security Requirement**	**Security Control**
	Denial Of Service		Availability	Av1: Use multiple alternate frequency channels according to a hardcoded sequence
	Malware injection		Availability Integrity	Av2: Anomaly Intrusion Detection Systems (IDS)
				Av3: Specification-based IDS
				Av4: Intrusion Prevention Systems (IPS)
				Av5: Quality of Services (QoS)
				Av6: Load balancing
				Av7: Operating system-independent Applications
				Av8: Redundancy
				Av10: Anti-DDOS algorithm
				Av11: Segregation, Segmentation, Data Diode isolation, DMZ, and Air gap
				Common3: Endpoint for Detection and Response EDR
	False data injection		Integrity Non-Repudiation	N1: Mutual Inspection technique
				N2: Unique keys and digital signatures
				N3: Transaction log

Table B-5: SCADA Systems matrix

A	B		C	D
Access Point	**Internet-Based Security Threats from the STRIDE Analysis**	**Common Controls**	**Security Requirement**	**Security Control**
5. SCADA/ SAS control centre	Spoofing	Common1: Patch management for flaw remediation	Authentication Authorisation	Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
				Aun2: Physically Unclonable Functions (PUF)
				Aun3: MAC-attached, and HORS-signed messages
				Aun6: Automatic lockouts
				Aun8: Anti-Spoofing algorithm
				Aur1: Attribute-Based Encryption
				Aur3: Attribute-Based Access Control System based on XACML
				Aur6: Privileged Access Management (PAM)
				Aur7: Principle of Least Privilege (POLP)
	Eavesdropping/ Traffic Analysis/ Man In The Middle (MITM)		Confidentiality	Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
				Aun2: Physically Unclonable Functions (PUF)
				Aun3: MAC-attached, and HORS-signed messages
				Aun6: Automatic lockouts
				P2: Trusted aggregators
				P3: Encryption

A	B		C	D
Access Point	**Internet-Based Security Threats from the STRIDE Analysis**	**Common Controls**	**Security Requirement**	**Security Control**
	Replay Attack		Integrity	In5: Load profiling algorithms
	Data Tampering		Integrity	In6: Timestamps
				In7: Sequence numbers
				In9: Nonces
	Denial Of Service		Availability	Av2: Anomaly Intrusion Detection Systems (IDS)
	Malware injection		Availability Integrity	Av4: Intrusion Prevention Systems (IPS)
				Av5: Quality of Services (QoS)
				Av8: Redundancy
				Av10: Anti-DDOS algorithm
				Av11: Segregation, Segmentation, Data Diode isolation, DMZ, and Air gap
				Common3: Endpoint for Detection and Response EDR
	False data injection		Integrity Non-Repudiation	N3: Transaction log

Table B-6: Utility data centre matrix

A	B		C	D
Access Point	**Internet-Based Security Threats from the STRIDE Analysis**	**Common Controls**	**Security Requirement**	**Security Control**
6. Utility data centre	Spoofing	Common1: Patch management for flaw remediation	Authentication Authorisation	Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
				Aun3: MAC-attached, and HORS-signed messages
				Aun4: Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)
				Aun5: Multi-factor authentication mechanism
				Aun6: Automatic lockouts
				Aun7: Secure Session Management
				Aun8: Anti-Spoofing algorithm
				Aur1: Attribute-Based Encryption
				Aur2: Attribute Certificates
				Aur3: Attribute-Based Access Control System based on XACML (Extensible Access Control Markup Language)
				Aur4: Role-Based Access Control
				Aur5: Allow listing
				Aur6: Privileged Access Management (PAM)
				Aur7: Principle of Least Privilege (POLP)

A	B		C	D
Access Point	**Internet-Based Security Threats from the STRIDE Analysis**	**Common Controls**	**Security Requirement**	**Security Control**
	Eavesdropping/ Traffic Analysis/ Man In The Middle (MITM)		Confidentiality	C1: Symmetric and asymmetric algorithms and Public Key Infrastructure certificate (PKI)
				Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
				Aun3: MAC-attached, and HORS-signed messages
				Aun4: Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)
				Aun5: Multi-factor authentication mechanism
				Aun6: Automatic lockouts
				Aun7: Secure Session Management
				P1: Anonymisation
				P2: Trusted aggregators
				P3: Encryption
	Phishing		Confidentiality	
	Replay Attack		Integrity	In1: Cryptographic hashing functions and Session keys
	Data Tampering		Integrity	In2: Digital watermarking
				In3: Adaptive cumulative sum algorithm
				In5: Load profiling algorithms

A	B		C	D
Access Point	**Internet-Based Security Threats from the STRIDE Analysis**	**Common Controls**	**Security Requirement**	**Security Control**
				In6: Timestamps
				In7: Sequence numbers
				In8: Query sanitisation
				In9: Nonces
	SQL injection		Integrity	
	Denial of Service		Availability	Av2: Anomaly Intrusion Detection Systems (IDS)
	Malware injection		Availability	Av3: Specification-based IDS
			Integrity	Av4: Intrusion Prevention Systems (IPS)
				Av5: Quality of Services (QoS)
				Av6: Load balancing
				Av7: Operating system-independent Applications
				Av8: Redundancy
				Av9: Web Application Firewall (WAF)
				Av10: Anti-DDOS algorithm
				Av11: Segregation, Segmentation, Data Diode isolation, DMZ, and Air gap
				Common3: Endpoint for Detection and Response EDR
	False data injection		Integrity	N1: Mutual Inspection technique
				N2: Unique keys and digital signatures

A	B		C	D
Access Point	**Internet-Based Security Threats from the STRIDE Analysis**	**Common Controls**	**Security Requirement**	**Security Control**
			Non-Repudiation	N3: Transaction log

Table B-7: Market matrix

A	B		C	D
Access Point	**Internet-Based Security Threats from the STRIDE Analysis**	**Common Controls**	**Security Requirement**	**Security Control**
7. Market	Spoofing	Common1: Patch management for flaw remediation	Authentication Authorisation	Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
				Aun3: MAC-attached, and HORS-signed messages
				Aun4: Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)
				Aun5: Multi-factor authentication mechanism
				Aun6: Automatic lockouts
				Aun7: Secure Session Management
				Aun8: Anti-Spoofing algorithm
				Aur1: Attribute-Based Encryption
				Aur2: Attribute Certificates
				Aur3: Attribute-Based Access Control System based on XACML (Extensible Access Control Markup Language)
				Aur4: Role-Based Access Control
				Aur5: allow listing
				Aur6: Privileged Access Management (PAM)
				Aur7: Principle of Least Privilege (POLP)

A	B		C	D
Access Point	**Internet-Based Security Threats from the STRIDE Analysis**	**Common Controls**	**Security Requirement**	**Security Control**
	Eavesdropping/ Traffic Analysis/ Man In The Middle (MITM)		Confidentiality	C1: Symmetric and asymmetric algorithms and Public Key Infrastructure certificate (PKI)
				Aun1: Keyed cryptographic hash functions (HMAC), digital signatures, and Random number generators
				Aun3: MAC-attached, and HORS-signed messages
				Aun4: Secure Sockets layer Certificates (SSL Certificates) and Transport Layer Security (TLS)
				Aun5: Multi-factor authentication mechanism
				Aun6: Automatic lockouts
				Aun7: Secure Session Management
				P1: Anonymisation
				P2: Trusted aggregators
				P3: Encryption
	Phishing		Confidentiality	
	Replay Attack		Integrity	In1: Cryptographic hashing functions and session keys
	Data Tampering		Integrity	In2: Digital watermarking
				In3: Adaptive cumulative sum algorithm
				In5: Load profiling algorithms

A	B		C	D
Access Point	**Internet-Based Security Threats from the STRIDE Analysis**	**Common Controls**	**Security Requirement**	**Security Control**
				In6: Timestamps
	SQL injection		Integrity	In7: Sequence numbers
				In8: Query sanitisation
				In9: Nonces
	Denial of Service		Availability	Av2: Anomaly Intrusion Detection Systems (IDS)
	Malware injection		Availability	Av3: Specification-based IDS
			Integrity	Av4: Intrusion Prevention Systems (IPS)
				Av5: Quality of Services (QoS)
				Av6: Load balancing
				Av7: Operating system-independent Applications
				Av8: Redundancy
				Av9: Web Application Firewall (WAF)
				Av10: Anti-DDOS algorithm
				Av11: Segregation, Segmentation, Data Diode isolation, DMZ, and Air gap
				Common3: Endpoint for Detection and Response EDR
	False data injection		Integrity	N1: Mutual Inspection technique
			Non-Repudiation	N2: Unique keys and digital signatures
				N3: Transaction log

Appendix C An Overview of Formal Methods and Event-B

Overview of Formal Methods: Definition, Features and Limitations

Definition

Formal methods are the mathematical modelling used to represent the specifications of a system using a structured approach. This type of representation can better describe the requirements and verify the systems (Hoang, 2013). The mathematical representation for formal methods is symbolised by a formal language that contains the notations, model behaviours, and precise rules to describe the behaviours of the model.

Features

First, reducing the cost of fixing software errors at an early phase of the development lifecycle. For instance, fixing software errors in the implementation and testing stages can be very costly than fixing the errors in the earlier design stage (Boehm and Papaccio, 1988). Moreover, formal methods express the informal requirements into formal specifications, which allows a better understanding of the system and reduces software errors caused by misinterpreted requirements (Wing, 1990).

Second, using formal reasoning to establish the model properties via Proof Obligations (POs) (Abrial and Hallerstede, 2007). This can help in revealing design errors instead of fixing those errors at the later testing phase. Thus, these design models can be refined during the design phase to avoid any significant changes during the implementation phase.

Limitations

The following points show the limitations of formal methodologies (Hall, 1990):

- **Representation of system properties:** Some system properties cannot be modelled. For instance, the non-functional requirements such as security and performance are challenging to be modelled.
- **Representation of a real-world environment:** Representing the real-world environment in a formal systematic model is challenging due to the formal language capabilities and tools.
- **Proving a complicated property:** Some properties are generated by combining multiple properties, which cannot be represented in formal language or are impossible to prove.
- **Proving is not always correct:** Although formal modelling tools and automated theorem provers may reduce the mistakes present, the mistakes still occur during modelling and proving.

Formal Specification Approaches

Formal Specification approaches are categorised into the System-based approach and the Logic-based approach and the Finite-state automata (FSA). FSA is an approach used to specify system behaviours as a state transition system. It includes the formalisms for discrete system events capturing time and probabilistic transition. The Logic-based approach formalises the behaviours and the critical properties of the system. Whereas, the System-based approach formalises the operations and states of the target system. Many formal languages, such as Z, B-Method and Event-B, are examples of this category.

Overview of Event-B Modelling Language Notations and Syntax

Event-B is a formal method commonly used for system development by introducing the system specifications gradually into the formal model through refinement techniques. A formal model in Event-B includes two parts: ***contexts*** and ***machines*** (Hoang, 2013). **Contexts** represent the static parts of a model and provide axiomatic characteristics. An Event-B context includes the following clauses:

1. **Carrier sets:** these are abstract types and cannot be non-empty.
2. **Constants:** these are logical variables whose values are not changing.
3. **Axioms:** these are logical predicates for constraining the properties of carrier sets and constants.
4. **Theorems:** these are properties that can be proven based on axioms or other theorems.

Machines represent the dynamic parts of a model. An Event-B machine includes the following clauses:

1. **Variables**: these describe the system states. Variables are defined based on mathematical formulas, such as sets, relations and functions.
2. **Invariants**: these describe the properties of a system. They also constrain variables and must always be true. For example, if I(v) represents an invariant for the variable (V), this invariant I(v) will hold 'True' for any change to the value of (V).
3. **Refines:** these are used to introduce more details to a concrete machine (e.g. M0 indicates the abstract machine, and M1 refers to a concrete machine, which is noted as 'M1 refines M0').
4. **Events:** these are atomic transitions that change the states of the system and represent the dynamic behaviour of a system. The transition state of an event is constrained by the guards and actions. A guard indicates an enabled condition of an event, while an action represents a transition state of a variable. An event may have parameters that make the guard hold a state and describe how the variables of a machine change simultaneously to preserve the atomic nature of the event. For instance, for an event **e** with parameters **t**, the guard of the event can be written as **G (t, v)**, and the action of the event can be represented as **S (t, v)**, as shown in form 3.1.

$$\mathbf{e} \cong \textit{any}\ \mathbf{t}\ \textit{where}\ \mathbf{G(t,v)}\ \textit{then}\ \mathbf{S(t,v)}\ \textit{end} \tag{3.1}$$

An event may not include parameters but only guards and actions to directly change variables. The following form 3.2 shows how a non-parameter event can be written.

$$\mathbf{e} \cong \textit{where}\ \mathbf{G(t,v)}\ \ \textit{then}\ \mathbf{S(t,v)}\ \textit{end} \tag{3.2}$$

An event may not include parameters and guards to initialise the machine variables, as shown in form 3.3.

$$\mathbf{e} \sim= \textit{begin than}\ \mathbf{S(t,v)}\ \textit{end} \tag{3.3}$$

The actions of events may involve different assignments and can be expressed as follows:

$$v := \textit{Expression}\ (\mathbf{t}, \mathbf{v}) \tag{3.4}$$

$$v :\in \textit{Expression}\ (\mathbf{t}, \mathbf{v}) \tag{3.5}$$

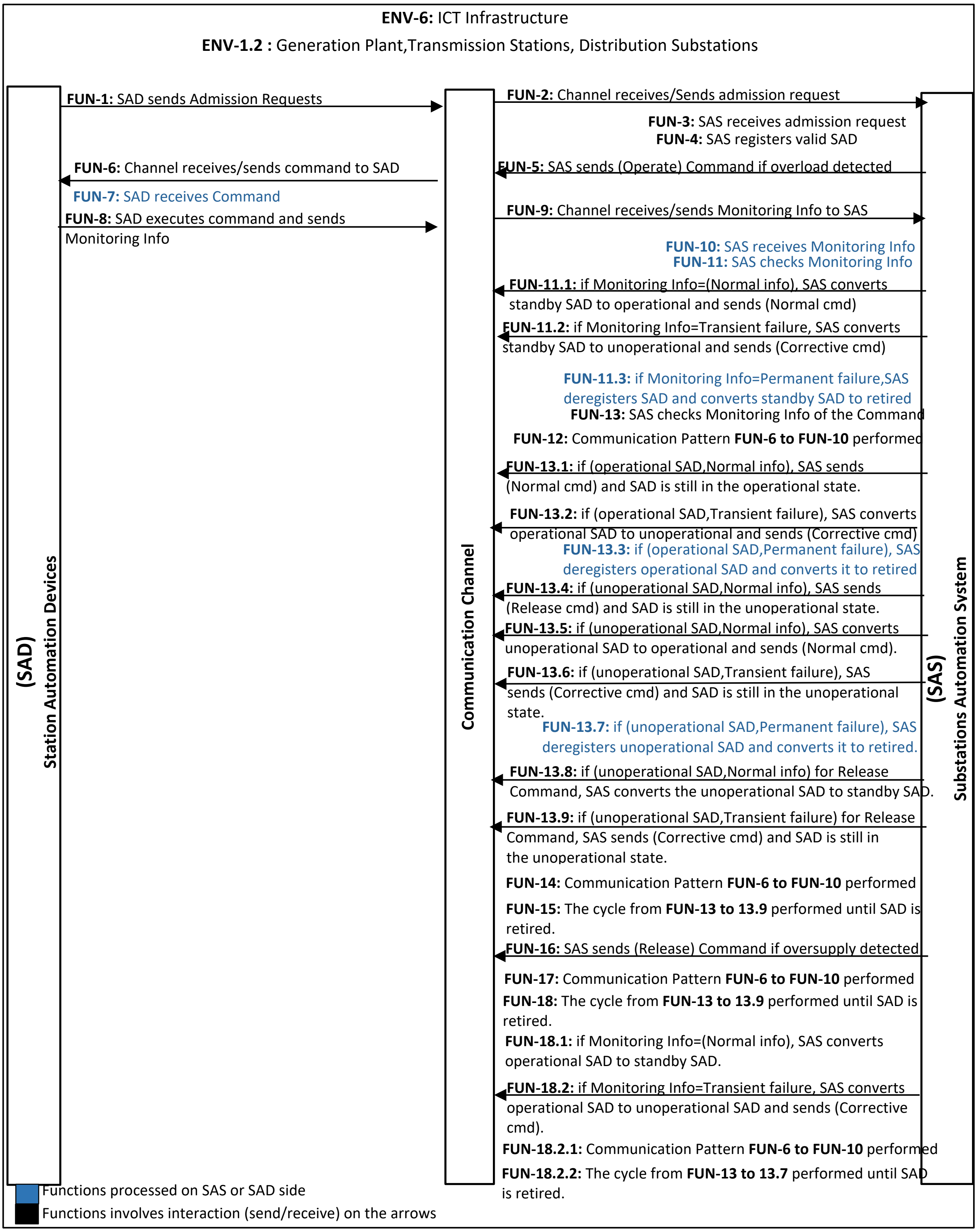
$$v : \|\ \textit{Predicate}\ (\mathbf{t}, \mathbf{v}) \tag{3.6}$$

The assignments can be deterministic (as shown in form 3.4), where an assignment specifies the value of the expression (t, v) to v. The assignments can also be non-deterministic (as shown in forms 3.5 and 3.6). Specifically, form 3.5 assigns any value from the set of the expression (t, v) to v, while form 3.6 assigns any value fulfilled in the predicate (t, v) to v.

Moreover, as invariants I(v) are inductive, they must be maintained by all events. This strictly ensures that all events obey the invariants when any event attempts to modify the state of variables. For this reason, Event-B machines use theorem proving to verify the consistency of events. Moreover, the Event-B machines apply model-checking techniques that aim to explore all reachable states of the system while transforming the invariants as safety properties of a system.

Appendix D Event-B full Codes, Full Interaction Diagram, and AMI-SM Case Study

Full Interaction Diagram for Detailed Information Flow Between SAS and SAD



ENV-6: ICT Infrastructure

ENV-1.2 : Generation Plant,Transmission Stations, Distribution Substations

FUN-1: SAD sends Admission Requests

FUN-2: Channel receives/Sends admission request

FUN-3: SAS receives admission request

FUN-4: SAS registers valid SAD

FUN-5: SAS sends (Operate) Command if overload detected

FUN-6: Channel receives/sends command to SAD

FUN-7: SAD receives Command

FUN-8: SAD executes command and sends Monitoring Info

FUN-9: Channel receives/sends Monitoring Info to SAS

FUN-10: SAS receives Monitoring Info

FUN-11: SAS checks Monitoring Info

FUN-11.1: if Monitoring Info=(Normal info), SAS converts standby SAD to operational and sends (Normal cmd)

FUN-11.2: if Monitoring Info=Transient failure, SAS converts standby SAD to unoperational and sends (Corrective cmd)

FUN-11.3: if Monitoring Info=Permanent failure,SAS deregisters SAD and converts standby SAD to retired

FUN-13: SAS checks Monitoring Info of the Command

FUN-12: Communication Pattern **FUN-6 to FUN-10** performed

FUN-13.1: if (operational SAD,Normal info), SAS sends (Normal cmd) and SAD is still in the operational state.

FUN-13.2: if (operational SAD,Transient failure), SAS converts operational SAD to unoperational and sends (Corrective cmd)

FUN-13.3: if (operational SAD,Permanent failure), SAS deregisters operational SAD and converts it to retired

FUN-13.4: if (unoperational SAD,Normal info), SAS sends (Release cmd) and SAD is still in the unoperational state.

FUN-13.5: if (unoperational SAD,Normal info), SAS converts unoperational SAD to operational and sends (Normal cmd).

FUN-13.6: if (unoperational SAD,Transient failure), SAS sends (Corrective cmd) and SAD is still in the unoperational state.

FUN-13.7: if (unoperational SAD,Permanent failure), SAS deregisters unoperational SAD and converts it to retired.

FUN-13.8: if (unoperational SAD,Normal info) for Release Command, SAS converts the unoperational SAD to standby SAD.

FUN-13.9: if (unoperational SAD,Transient failure) for Release Command, SAS sends (Corrective cmd) and SAD is still in the unoperational state.

FUN-14: Communication Pattern **FUN-6 to FUN-10** performed

FUN-15: The cycle from **FUN-13 to 13.9** performed until SAD is retired.

FUN-16: SAS sends (Release) Command if oversupply detected

FUN-17: Communication Pattern **FUN-6 to FUN-10** performed

FUN-18: The cycle from **FUN-13 to 13.9** performed until SAD is retired.

FUN-18.1: if Monitoring Info=(Normal info), SAS converts operational SAD to standby SAD.

FUN-18.2: if Monitoring Info=Transient failure, SAS converts operational SAD to unoperational SAD and sends (Corrective cmd).

FUN-18.2.1: Communication Pattern **FUN-6 to FUN-10** performed

FUN-18.2.2: The cycle from **FUN-13 to 13.7** performed until SAD is retired.

🟦 Functions processed on SAS or SAD side

⬛ Functions involves interaction (send/receive) on the arrows

Figure D-0-1: System block Diagram for the full interaction between SAS and SAD access points, showing the information flow

Justifications of Modelling Monitoring Information and Commands

This research does not aim to be explicit about the detailed types of commands and monitoring information, or how they are constructed. This is for the following reasons:

- For abstraction and simplification purposes for a such complex system in the IoT-enabled Smart Grid for better understanding of the main aims/concepts of the system, which are Monitoring and Controlling (Command) that the information transmission plays the main role in achieving these aims.
- It would affect the generalisability of the model and can be unduly restrictive if the research focused on specific internal commands and monitoring information of particular equipment/access point.
- The detailed types of Monitoring information or detailed types of Commands are similar in terms of security perspective as this research is concerned with the effect of threatened Monitoring information, which is an incorrect input/decision, and the effect of threatened Command, which is a fault or conflicting control action.
- The focus of this research is the security of the information flow as being transmitted around the IoT-enabled Smart Grid against internet-based threats. For the adversary, it is a black box of data packets regardless of the operational identity/type of the information itself, and the goal of this formal model is to prevent these threats.
- In this security verification, the black box modelling approach is a popular method for gaining insight into the overall (input-output) process behaviour. The black box model is the functional relationships between system inputs and system outputs without revealing the internal processing. This modelling approach is effective whenever the aim is merely to represent faithfully some trends in process behaviour (Zhang, 2010). The concept of black boxes has been initiated since the early days of systems theory and the first use was in the field of electrical engineering (Green, 2014). According to the Cybersecurity and Infrastructure Security Agency (CISA), black box testing tools are significant in web application security, such as input checking and validation, SQL injection attacks, False Data injection, Session management issues, and Buffer Overflow attacks. These are the threats within the scope of this research, being frequent internet-based threats to the IoT-enabled Smart Grid (CISA, 2013).

Full Event B Codes for the Formal Model

- **The Abstract Context C0**

```
context C0
sets
    AccessPoint
    MONITORING_INFO
    COMMAND
    FAULT_PASSAGE_INDICATOR
    MNTR_TYPE
    CMD_TYPE
constants
     SAD
    SAS
    OTHER_AccessPoint
    Overload
    Oversupply
    cmdid
    cmd_cntnt
    mntrid
    mntr_cntnt
    Normal_info
    Permanent_failure
    Transient_failure
    OTHER_MNTR_TYPE
    Normal_Cmd
    Operate_Cmd
    Release_Cmd
    Corrective_Cmd
axioms
    @axm1: SAS ⊆ AccessPoint
    @axm2: SAD ⊆ AccessPoint
    @axm3: OTHER_AccessPoint ⊆ AccessPoint
    @axm4: partition (AccessPoint, SAS, SAD, OTHER_AccessPoint)
    @axm5: Normal_info ∈ MNTR_TYPE
    @axm6: Permanent_failure ⊆ MNTR_TYPE
    @axm7: Transient_failure ⊆ MNTR_TYPE
    @axm8: OTHER_MNTR_TYPE ⊆ MNTR_TYPE
    @Valid_Mntr: partition (MNTR_TYPE, {Normal_info}, Permanent_failure, Transient_failure, OTHER_MNTR_TYPE)
    @axm10: Overload ∈ FAULT_PASSAGE_INDICATOR
    @axm11: Oversupply ∈ FAULT_PASSAGE_INDICATOR
    @axm12: partition (FAULT_PASSAGE_INDICATOR, {Overload}, {Oversupply})
    @axm13: Normal_Cmd ∈ CMD_TYPE
    @axm14: Operate_Cmd ∈ CMD_TYPE
    @axm15: Release_Cmd ∈ CMD_TYPE
    @axm16: Corrective_Cmd ⊆ CMD_TYPE
    @Valid_Command: partition (CMD_TYPE, {Normal_Cmd}, {Operate_Cmd}, {Release_Cmd}, Corrective_Cmd)
    @axm18: cmdid ∈ COMMAND → SAD
    @axm19: cmd_cntnt ∈ COMMAND → CMD_TYPE
    @axm20: mntrid ∈ MONITORING_INFO → SAD
    @axm21: mntr_cntnt ∈ MONITORING_INFO → MNTR_TYPE
end
```

- **The Abstract Machine M0**

```
machine M0 sees C0
variables
    valid_SAD
    registered_SAD
    unregistered_SAD
    retired_SAD
    standby_SAD
    channel_AdmRequest
     SAS_rcv_admRequest
    unoperational_SAD
    operational_SAD
    Operate_executed_SAD
    AllCmd_executed_SAD
```

```
   Release_executed_SAD
   fault
   channel_command
   SAD_received_command
   channel_mntrInfo
   SAS_received_mntrInfo
   previous_cmd
invariants
   @inv1: partition (SAD, valid_SAD, retired_SAD)
   @Legitimate_SAD: registered_SAD ⊆ valid_SAD
   @inv3: unregistered_SAD ⊆ valid_SAD
   @inv4: partition (valid_SAD, registered_SAD, unregistered_SAD)
   @inv5: operational_SAD ⊆ registered_SAD
   @inv6: unoperational_SAD ⊆ registered_SAD
   @inv7: standby_SAD ⊆ registered_SAD
   @inv9: partition (registered_SAD, operational_SAD, unoperational_SAD, standby_SAD)
   @inv10: channel_AdmRequest ⊆ unregistered_SAD
   @inv12: SAS_rcv_admRequest ⊆ unregistered_SAD
   @inv14: channel_mntrInfo ⊆ MONITORING_INFO
   @inv16: channel_command ⊆ COMMAND
   @inv18: Operate_executed_SAD ⊆ registered_SAD
   @inv19: AllCmd_executed_SAD ⊆ registered_SAD
   @inv22: Release_executed_SAD ⊆ registered_SAD
   @inv25: fault ⊆ FAULT_PASSAGE_INDICATOR
   @inv26: SAD_received_command ⊆ COMMAND
   @inv29: SAS_received_mntrInfo ⊆ MONITORING_INFO
   @inv30: previous_cmd ⊆ COMMAND
events
   event INITIALISATION
   then
      @act1: registered_SAD ≔ ∅
      @act2: retired_SAD ≔ ∅
      @act3: valid_SAD ≔ SAD
      @act4: unregistered_SAD ≔ SAD
      @act5: channel_AdmRequest ≔ ∅
      @act7: SAS_rcv_admRequest ≔ ∅
      @act9: unoperational_SAD ≔ ∅
      @act10: operational_SAD ≔ ∅
      @act11: standby_SAD ≔ ∅
      @act15: Operate_executed_SAD ≔ ∅
      @act18: AllCmd_executed_SAD ≔ ∅
      @act19: Release_executed_SAD ≔ ∅
      @act22: fault ≔ ∅
       @act23: channel_command ≔ COMMAND
      @act24: channel_mntrInfo ≔ MONITORING_INFO
      @act26: SAD_received_command ≔ ∅
      @act30: SAS_received_mntrInfo ≔ ∅
      @act31: previous_cmd ≔ ∅
   end

   event SAD_Send_Admission_Request
   any
      sad
   where
      @grd1: sad ∈ unregistered_SAD
      @grd3: sad ∉ channel_AdmRequest
      @grd5: sad ∉ SAS_rcv_admRequest
   then
      @act1: channel_AdmRequest ≔ channel_AdmRequest ∪ {sad}
   end

   event SAS_Receive_Admission_Request
   any
      sad
   where
      @grd1: sad ∈ unregistered_SAD
      @grd3: sad ∈ channel_AdmRequest
   then
      @act1: SAS_rcv_admRequest ≔ SAS_rcv_admRequest ∪ {sad}
      @act4: channel_AdmRequest ≔ channel_AdmRequest \ {sad}
   end

   event SAS_RegSAD_Success
```

```
any
   sad
where
   @grd1: sad ∈ unregistered_SAD
   @grd2: sad ∈ SAS_rcv_admRequest
   @grd3: sad ∉ channel_AdmRequest
then
   @act1: registered_SAD ≔ registered_SAD ∪ {sad}
   @act2: standby_SAD ≔ standby_SAD ∪ {sad}
   @act3: SAS_rcv_admRequest ≔ SAS_rcv_admRequest ∖ {sad}
   @act4: unregistered_SAD ≔ unregistered_SAD ∖ {sad}
end

event SAS_RegSAD_Fail
any
   sad
where
   @no-double-registered: sad ∈ registered_SAD ∨ sad ∉ valid_SAD
   @grd3: sad ∈ SAS_rcv_admRequest
then
   @act2: SAS_rcv_admRequest ≔ SAS_rcv_admRequest ∖ {sad}
end

event SAS_Detect_fault
any
   f
where
   @grd1: f ∈ FAULT_PASSAGE_INDICATOR
then
   @act1: fault ≔ {f}
end

event SAS_Send_OperateCmd
any
   sad
   cmd
where
   @grd1: sad ∈ standby_SAD
   @grd2: cmd ∈ COMMAND
   @grd3: cmd ∉ channel_command
   @grd4: fault = {Overload}
   @grd5: sad ∉ Operate_executed_SAD
   @grd6: cmd_cntnt(cmd) = Operate_Cmd
   @grd7: cmdid(cmd) = sad
then
   @act1: channel_command ≔ channel_command ∪ {cmd}
   @act2: previous_cmd ≔ previous_cmd ∪ {cmd}
end

event SAD_Receive_Cmd
any
   sad

   cmd
where
   @grd1: sad ∈ registered_SAD
   @grd2: cmd ∈ channel_command
then
   @act3: SAD_received_command ≔ SAD_received_command ∪ {cmd}
   @act4: channel_command ≔ channel_command ∖ {cmd}
end

event SAD_ExecuteSend_MntrInfo_OperateCmd
any
   sad
   mntr
   cmd
   exe_res
where
   @grd1: sad ∈ standby_SAD
   @grd2: mntr ∈ MONITORING_INFO
   @grd3: exe_res ∈ MNTR_TYPE
   @grd5: cmd ∈ SAD_received_command
   @grd8: cmd_cntnt(cmd) = Operate_Cmd
   @grd9: cmdid(cmd) = sad
   @grd10: mntr_cntnt(mntr) = exe_res
   @grd11: mntrid(mntr) = sad
then
   @act3: channel_mntrInfo ≔ channel_mntrInfo ∪ {mntr}
   @act4: Operate_executed_SAD ≔ Operate_executed_SAD ∪ {sad}
   @act8: SAD_received_command ≔ SAD_received_command ∖ {cmd}
end

event SAD_ExecuteSend_MntrInfo_OtherCmd
any
   sad
   mntr
   cmd
   exe_res
where
   @grd1: sad ∈ operational_SAD ∨ sad ∈ unoperational_SAD
   @grd2: mntr ∈ MONITORING_INFO
   @grd3: exe_res ∈ MNTR_TYPE
   @grd4: sad ∉ AllCmd_executed_SAD
   @grd5: cmd ∈ SAD_received_command
   @grd6: cmd_cntnt(cmd) ≠ Operate_Cmd ∧ cmd_cntnt(cmd) ≠ Release_Cmd
   @grd7: cmdid(cmd) = sad
   @grd8: mntr_cntnt(mntr) = exe_res
   @grd9: mntrid(mntr) = sad
then
   @act1: channel_mntrInfo ≔ channel_mntrInfo ∪ {mntr}
   @act2: AllCmd_executed_SAD ≔ AllCmd_executed_SAD ∪ {sad}
   @act3: SAD_received_command ≔ SAD_received_command ∖ {cmd}
end

event SAD_ExecuteSend_MntrInfo_ReleaseCmd
any
   sad
   mntr
   cmd
   exe_res
where
   @grd1: sad ∈ operational_SAD ∨ sad ∈ unoperational_SAD
   @grd2: mntr ∈ MONITORING_INFO
   @grd3: exe_res ∈ MNTR_TYPE
   @grd6: cmd ∈ SAD_received_command
   @grd9: cmd_cntnt(cmd) = Release_Cmd
   @grd10: cmdid(cmd) = sad
   @grd11: mntr_cntnt(mntr) = exe_res
   @grd12: mntrid(mntr) = sad
then
   @act3: channel_mntrInfo ≔ channel_mntrInfo ∪ {mntr}
   @act6: SAD_received_command ≔ SAD_received_command ∖ {cmd}
   @act7: Release_executed_SAD ≔ Release_executed_SAD ∪ {sad}
end

event SAS_Receive_MntrInfo
any
   sad
   mntr
where
   @grd1: sad ∈ registered_SAD
   @grd2: mntr ∈ channel_mntrInfo
then
   @act3: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∪ {mntr}
   @act4: channel_mntrInfo ≔ channel_mntrInfo ∖ {mntr}
end

event SAS_CheckMntrSend_Cmd_OperateCmd_Success
any
   sad
   mntr
   cmd
where
   @grd1: sad ∈ standby_SAD
   @grd2: mntr ∈ SAS_received_mntrInfo
```

```
    @grd3: cmd ∈ COMMAND
    @grd4: mntr_cntnt(mntr) = Normal_info
    @grd5: mntrid(mntr) = sad
    @grd6: cmdid(cmd) = sad
    @grd7: cmd_cntnt(cmd) = Normal_Cmd
    @grd8: sad ∈ Operate_executed_SAD
then
    @act1: operational_SAD ≔ operational_SAD ∪ {sad}
    @act2: channel_command ≔ channel_command ∪ {cmd}
    @act3: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∖ {mntr}
    @act4: standby_SAD ≔ standby_SAD ∖ {sad}
    @act6: Operate_executed_SAD ≔ Operate_executed_SAD ∖ {sad}
end

event SAS_CheckMntrSend_Cmd_OperateCmd_Fail
any
    sad
    mntr
    cmd
    corCmd
where
    @grd1: sad ∈ standby_SAD
    @grd2: mntr ∈ SAS_received_mntrInfo
    @grd3: corCmd ∈ Corrective_Cmd
    @grd4: cmd ∈ COMMAND
    @grd5: mntr_cntnt(mntr) ∈ Transient_failure
    @grd6: mntrid(mntr) = sad
    @grd7: cmdid(cmd) = sad
    @grd8: cmd_cntnt(cmd) = corCmd
    @grd9: sad ∈ Operate_executed_SAD
then
    @act1: unoperational_SAD ≔ unoperational_SAD ∪ {sad}
    @act2: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∖ {mntr}
    @act3: channel_command ≔ channel_command ∪ {cmd}
    @act4: standby_SAD ≔ standby_SAD ∖ {sad}
    @act5: Operate_executed_SAD ≔ Operate_executed_SAD ∖ {sad}
end

event SAS_CheckMntrSend_Cmd_OP_Normal
any
    sad
    cmd
    mntr
    type
where
    @grd1: sad ∈ operational_SAD
    @grd2: cmd ∈ COMMAND
    @grd3: fault = {Overload}
    @grd4: mntr ∈ SAS_received_mntrInfo
    @grd5: type ∈ MNTR_TYPE
    @grd7: mntr_cntnt(mntr) = Normal_info
    @grd8: mntrid(mntr) = sad
    @grd9: cmd_cntnt(cmd) = Normal_Cmd
    @grd10: cmdid(cmd) = sad
    @grd11: sad ∉ Release_executed_SAD
    @grd12: sad ∈ AllCmd_executed_SAD
then
    @act1: channel_command ≔channel_command ∪ {cmd}
    @act2: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∖ {mntr}
    @act5: AllCmd_executed_SAD ≔ AllCmd_executed_SAD ∖ {sad}
end

    event SAS_CheckMntrSend_Cmd_OP_Release
any
    sad
    cmd
    mntr
where
    @grd1: sad ∈ operational_SAD
    @grd2: fault = {Oversupply}
    @grd3: cmd ∈ COMMAND
    @grd4: mntr ∈ SAS_received_mntrInfo
    @grd7: mntr_cntnt(mntr) = Normal_info
    @grd8: mntrid(mntr) = sad
    @grd9: cmd_cntnt(cmd) = Release_Cmd
    @grd10: cmdid(cmd) = sad
    @grd11: sad ∉ Release_executed_SAD
    @grd12: sad ∈ AllCmd_executed_SAD
 then
    @act1: channel_command ≔ channel_command ∪ {cmd}
    @act2: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∖ {mntr}
    @act5: AllCmd_executed_SAD ≔ AllCmd_executed_SAD ∖ {sad}
 end

 event SAS_CheckMntrSend_Cmd_OP_TransFailure
 any
    sad
    cmd
    mntr
    corCmd
 where
    @grd1: sad ∈ operational_SAD
    @grd2: cmd ∈ COMMAND
    @grd3: corCmd ∈ Corrective_Cmd
    @grd6: mntr ∈ SAS_received_mntrInfo
    @grd7: mntr_cntnt(mntr) ∈ Transient_failure
    @grd8: mntrid(mntr) = sad
    @grd9: cmd_cntnt(cmd) = corCmd
    @grd10: cmdid(cmd) = sad
    @grd11: sad ∉ Release_executed_SAD
    @grd12: sad ∈ AllCmd_executed_SAD
 then
    @act1: unoperational_SAD ≔ unoperational_SAD ∪ {sad}
    @act2: operational_SAD ≔ operational_SAD ∖ {sad}
    @act3: channel_command ≔ channel_command ∪ {cmd}
    @act4: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∖ {mntr}
    @act7: AllCmd_executed_SAD ≔ AllCmd_executed_SAD ∖ {sad}
 end

event SAS_CheckMntrSend_Cmd_UNOP_Normal1
 any
    sad
    mntr
    cmd
 where
    @grd1: sad ∈ unoperational_SAD
    @grd2: cmd ∈ COMMAND
    @grd3: mntr ∈ SAS_received_mntrInfo
    @grd4: mntr_cntnt(mntr) = Normal_info
    @grd5: mntrid(mntr) = sad
    @grd6: cmd_cntnt(cmd) = Release_Cmd
    @grd7: cmdid(cmd) = sad
    @grd8: sad ∈ AllCmd_executed_SAD
 then
    @act3: channel_command ≔ channel_command ∪ {cmd}
    @act4: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∖ {mntr}
    @act5: AllCmd_executed_SAD ≔ AllCmd_executed_SAD ∖ {sad}
 end

 event SAS_CheckMntrSend_Cmd_UNOP_Normal2
 any
    sad
    cmd
    mntr
 where
    @grd1: sad ∈ unoperational_SAD
    @grd2: cmd ∈ COMMAND
    @grd3: mntr ∈ SAS_received_mntrInfo
    @grd6: mntr_cntnt(mntr) = Normal_info
    @grd7: mntrid(mntr) = sad
    @grd8: cmd_cntnt(cmd) = Normal_Cmd
    @grd9: cmdid(cmd) = sad
    @grd10: sad ∈ AllCmd_executed_SAD
 then
    @act1: operational_SAD ≔ operational_SAD ∪ {sad}
    @act2: unoperational_SAD ≔ unoperational_SAD ∖ {sad}
```

```
        @act3: channel_command ≔ channel_command ∪ {cmd}
        @act4: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∖ {mntr}
        @act5: AllCmd_executed_SAD ≔ AllCmd_executed_SAD ∖ {sad}
    end

    event SAS_CheckMntrSend_Cmd_UNOP_TransFailure
    any
        sad
        cmd
        mntr
        corCmd
    where
        @grd1: sad ∈ unoperational_SAD
        @grd2: cmd ∈ COMMAND
        @grd3: corCmd ∈ Corrective_Cmd
        @grd4: mntr ∈ SAS_received_mntrInfo
        @grd7: mntr_cntnt(mntr) ∈ Transient_failure
        @grd8: mntrid(mntr) = sad
        @grd9: cmd_cntnt(cmd) = corCmd
        @grd10: cmdid(cmd) = sad
        @grd11: sad ∈ AllCmd_executed_SAD
    then
        @act1: channel_command ≔ channel_command ∪ {cmd}
        @act2: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∖ {mntr}
        @act5: AllCmd_executed_SAD ≔ AllCmd_executed_SAD ∖ {sad}
    end

    event SAS_DeregisterSAD_OP_PermFailure
    any
        sad mntr
    where
        @grd1: sad ∈ operational_SAD
        @grd2: mntr ∈ SAS_received_mntrInfo
        @grd4: mntr_cntnt(mntr) ∈ Permanent_failure
        @grd5: mntrid(mntr) = sad
    then
        @act1: retired_SAD ≔ retired_SAD ∪ {sad}
        @act2: operational_SAD ≔ operational_SAD ∖ {sad}
        @act3: registered_SAD ≔ registered_SAD ∖ {sad}
        @act4: valid_SAD ≔ valid_SAD ∖ {sad}
        @act5: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∖ {mntr}
        @act9: Operate_executed_SAD ≔ Operate_executed_SAD ∖ {sad}
        @act10: Release_executed_SAD ≔ Release_executed_SAD ∖ {sad}
        @act11: AllCmd_executed_SAD ≔ AllCmd_executed_SAD ∖ {sad}
    end

    event SAS_DeregisterSAD_UNOP_PermFailure
    any
        sad
        mntr
    where
        @grd1: sad ∈ unoperational_SAD
        @grd2: mntr ∈ SAS_received_mntrInfo
        @grd3: mntr_cntnt(mntr) ∈ Permanent_failure
        @grd4: mntrid(mntr) = sad
    then
        @act1: retired_SAD ≔ retired_SAD ∪ {sad}
        @act2: unoperational_SAD ≔ unoperational_SAD ∖ {sad}
        @act3: registered_SAD ≔ registered_SAD ∖ {sad}
        @act4: valid_SAD ≔ valid_SAD ∖ {sad}
        @act5: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∖ {mntr}
        @act6: Operate_executed_SAD ≔ Operate_executed_SAD ∖ {sad}
        @act7: Release_executed_SAD ≔ Release_executed_SAD ∖ {sad}
        @act8: AllCmd_executed_SAD ≔ AllCmd_executed_SAD ∖ {sad}
    end

event SAS_DeregisterSAD_Standby_PermFailure
    any
        sad mntr
    where
        @grd1: sad ∈ standby_SAD
        @grd2: mntr ∈ SAS_received_mntrInfo
        @grd4: mntr_cntnt(mntr) ∈ Permanent_failure
```

```
   @grd5: mntrid(mntr) = sad
then
   @act1: retired_SAD ≔ retired_SAD ∪ {sad}
   @act2: standby_SAD ≔ standby_SAD \ {sad}
   @act3: registered_SAD ≔ registered_SAD \ {sad}
   @act4: valid_SAD ≔ valid_SAD \ {sad}
   @act5: SAS_received_mntrInfo ≔ SAS_received_mntrInfo \ {mntr}
   @act9: Operate_executed_SAD ≔ Operate_executed_SAD \ {sad}
   @act10: Release_executed_SAD ≔ Release_executed_SAD \ {sad}
   @act11: AllCmd_executed_SAD ≔ AllCmd_executed_SAD \ {sad}
end

event SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Success
any
   sad
   mntr
where
   @grd1: sad ∈ operational_SAD
   @grd2: mntr ∈ SAS_received_mntrInfo
   @grd4: mntr_cntnt(mntr) = Normal_info
   @grd5: mntrid(mntr) = sad
   @grd10: sad ∈ Release_executed_SAD
then
   @act1: standby_SAD ≔ standby_SAD ∪ {sad}
   @act2: SAS_received_mntrInfo ≔ SAS_received_mntrInfo \ {mntr}
   @act5: operational_SAD ≔ operational_SAD \ {sad}
   @act9: Release_executed_SAD ≔ Release_executed_SAD \ {sad}
end

event SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Fail
any
   sad
   mntr
   cmd
   corCmd
where
   @grd1: sad ∈ operational_SAD
   @grd2: mntr ∈ SAS_received_mntrInfo
   @grd3: corCmd ∈ Corrective_Cmd
   @grd4: cmd ∈ COMMAND
   @grd5: mntr_cntnt(mntr) ∈ Transient_failure
   @grd6: mntrid(mntr) = sad
   @grd7: cmdid(cmd) = sad
   @grd8: cmd_cntnt(cmd) = corCmd
   @grd9: sad ∈ Release_executed_SAD
then
   @act1: unoperational_SAD ≔ unoperational_SAD ∪ {sad}
   @act2: SAS_received_mntrInfo ≔ SAS_received_mntrInfo \ {mntr}
   @act3: channel_command ≔ channel_command ∪ {cmd}
   @act4: operational_SAD ≔ operational_SAD \ {sad}
   @act5: Release_executed_SAD ≔ Release_executed_SAD \ {sad}
end


event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Success
any
   sad
   mntr
where
   @grd1: sad ∈ unoperational_SAD
   @grd2: mntr ∈ SAS_received_mntrInfo
   @grd3: mntr_cntnt(mntr) = Normal_info
   @grd4: mntrid(mntr) = sad
   @grd5: sad ∈ Release_executed_SAD
then
   @act1: standby_SAD ≔ standby_SAD ∪ {sad}
   @act2: SAS_received_mntrInfo ≔ SAS_received_mntrInfo \ {mntr}
   @act3: unoperational_SAD ≔ unoperational_SAD \ {sad}
   @act4: Release_executed_SAD ≔ Release_executed_SAD \ {sad}
end

event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Fail
any
```

```
      sad
      mntr
      cmd
      corCmd
   where
      @grd1: sad ∈ unoperational_SAD
      @grd2: mntr ∈ SAS_received_mntrInfo
      @grd3: corCmd ∈ Corrective_Cmd
      @grd4: cmd ∈ COMMAND
      @grd5: mntr_cntnt(mntr) ∈ Transient_failure
      @grd6: mntrid(mntr) = sad
      @grd7: cmdid(cmd) = sad
      @grd8: cmd_cntnt(cmd) = corCmd
      @grd9: sad ∈ Release_executed_SAD
   then
      @act1: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∖ {mntr}
      @act2: channel_command ≔ channel_command ∪ {cmd}
      @act4: Release_executed_SAD ≔ Release_executed_SAD ∖ {sad}

   End
```

- ## The Context of the First Refinement C1

```
context C1
extends C0
constants
   Overheat
   Malfunction
   SW_outdated
   Finishing_prev_cmd
   Life_expire
   Persist_malfunction
   Memory_failure
   Troubleshooting_Cmd
   Patching_Cmd
   Acknowledge_powerON
   Acknowledge_finishPrev
   OTHER_CORRECTIVE_CMD
axioms
   @axm24: Overheat ∈ Transient_failure
   @axm25: Malfunction ∈ Transient_failure
   @axm26: SW_outdated ∈ Transient_failure
   @axm27: Finishing_prev_cmd ∈ Transient_failure
   @axm29: Life_expire ∈ Permanent_failure
   @axm30: Memory_failure ∈ Permanent_failure
   @axm31: Acknowledge_powerON ∈ Corrective_Cmd
   @axm32: Troubleshooting_Cmd ∈ Corrective_Cmd
   @axm33: Patching_Cmd ∈ Corrective_Cmd
   @axm34: Acknowledge_finishPrev ∈ Corrective_Cmd
   @axm36: OTHER_CORRECTIVE_CMD ⊆ Corrective_Cmd
   @Valid_Mntr2: partition (Transient_failure, {Overheat}, {Malfunction}, {SW_outdated},
{Finishing_prev_cmd})
   @Valid_Mntr3: partition (Permanent_failure, {Life_expire}, {Memory_failure}, {Persist_malfunction})
   @Valid_Command2: partition (Corrective_Cmd, {Acknowledge_powerON}, {Troubleshooting_Cmd}, {Patching_Cmd},
{Acknowledge_finishPrev}, OTHER_CORRECTIVE_CMD)

End
```

- ## The Machine of the First Refinement M1

```
machine M1
refines M0
sees C1
variables
   valid_SAD
   registered_SAD
   unregistered_SAD
   retired_SAD
   standby_SAD
   channel_AdmRequest
    SAS_rcv_admRequest
   unoperational_SAD
   operational_SAD
   OTHER_UNOPERATIONAL_SAD
   Operate_executed_SAD
   AllCmd_executed_SAD
   rolling_blackout_SAD
   on_repair_SAD
   on_upgrade_SAD
   busy_SAD
   Release_executed_SAD
   fault
   channel_command
   channel_mntrInfo
   //Ch_received_command
   SAD_received_command
   //Ch_received_mntrInfo
   SAS_received_mntrInfo
   previous_cmd
invariants
   @inv60: rolling_blackout_SAD ⊆ unoperational_SAD
   @inv61: on_repair_SAD ⊆ unoperational_SAD
   @inv62: on_upgrade_SAD ⊆ unoperational_SAD
   @inv63: busy_SAD ⊆ unoperational_SAD
   @inv65: OTHER_UNOPERATIONAL_SAD ⊆ unoperational_SAD
   @inv66: partition (unoperational_SAD, rolling_blackout_SAD, on_repair_SAD, on_upgrade_SAD, busy_SAD,
OTHER_UNOPERATIONAL_SAD)

events
   event INITIALISATION extends INITIALISATION
   then
      @act60: rolling_blackout_SAD ≔ ∅
      @act61: on_repair_SAD ≔ ∅
      @act62: on_upgrade_SAD ≔ ∅
      @act63: busy_SAD ≔ ∅
      @act65: OTHER_UNOPERATIONAL_SAD ≔ ∅
   end

   event SAD_Send_Admission_Request extends SAD_Send_Admission_Request
   end


   event SAS_Receive_Admission_Request extends SAS_Receive_Admission_Request
   end

   event SAS_RegSAD_Success extends SAS_RegSAD_Success
   end

   event SAS_RegSAD_Fail extends SAS_RegSAD_Fail
   end

   event SAS_Detect_fault extends SAS_Detect_fault
   end

   event SAS_Send_OperateCmd extends SAS_Send_OperateCmd
   end

   event SAD_Receive_Cmd extends SAD_Receive_Cmd
   end

   event SAD_ExecuteSend_MntrInfo_OperateCmd extends SAD_ExecuteSend_MntrInfo_OperateCmd
   end

   event SAD_ExecuteSend_MntrInfo_OtherCmd extends SAD_ExecuteSend_MntrInfo_OtherCmd
   end

   event SAD_ExecuteSend_MntrInfo_ReleaseCmd extends SAD_ExecuteSend_MntrInfo_ReleaseCmd
   end

   event SAS_Receive_MntrInfo extends SAS_Receive_MntrInfo
   end
   event SAS_CheckMntrSend_Cmd_OperateCmd_Success extends SAS_CheckMntrSend_Cmd_OperateCmd_Success
   end
   event SAS_CheckMntrSend_Cmd_OP_Normal extends SAS_CheckMntrSend_Cmd_OP_Normal
   end

   event SAS_CheckMntrSend_Cmd_OP_Release extends SAS_CheckMntrSend_Cmd_OP_Release
   end

   event SAS_DeregisterSAD_OP_PermFailure extends SAS_DeregisterSAD_OP_PermFailure
   end

   event SAS_DeregisterSAD_Standby_PermFailure extends SAS_DeregisterSAD_Standby_PermFailure
   end

   event SAS_CheckMntrSend_Cmd_OP_Overheat extends SAS_CheckMntrSend_Cmd_OP_TransFailure //the Transient
failure is Overheat
   where
      @grd15: corCmd = Acknowledge_powerON
      @grd16: mntr_cntnt(mntr) = Overheat
   then
      @act10: rolling_blackout_SAD ≔ rolling_blackout_SAD ∪ {sad}
   end

   event SAS_CheckMntrSend_Cmd_Rolling_PowerON_Success extends SAS_CheckMntrSend_Cmd_UNOP_Normal1
   where
      @grd9: sad ∈ rolling_blackout_SAD
   end

   event SAS_CheckMntrSend_Cmd_Rolling_PowerON_Fail extends SAS_CheckMntrSend_Cmd_UNOP_TransFailure
   where
      @grd14: sad ∈ rolling_blackout_SAD
      @grd15: mntr_cntnt(mntr) = Overheat
      @grd16: corCmd = Acknowledge_powerON
   end

   event SAS_CheckMntrSend_Cmd_OP_Malfunction extends SAS_CheckMntrSend_Cmd_OP_TransFailure //the Transient
failure is Malfunction
   where
      @grd15: corCmd = Troubleshooting_Cmd
      @grd16: mntr_cntnt(mntr) = Malfunction
   then
      @act10: on_repair_SAD ≔ on_repair_SAD ∪ {sad}
   end

   event SAS_CheckMntrSend_Cmd_Repair_Troubleshooting_Success extends SAS_CheckMntrSend_Cmd_UNOP_Normal1
   where
      @grd9: sad ∈ on_repair_SAD
   end

   event SAS_CheckMntrSend_Cmd_Repair_Troubleshooting_Fail extends SAS_CheckMntrSend_Cmd_UNOP_TransFailure
   where
      @grd14: sad ∈ on_repair_SAD
      @grd15: mntr_cntnt(mntr) = Malfunction
      @grd16: corCmd = Troubleshooting_Cmd
   end

   event SAS_CheckMntrSend_Cmd_OP_SwOutdated extends SAS_CheckMntrSend_Cmd_OP_TransFailure //the Transient
failure is SW Outdated
   where
      @grd15: corCmd = Patching_Cmd
      @grd16: mntr_cntnt(mntr) = SW_outdated
   then
      @act10: on_upgrade_SAD ≔ on_upgrade_SAD ∪ {sad}
   end

   event SAS_CheckMntrSend_Cmd_Upgrade_Patch_Success extends SAS_CheckMntrSend_Cmd_UNOP_Normal1
   where
      @grd9: sad ∈ on_upgrade_SAD
   end

   event SAS_CheckMntrSend_Cmd_Upgrade_Patch_Fail extends SAS_CheckMntrSend_Cmd_UNOP_TransFailure
   where
      @grd14: sad ∈ on_upgrade_SAD
      @grd15: mntr_cntnt(mntr) = SW_outdated
      @grd16: corCmd = Patching_Cmd
   end
   event SAS_CheckMntrSend_Cmd_OP_FnshPrevCmd extends SAS_CheckMntrSend_Cmd_OP_TransFailure //the Transient
failure is finishing the previous command
   where
      @grd15: corCmd = Acknowledge_finishPrev
      @grd16: mntr_cntnt(mntr) = Finishing_prev_cmd
   then
      @act10: busy_SAD ≔ busy_SAD ∪ {sad}
   end

   event SAS_CheckMntrSend_Cmd_Busy_FnshPrevCmd_Success extends SAS_CheckMntrSend_Cmd_UNOP_Normal2
   where
      @grd13: sad ∈ busy_SAD
   then
      @act12: busy_SAD ≔ busy_SAD \ {sad}
   end

   event SAS_CheckMntrSend_Cmd_Busy_FnshPrevCmd_Fail extends SAS_CheckMntrSend_Cmd_UNOP_TransFailure
   where
      @grd14: sad ∈ busy_SAD
      @grd15: mntr_cntnt(mntr) = Finishing_prev_cmd
      @grd16: corCmd = Acknowledge_finishPrev
   end

   event SAS_DeregisterSAD_Rolling_PermFailure extends SAS_DeregisterSAD_UNOP_PermFailure
   where
      @grd15: sad ∈ rolling_blackout_SAD
      @grd16: mntr_cntnt(mntr) = Life_expire ∨ mntr_cntnt(mntr) = Memory_failure
   then
      @act13: rolling_blackout_SAD ≔ rolling_blackout_SAD \ {sad}
   end

   event SAS_DeregisterSAD_Repair_PermFailure extends SAS_DeregisterSAD_UNOP_PermFailure
   where
      @grd15: sad ∈ on_repair_SAD
      @grd16: mntr_cntnt(mntr) = Life_expire ∨ mntr_cntnt(mntr) = Memory_failure
   then
      @act13: on_repair_SAD ≔ on_repair_SAD \ {sad}
   end

   event SAS_DeregisterSAD_Upgrade_PermFailure extends SAS_DeregisterSAD_UNOP_PermFailure
   where
      @grd15: sad ∈ on_upgrade_SAD
      @grd16: mntr_cntnt(mntr) = Life_expire ∨ mntr_cntnt(mntr) = Memory_failure
   then
      @act13: on_upgrade_SAD ≔ on_upgrade_SAD \ {sad}
   end

   event SAS_DeregisterSAD_Busy_PermFailure extends SAS_DeregisterSAD_UNOP_PermFailure
   where
      @grd15: sad ∈ busy_SAD
      @grd16: mntr_cntnt(mntr) = Life_expire ∨ mntr_cntnt(mntr) = Memory_failure
   then
      @act13: busy_SAD ≔ busy_SAD \ {sad}
   end

   event SAS_CheckMntrSend_Cmd_OperateCmd_Fail_Overheat extends SAS_CheckMntrSend_Cmd_OperateCmd_Fail
   where
      @grd12: mntr_cntnt(mntr) = Overheat
      @grd13: corCmd = Acknowledge_powerON
   then
      @act9: rolling_blackout_SAD ≔ rolling_blackout_SAD ∪ {sad}
   end

   event SAS_CheckMntrSend_Cmd_OperateCmd_Fail_Malfunction extends SAS_CheckMntrSend_Cmd_OperateCmd_Fail
   where
      @grd12: mntr_cntnt(mntr) = Malfunction
      @grd13: corCmd = Troubleshooting_Cmd
   then
      @act9: on_repair_SAD ≔ on_repair_SAD ∪ {sad}
   end

   event SAS_CheckMntrSend_Cmd_OperateCmd_Fail_SwOutdated extends SAS_CheckMntrSend_Cmd_OperateCmd_Fail
   where
      @grd12: mntr_cntnt(mntr) = SW_outdated
      @grd13: corCmd = Patching_Cmd
   then
      @act9: on_upgrade_SAD ≔ on_upgrade_SAD ∪ {sad}
   end

   event SAS_CheckMntrSend_Cmd_OperateCmd_Fail_FinishingPrevCmd extends SAS_CheckMntrSend_Cmd_OperateCmd_Fail
   where
      @grd12: mntr_cntnt(mntr) = Finishing_prev_cmd
      @grd13: corCmd = Acknowledge_finishPrev
   then
      @act9: busy_SAD ≔ busy_SAD ∪ {sad}
   end


   event SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Fail_Overheat extends SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Fail
   where
      @grd13: mntr_cntnt(mntr) = Overheat
      @grd14: corCmd = Acknowledge_powerON
   then
      @act9: rolling_blackout_SAD ≔ rolling_blackout_SAD ∪ {sad}
   end

   event SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Fail_Malfunction extends
SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Fail
   where
      @grd13: mntr_cntnt(mntr) = Malfunction
      @grd14: corCmd = Troubleshooting_Cmd
   then
      @act9: on_repair_SAD ≔ on_repair_SAD ∪ {sad}
   end

   event SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Fail_SwOutdated extends SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Fail
   where
      @grd13: mntr_cntnt(mntr) = SW_outdated
      @grd14: corCmd = Patching_Cmd
   then
      @act9: on_upgrade_SAD ≔ on_upgrade_SAD ∪ {sad}
   end

   event SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Fail_FinishingPrevCmd extends
SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Fail
   where
      @grd13: mntr_cntnt(mntr) = Finishing_prev_cmd
      @grd14: corCmd = Acknowledge_finishPrev
   then
      @act9: busy_SAD ≔ busy_SAD ∪ {sad}
   end

   event SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Success extends SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Success
   end

   event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Success_Overheat extends
SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Success
   where
      @grd6: sad ∈ rolling_blackout_SAD
   then
      @act5: rolling_blackout_SAD ≔ rolling_blackout_SAD ∖ {sad}
   end

   event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Success_Malfunction extends
SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Success
   where
      @grd6: sad ∈ on_repair_SAD
   then
      @act5: on_repair_SAD ≔ on_repair_SAD ∖ {sad}
   end

   event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Success_SwOutdated extends
SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Success
   where
      @grd6: sad ∈ on_upgrade_SAD
   then
      @act5: on_upgrade_SAD ≔ on_upgrade_SAD ∖ {sad}
   end
   event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Success_FinishingPrevCmd extends
SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Success
   where
      @grd6: sad ∈ busy_SAD
   then
      @act5: busy_SAD ≔ busy_SAD ∖ {sad}
   end


   event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Fail_Overheat extends
SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Fail
   where
      @grd10: sad ∈ rolling_blackout_SAD
      @grd11: mntr_cntnt(mntr) = Overheat
      @grd12: corCmd = Acknowledge_powerON
   end

   event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Fail_Malfunction extends
SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Fail
   where
      @grd10: sad ∈ on_repair_SAD
      @grd11: mntr_cntnt(mntr) = Malfunction
      @grd12: corCmd = Troubleshooting_Cmd
   end

   event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Fail_SwOutdated extends
SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Fail
   where
      @grd10: sad ∈ on_upgrade_SAD
      @grd11: mntr_cntnt(mntr) = SW_outdated
      @grd12: corCmd = Patching_Cmd
   end

   event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Fail_FinishingPrevCmd extends
SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Fail
   where
      @grd10: sad ∈ busy_SAD
      @grd11: mntr_cntnt(mntr) = Finishing_prev_cmd
      @grd12: corCmd = Acknowledge_finishPrev
   end


      event SAS_CheckMntrSend_Cmd_UNOP_Normal2_OtherUNOP extends SAS_CheckMntrSend_Cmd_UNOP_Normal2
   where
      @grd11: sad ∈ OTHER_UNOPERATIONAL_SAD
   then
      @act6: OTHER_UNOPERATIONAL_SAD ≔ OTHER_UNOPERATIONAL_SAD ∖ {sad}
   end


   event SAS_CheckMntrSend_Cmd_OP_OtherFailures extends SAS_CheckMntrSend_Cmd_OP_OtherFailures
   then
      @act8: OTHER_UNOPERATIONAL_SAD ≔ OTHER_UNOPERATIONAL_SAD ∪ {sad}
   end

   event SAS_CheckMntrSend_Cmd_UNOP_Upgrade_OtherFailures extends SAS_CheckMntrSend_Cmd_UNOP_OtherFailures
      where
       @grd14: sad ∈ on_upgrade_SAD
    then
      @act10: on_upgrade_SAD ≔ on_upgrade_SAD ∖ {sad}
      @act11: OTHER_UNOPERATIONAL_SAD ≔ OTHER_UNOPERATIONAL_SAD ∪ {sad}
   end

   event SAS_CheckMntrSend_Cmd_UNOP_Repair_OtherFailures extends SAS_CheckMntrSend_Cmd_UNOP_OtherFailures
      where
       @grd14: sad ∈ on_repair_SAD
   end

   event SAS_CheckMntrSend_Cmd_Corrective_Fail extends SAS_CheckMntrSend_Cmd_Corrective_Fail
   end

   event SAS_CheckMntrSend_Cmd_OperateCmd_OtherFailures extends
SAS_CheckMntrSend_Cmd_OperateCmd_OtherFailures
```

```
   then
      @act11: OTHER_UNOPERATIONAL_SAD ≔ OTHER_UNOPERATIONAL_SAD ∪ {sad}
   end

   event SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_OtherFailures extends
SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_OtherFailures
   then
      @act11: OTHER_UNOPERATIONAL_SAD ≔ OTHER_UNOPERATIONAL_SAD ∪ {sad}
   end

   event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_OtherFailures extends
SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_OtherFailures
   end

   event SAD_ExecuteSend_OtherMntrType extends SAD_ExecuteSend_OtherMntrType
   end
      event SAD_Receive_Cmd_Failure extends SAD_Receive_Cmd_Failure
   end

   event SAS_CheckMntrSend_Cmd_Failure extends SAS_CheckMntrSend_Cmd_Failure
   end

end
```

Event-B code for Other Failures-related Events

```
   event SAS_CheckMntrSend_Cmd_OP_OtherFailures
   any
      sad
      cmd
      mntr
      corCmd
   where
       @grd1: sad ∈ operational_SAD
      @grd2: cmd ∈ COMMAND
      @grd3: corCmd ∈ Corrective_Cmd
      @grd4: mntr ∈ SAS_received_mntrInfo
   then
      @act2: operational_SAD ≔ operational_SAD ∖ {sad}
      @act3: channel_command ≔ channel_command ∪ {cmd}
      @act4: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∖ {mntr}
      @act5: unoperational_SAD ≔  unoperational_SAD ∪ {sad}
      @act7: AllCmd_executed_SAD ≔ AllCmd_executed_SAD ∖ {sad}
      @act1: previous_cmd ≔ previous_cmd ∪ {cmd}
   end

   event SAS_CheckMntrSend_Cmd_UNOP_OtherFailures
   any
      sad
      cmd
      mntr
      corCmd
   where
      @grd1: sad ∈ unoperational_SAD
      @grd2: cmd ∈ COMMAND
      @grd3: corCmd ∈ Corrective_Cmd
      @grd4: mntr ∈ SAS_received_mntrInfo
   then
      @act3: channel_command ≔ channel_command ∪ {cmd}
      @act4: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∖ {mntr}
      @act7: AllCmd_executed_SAD ≔ AllCmd_executed_SAD ∖ {sad}
      @act1: previous_cmd ≔ previous_cmd ∪ {cmd}
   end

   event SAS_CheckMntrSend_Cmd_Corrective_Fail
    any
        sad
        mntr
        cmd
        corCmd
```

```
 where
   @grd1: sad ∈ unoperational_SAD
   @grd2: cmd ∈ COMMAND
   @grd3: corCmd ∈ Corrective_Cmd
   @grd4: mntr ∈ SAS_received_mntrInfo
then
   @act1: channel_command ≔ channel_command ∪ {cmd}
   @act2: SAS_received_mntrInfo ≔ SAS_received_mntrInfo \ {mntr}
   @act3: AllCmd_executed_SAD ≔ AllCmd_executed_SAD \ {sad}
   @act4: previous_cmd ≔ previous_cmd ∪ {cmd}
end

event SAS_CheckMntrSend_Cmd_OperateCmd_OtherFailures
any
     sad
     mntr
     cmd
     corCmd
 where
   @grd1: sad ∈ standby_SAD
   @grd2: cmd ∈ COMMAND
   @grd3: corCmd ∈ Corrective_Cmd
   @grd4: mntr ∈ SAS_received_mntrInfo
then
   @act2: standby_SAD ≔standby_SAD \ {sad}
   @act3: channel_command ≔ channel_command ∪ {cmd}
   @act4: SAS_received_mntrInfo ≔ SAS_received_mntrInfo \ {mntr}
   @act7: Operate_executed_SAD ≔ Operate_executed_SAD \ {sad}
   @act8: unoperational_SAD ≔ unoperational_SAD ∪ {sad}
   @act1: previous_cmd ≔ previous_cmd ∪ {cmd}
end

event SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_OtherFailures
any
   sad
   mntr
   cmd
   corCmd
where
   @grd1: sad ∈ operational_SAD
   @grd2: mntr ∈ SAS_received_mntrInfo
   @grd3: corCmd ∈ Corrective_Cmd
   @grd4: cmd ∈ COMMAND
then
    @act2: SAS_received_mntrInfo ≔ SAS_received_mntrInfo \ {mntr}
   @act3: channel_command ≔ channel_command ∪ {cmd}
   @act4: operational_SAD ≔ operational_SAD \ {sad}
   @act5: Release_executed_SAD ≔ Release_executed_SAD \ {sad}
   @act7: unoperational_SAD ≔ unoperational_SAD ∪ {sad}
   @act1: previous_cmd ≔ previous_cmd ∪ {cmd}
end

event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_OtherFailures
any
   sad
   mntr
   cmd
   corCmd
where
   @grd1: sad ∈ unoperational_SAD
   @grd2: mntr ∈ SAS_received_mntrInfo
   @grd3: corCmd ∈ Corrective_Cmd
   @grd4: cmd ∈ COMMAND
then
   @act1: SAS_received_mntrInfo ≔ SAS_received_mntrInfo \ {mntr}
   @act2: channel_command ≔ channel_command ∪ {cmd}
   @act4: Release_executed_SAD ≔ Release_executed_SAD \ {sad}
   @act5: previous_cmd ≔ previous_cmd ∪ {cmd}
end

event SAD_ExecuteSend_OtherMntrType
any
   sad
```

```
        mntr
        cmd
        exe_res
    where
        @grd1: sad ∈ registered_SAD
        @grd2: mntr ∈ MONITORING_INFO
        @grd3: exe_res ∈ MNTR_TYPE
        @grd5: cmd ∈ SAD_received_command
    then
        @act3: channel_mntrInfo ≔ channel_mntrInfo ∪ {mntr}
        @act4: SAD_received_command ≔ SAD_received_command ∖ {cmd}
    end

    event SAD_Receive_Cmd_Failure
    any
        cmd
        mntr
    where
        @grd2: cmd ∈ channel_command
        @grd4: mntr ∈ MONITORING_INFO
    then
        @act1: channel_command ≔ channel_command ∖ {cmd}
        @act2: channel_mntrInfo ≔ channel_mntrInfo ∪ {mntr}
    end

    event SAS_CheckMntrSend_Cmd_Failure
    any
         mntr
         cmd
    where
        @grd1: cmd ∈ channel_command
        @grd2: mntr ∈ SAS_received_mntrInfo
    then
        @act2: channel_command ≔ channel_command ∪ {cmd}
        @act3: SAS_received_mntrInfo ≔ SAS_received_mntrInfo ∖ {mntr}
        @act1: previous_cmd ≔ previous_cmd ∪ {cmd}
    end

end
```

• The Context of the Second Refinement C2

```
context C2
extends C1
sets
    MESSAGE
    CYPHER_COMMAND
    KEY
constants
     Security_Alert
     isolate_cmd
     Legitimate_SAS
     Encryption
    Key
axioms
     @axm51: isolate_cmd ∈ OTHER_CORRECTIVE_CMD
    @axm52: partition (OTHER_CORRECTIVE_CMD, {isolate_cmd})
    @axm53: Security_Alert ∈ OTHER_MNTR_TYPE
    @axm54: {Legitimate_SAS} ⊆ SAS
    @axm55: Encryption ∈ COMMAND × KEY ↣ CYPHER_COMMAND
    @axm56: Key ∈ KEY
End
```

• The Machine of the Second Refinement M2

```
machine M2
refines M1
sees C2
```

```
variables
   valid_SAD
   registered_SAD
   unregistered_SAD
   retired_SAD
   standby_SAD
   channel_AdmRequest
    SAS_rcv_admRequest
   unoperational_SAD
   operational_SAD
   OTHER_UNOPERATIONAL_SAD
   Operate_executed_SAD
   AllCmd_executed_SAD
   rolling_blackout_SAD
   on_repair_SAD
   on_upgrade_SAD
   busy_SAD
   Release_executed_SAD
   fault
   channel_command //command Buffer btw SAS & SAD for sync
   channel_mntrInfo//monitoring info Buffer btw SAS & SAD for sync
   SAD_received_command
   SAS_received_mntrInfo
   previous_cmd
   compromised_SAD
   OTHER_UNOPERATIONAL_SAD2
   Blocked_AccessPoint
   counter
   ManualChecked_SAD
    encrypted_commands
   unencrypted_commands
   attacker_knowledge
invariants
   @inv81: compromised_SAD ⊆ OTHER_UNOPERATIONAL_SAD
    @inv82: OTHER_UNOPERATIONAL_SAD2 ⊆ OTHER_UNOPERATIONAL_SAD
   @inv83: partition (OTHER_UNOPERATIONAL_SAD, compromised_SAD, OTHER_UNOPERATIONAL_SAD2)
   @inv84: Blocked_AccessPoint ⊆ OTHER_AccessPoint
   @inv85: counter ∈ SAD → ℕ
   @inv86: ManualChecked_SAD ⊆ on_repair_SAD ∪ compromised_SAD
   @inv87: ∀ x· ((mntr_cntnt(x) = Security_Alert) ∧ (counter(mntrid(x))>3)) ⇒ mntrid(x) ∈ ManualChecked_SAD
   @inv88: ∀ x· ((mntrid(x) ∈ on_repair_SAD) ∧ (counter(mntrid(x))>3)) ⇒ mntrid(x) ∈ ManualChecked_SAD
   @inv89: ∀ x· ((mntrid(x) ∈ compromised_SAD) ∧ (counter(mntrid(x))>3)) ⇒ mntrid(x) ∈ ManualChecked_SAD
   @Abnormal_mntr: ∀ x· (x ∈ channel_mntrInfo ∧  mntr_cntnt(x) ∉  MNTR_TYPE ⇒ mntrid(x) ∈ compromised_SAD)
   @illegitimate_SAD: ∀ x· ((x ∈ channel_mntrInfo) ∧ (mntrid(x) ∉ valid_SAD) ∧ (mntr_cntnt(x) ∉ MNTR_TYPE))
⇒ mntrid(x) ∈ Blocked_AccessPoint
   @Abnormal_Cmd: ∀ x,y· ((x ∈ channel_mntrInfo) ∧ (y ∈ channel_command) ∧ (cmd_cntnt(y) ∉  CMD_TYPE)) ⇒
mntr_cntnt(x) = Security_Alert
   @inv90: encrypted_commands ∈ MESSAGE ⇸ CYPHER_COMMAND
   @inv91: unencrypted_commands ∈ MESSAGE ⇸ channel_command
   @inv92: attacker_knowledge ⊆ channel_command
   @inv93: channel_command ∩ attacker_knowledge = ∅    // Security property: CONFIDENTIALITY of Commands
   @inv94: dom(encrypted_commands) ∩ dom(unencrypted_commands) = ∅ // The messages have either encrypted
Commands or unencrypted Commands, it cannot have encrypted Commands and unencrypted Commands at same time
   @inv95: channel_command ∩ ran(unencrypted_commands) = ∅ //Channel is conatin no unencrypted Commands
events
   event INITIALISATION extends INITIALISATION
   then
      @act71: compromised_SAD ≔ ∅
      @act72: OTHER_UNOPERATIONAL_SAD2  ≔ ∅
      @act73: Blocked_AccessPoint ≔ ∅
      @act74: counter :∈ SAD → {0}
      @act75: ManualChecked_SAD ≔ ∅
      @act76: encrypted_commands ≔ ∅
      @act77: unencrypted_commands ≔ ∅
      @act78: attacker_knowledge ≔ ∅
   end

   event SAD_Send_Admission_Request extends SAD_Send_Admission_Request
   end


   event SAS_Receive_Admission_Request extends SAS_Receive_Admission_Request
   end
```

```
    event SAS_RegSAD_Success extends SAS_RegSAD_Success
    end

    event SAS_RegSAD_Fail extends SAS_RegSAD_Fail
    end

    event SAS_Detect_fault extends SAS_Detect_fault
    end

    event SAD_ExecuteSend_MntrInfo_OperateCmd extends SAD_ExecuteSend_MntrInfo_OperateCmd
    end

    event SAD_ExecuteSend_MntrInfo_OtherCmd extends SAD_ExecuteSend_MntrInfo_OtherCmd
    end

    event SAD_ExecuteSend_MntrInfo_ReleaseCmd extends SAD_ExecuteSend_MntrInfo_ReleaseCmd
    end


    event SAS_Receive_MntrInfo extends SAS_Receive_MntrInfo
    end

    event SAS_CheckMntrSend_Cmd_OperateCmd_Success extends SAS_CheckMntrSend_Cmd_OperateCmd_Success
    end

    event SAS_CheckMntrSend_Cmd_OP_Normal extends SAS_CheckMntrSend_Cmd_OP_Normal
    end

    event SAS_CheckMntrSend_Cmd_OP_Release extends SAS_CheckMntrSend_Cmd_OP_Release
    end

    event SAS_DeregisterSAD_OP_PermFailure extends SAS_DeregisterSAD_OP_PermFailure
    end

    event SAS_DeregisterSAD_Standby_PermFailure extends SAS_DeregisterSAD_Standby_PermFailure
    end

    event SAS_CheckMntrSend_Cmd_OP_Overheat extends SAS_CheckMntrSend_Cmd_OP_Overheat
    end

    event SAS_CheckMntrSend_Cmd_Rolling_PowerON_Success extends SAS_CheckMntrSend_Cmd_Rolling_PowerON_Success
    end

    event SAS_CheckMntrSend_Cmd_Rolling_PowerON_Fail extends SAS_CheckMntrSend_Cmd_Rolling_PowerON_Fail
    end

    event SAS_CheckMntrSend_Cmd_OP_Malfunction extends SAS_CheckMntrSend_Cmd_OP_Malfunction
    where
       @grd17: counter(sad) ≤ 3
    end

    event SAS_CheckMntrSend_Cmd_Repair_Troubleshooting_Success extends
SAS_CheckMntrSend_Cmd_Repair_Troubleshooting_Success
    end

    event SAS_CheckMntrSend_Cmd_OP_SwOutdated extends SAS_CheckMntrSend_Cmd_OP_SwOutdated
    end

    event SAS_CheckMntrSend_Cmd_Upgrade_Patch_Success extends SAS_CheckMntrSend_Cmd_Upgrade_Patch_Success
    end

    event SAS_CheckMntrSend_Cmd_Upgrade_Patch_Fail extends SAS_CheckMntrSend_Cmd_Upgrade_Patch_Fail
    end

    event SAS_CheckMntrSend_Cmd_OP_FnshPrevCmd extends SAS_CheckMntrSend_Cmd_OP_FnshPrevCmd
    end

    event SAS_CheckMntrSend_Cmd_Busy_FnshPrevCmd_Success extends
SAS_CheckMntrSend_Cmd_Busy_FnshPrevCmd_Success
    end
    event SAS_CheckMntrSend_Cmd_Busy_FnshPrevCmd_Fail extends SAS_CheckMntrSend_Cmd_Busy_FnshPrevCmd_Fail
    end
```

```
   event SAS_DeregisterSAD_Rolling_PermFailure extends SAS_DeregisterSAD_Rolling_PermFailure
   end

   event SAS_DeregisterSAD_Repair_PermFailure extends SAS_DeregisterSAD_Repair_PermFailure
   //then
    //@act14: ManualChecked_SAD ≔ ManualChecked_SAD \ {sad}
   where
      @grd17: sad ∉ ManualChecked_SAD
   end

   event SAS_DeregisterSAD_Upgrade_PermFailure extends SAS_DeregisterSAD_Upgrade_PermFailure
   end

   event SAS_DeregisterSAD_Busy_PermFailure extends SAS_DeregisterSAD_Busy_PermFailure
   end

   event SAS_CheckMntrSend_Cmd_OperateCmd_Fail_Overheat extends
SAS_CheckMntrSend_Cmd_OperateCmd_Fail_Overheat
   end

   event SAS_CheckMntrSend_Cmd_OperateCmd_Fail_Malfunction extends
SAS_CheckMntrSend_Cmd_OperateCmd_Fail_Malfunction
   where
      @grd14: counter(sad) ≤ 3
   end

   event SAS_CheckMntrSend_Cmd_OperateCmd_Fail_SwOutdated extends
SAS_CheckMntrSend_Cmd_OperateCmd_Fail_SwOutdated
   end

   event SAS_CheckMntrSend_Cmd_OperateCmd_Fail_FinishingPrevCmd extends
SAS_CheckMntrSend_Cmd_OperateCmd_Fail_FinishingPrevCmd
   end

   event SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Fail_Overheat extends
SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Fail_Overheat
   end

   event SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Fail_Malfunction extends
SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Fail_Malfunction
   where
      @grd15: counter(sad) ≤ 3
   end

   event SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Fail_SwOutdated extends
SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Fail_SwOutdated
   end

   event SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Fail_FinishingPrevCmd extends
SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Fail_FinishingPrevCmd
   end


   event SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Success extends SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Success
   end

   event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Success_Overheat extends
SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Success_Overheat
   end

   event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Success_Malfunction extends
SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Success_Malfunction
   where
      @grd17: sad ∉ ManualChecked_SAD
   end

   event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Success_SwOutdated extends
SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Success_SwOutdated
   end

   event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Success_FinishingPrevCmd extends
SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Success_FinishingPrevCmd
   end
   event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Fail_Overheat extends
SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Fail_Overheat
   end

   event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Fail_Malfunction extends
SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Fail_Malfunction
   end

   event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Fail_SwOutdated extends
SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Fail_SwOutdated
   end

   event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Fail_FinishingPrevCmd extends
SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Fail_FinishingPrevCmd
   end

   event SAD_Receive_Cmd_Success extends SAD_Receive_Cmd
   any
      sas
   where
      @grd3: sas ∈ {Legitimate_SAS}
   end

   event SAD_Receive_Cmd_Fail extends SAD_Receive_Cmd_Failure
   any
      sas
      sad
   where
      @grd1: sad ∈ registered_SAD
      @grd3: sas ∉ {Legitimate_SAS}
      @grd5: mntr_cntnt(mntr) = Security_Alert
      @grd6: mntrid(mntr) = sad
   end

   event SAS_CheckMntrSend_Cmd_OP_DetectAbnormal_Mntr extends SAS_CheckMntrSend_Cmd_OP_OtherFailures
   where
      @grd5: mntr_cntnt(mntr) ≠ Normal_info
      @grd6: mntr_cntnt(mntr) ∉ Transient_failure
      @grd7: mntr_cntnt(mntr) ∉ Permanent_failure
      @grd8: mntrid(mntr) = sad
      @grd9: cmd_cntnt(cmd) = corCmd
      @grd10: cmdid(cmd) = sad
      @grd11: sad ∉ Release_executed_SAD
      @grd12: sad ∈ AllCmd_executed_SAD
      @grd15: corCmd = Patching_Cmd
      @grd16: counter(sad) ≤ 3
   then
      @act6: compromised_SAD ≔  compromised_SAD ∪ {sad}
   end


   event SAS_CheckMntrSend_Cmd_compromised_Patching_Success extends
SAS_CheckMntrSend_Cmd_UNOP_Normal2_OtherUNOP
   where
      @grd12: sad ∈ compromised_SAD
      @grd13: sad ∉ ManualChecked_SAD
   then
       @act8: compromised_SAD ≔ compromised_SAD \ {sad}
   end

   event SAS_CheckMntrSend_Cmd_compromised_Patching_Fail_AbnormalMntr extends
SAS_CheckMntrSend_Cmd_Corrective_Fail
   where
      @grd7: mntr_cntnt(mntr) ≠ Normal_info
      @grd8: sad ∈ compromised_SAD
       @grd9: cmd_cntnt(cmd) = Patching_Cmd
       @grd10: counter (sad) < 3
       @grd11: sad ∈ AllCmd_executed_SAD
    then
      @act5: counter(sad) ≔  counter(sad) + 1
   end

   event SAS_CheckMntrSend_Cmd_compromised_Patching_Fail_AbnormalMntr_MaxReached extends
SAS_CheckMntrSend_Cmd_Corrective_Fail
```

```
   where
      @grd7: mntr_cntnt(mntr) ≠ Normal_info
      @grd8: sad ∈ compromised_SAD
       @grd9: cmd_cntnt(cmd) = Patching_Cmd
       @grd10: counter (sad) = 3
       @grd11: sad ∈ AllCmd_executed_SAD
    then
      @act5: counter(sad) ≔ 0
   end

   event SAS_CheckMntrSend_Cmd_compromised_Patching_Fail_AbnormalMntr_MaxExceeded extends
SAS_CheckMntrSend_Cmd_Corrective_Fail
   where
      @grd7: mntr_cntnt(mntr) ≠ Normal_info
      @grd8: sad ∈ compromised_SAD
       @grd9: cmd_cntnt(cmd) = isolate_cmd
       @grd10: counter (sad) > 3
       @grd11: sad ∈ AllCmd_executed_SAD
    then
       @act5: ManualChecked_SAD ≔ ManualChecked_SAD ∪ {sad}
      @act8: counter(sad) ≔ 0
   end


   event SAS_CheckMntrSend_Cmd_OperateCmd_Fail_Abnormal_Mntr extends
SAS_CheckMntrSend_Cmd_OperateCmd_OtherFailures
    where
      @grd6: mntr_cntnt(mntr) ∉ Transient_failure
      @grd7: mntr_cntnt(mntr) ∉ Permanent_failure
      @grd8: mntr_cntnt(mntr) ≠ Normal_info
      @grd9: mntrid(mntr) = sad
      @grd10: cmd_cntnt(cmd) = corCmd
      @grd11: cmdid(cmd) = sad
      @grd12: sad ∈ Operate_executed_SAD
      @grd13: corCmd = Patching_Cmd
      @grd14: counter (sad) ≤ 3
   then
      @act10: compromised_SAD ≔  compromised_SAD ∪ {sad}
   end

   event SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_Fail_Abnormal_Mntr extends
SAS_CheckMntrSend_Cmd_OP_ReleaseCmd_OtherFailures
   where
      @grd5: mntr_cntnt(mntr) ∉ Transient_failure
      @grd6: mntr_cntnt(mntr) ∉ Permanent_failure
      @grd7: mntr_cntnt(mntr) ≠ Normal_info
      @grd8: mntrid(mntr) = sad
      @grd9: cmdid(cmd) = sad
      @grd10: cmd_cntnt(cmd) = corCmd
      @grd11: sad ∈ Release_executed_SAD
      @grd12: corCmd = Patching_Cmd
      @grd13: counter (sad) ≤ 3
   then
        @act6: compromised_SAD ≔ compromised_SAD ∪ {sad}
   end

   event SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_Fail_Abnormal_Mntr extends
SAS_CheckMntrSend_Cmd_UNOP_ReleaseCmd_OtherFailures
   where
      @grd5: mntr_cntnt(mntr) ∉ Transient_failure
      @grd6: mntr_cntnt(mntr) ∉ Permanent_failure
      @grd7: mntr_cntnt(mntr) ≠ Normal_info
      @grd8: mntrid(mntr) = sad
      @grd9: cmdid(cmd) = sad
      @grd10: cmd_cntnt(cmd) = corCmd
      @grd11: sad ∈ Release_executed_SAD
      @grd12: sad ∈ compromised_SAD
      @grd13: corCmd = Patching_Cmd
   end

   event SAS_CheckMntrSend_Cmd_Repair_Troubleshooting_Fail_counter extends
SAS_CheckMntrSend_Cmd_Repair_Troubleshooting_Fail
   where
      @grd17: counter (sad) < 3
   then
      @act7: counter(sad) ≔  counter(sad) + 1
   end

   event SAS_CheckMntrSend_Cmd_Repair_Troubleshooting_Fail_MaxReached extends
SAS_CheckMntrSend_Cmd_Repair_Troubleshooting_Fail
   where
      @grd17: counter (sad) = 3
   then
      @act7: counter(sad) ≔ 0
   end

   event SAS_CheckMntrSend_Cmd_Repair_Troubleshooting_Fail_MaxExceeded extends
SAS_CheckMntrSend_Cmd_UNOP_Repair_OtherFailures
   where
      @grd5: mntr_cntnt(mntr) = Malfunction
      @grd7: sad ∈ on_repair_SAD
       @grd8: cmd_cntnt(cmd) = isolate_cmd
       @grd9: counter (sad) > 3
       @grd10: sad ∈ AllCmd_executed_SAD
       @grd11: sad ∉ ManualChecked_SAD
    then
       @act8: ManualChecked_SAD ≔ ManualChecked_SAD ∪ {sad}
      @act9: counter(sad) ≔ 0
   end

   event SAS_CheckMntrSend_Cmd_Upgrade_DetectAbnormal_Mntr extends
SAS_CheckMntrSend_Cmd_UNOP_Upgrade_OtherFailures
   where
      @grd5: mntr_cntnt(mntr) ≠ Normal_info
      @grd6: mntr_cntnt(mntr) ∉ Transient_failure
      @grd7: mntr_cntnt(mntr) ∉ Permanent_failure
      @grd8: mntrid(mntr) = sad
      @grd9: cmd_cntnt(cmd) = corCmd
      @grd10: cmdid(cmd) = sad
      @grd11: sad ∈ AllCmd_executed_SAD
      @grd12: corCmd = Patching_Cmd
      @grd13: counter (sad) ≤ 3
   then
      @act12: compromised_SAD ≔  compromised_SAD ∪ {sad}
   end

   event SAS_Detect_Attack extends SAS_CheckMntrSend_Cmd_Failure
   any
       sad
   where
      @grd3: sad ∉ registered_SAD ∨ sad ∉ valid_SAD
      @grd7: mntrid(mntr) = sad
      @grd8: sad ∉ Blocked_AccessPoint
      @grd9: sad ∈ OTHER_AccessPoint
   then
      @act4: Blocked_AccessPoint ≔  Blocked_AccessPoint ∪ {sad}
   end

   event SAD_ExecuteSend_SecurityAlert_AbnormalReleaseCmd extends SAD_ExecuteSend_OtherMntrType
   where
      @grd8: cmd_cntnt(cmd) = Release_Cmd
      @grd9: cmdid(cmd) = sad
      @grd10: mntr_cntnt(mntr) = Security_Alert
      @grd11: mntrid(mntr) = sad
      @grd12: sad ∈ standby_SAD
   end

   event SAD_ExecuteSend_SecurityAlert_AbnormalOperateCmd extends SAD_ExecuteSend_OtherMntrType
   where
      @grd8: cmd_cntnt(cmd) = Operate_Cmd
      @grd9: cmdid(cmd) = sad
      @grd10: mntr_cntnt(mntr) = Security_Alert
      @grd11: mntrid(mntr) = sad
      @grd12: sad ∈ operational_SAD
   end


   event SAD_ExecuteSend_SecurityAlert_DetectAbnormal_Cmd extends SAD_ExecuteSend_OtherMntrType
```

```
where
   @grd8: cmd_cntnt(cmd) ≠ Normal_Cmd
   @grd9: cmd_cntnt(cmd) ≠ Operate_Cmd
   @grd10: cmd_cntnt(cmd) ≠ Release_Cmd
   @grd11: cmd_cntnt(cmd) ∉ Corrective_Cmd
   @grd12: cmdid(cmd) = sad
   @grd13: mntr_cntnt(mntr) = Security_Alert
   @grd14: mntrid(mntr) = sad
end

event SAS_CheckMntrSend_Cmd_SecurityAlert extends SAS_CheckMntrSend_Cmd_Failure
any
   sad sas
where
   @grd5: sad ∈ registered_SAD
   @grd6: sas ∈ OTHER_AccessPoint
   @grd7: mntr_cntnt(mntr) = Security_Alert
   @grd9: mntrid(mntr) = sad
   @grd10: cmd ∈ previous_cmd
   @grd11: counter (sad) < 3
then
    @act4: counter(sad) ≔  counter(sad) + 1
    @act5: Blocked_AccessPoint ≔  Blocked_AccessPoint ∪ {sas}
end

event SAS_CheckMntrSend_Cmd_SecurityAlert_MaxReached extends SAS_CheckMntrSend_Cmd_Failure
any
   sad sas
where
   @grd5: sad ∈ registered_SAD
   @grd6:sas ∈  OTHER_AccessPoint
   @grd7: mntr_cntnt(mntr) = Security_Alert
   @grd9: mntrid(mntr) = sad
   @grd10: cmd ∈ previous_cmd
   @grd11: counter (sad) = 3
then
    @act4: counter(sad) ≔ 0
    @act5: Blocked_AccessPoint ≔  Blocked_AccessPoint ∪ {sas}
end

event SAS_CheckMntrSend_Cmd_SecurityAlert_MaxExceeded extends SAS_CheckMntrSend_Cmd_Failure
any
   sad
where
   @grd5: sad ∈ registered_SAD
   @grd6: mntr_cntnt(mntr) = Security_Alert
   @grd9: mntrid(mntr) = sad
    @grd10: counter (sad) > 3
    @grd11: cmd_cntnt(cmd) = isolate_cmd
    @grd12: sad ∈ AllCmd_executed_SAD
 then
    @act4: ManualChecked_SAD ≔ ManualChecked_SAD ∪ {sad}
   @act6: counter(sad) ≔ 0
end

event SAS_Send_OperateCmd extends SAS_Send_OperateCmd
any
m
where
   @grd8: m ∉ dom(encrypted_commands)
   @grd9: m ∉ dom(unencrypted_commands)
then
@act3: encrypted_commands(m) ≔ Encryption(cmd ↦ Key)
end

event SAS_Send_OperateCmd_Unencrypted extends SAS_Send_OperateCmd
any
m
where
   @grd8: m ∉ dom(encrypted_commands)
   @grd9: m ∉ dom(unencrypted_commands)
then
@act3: unencrypted_commands(m) ≔ cmd
end
```

```
   event SAD_Receive_EncryptedCmd extends SAD_Receive_Cmd // Checking when receiving the Command if it is
encrypted, it will be d
   any
      m
   where
      @grd3: m ∈ dom(encrypted_commands)
      @grd4: encrypted_commands(m) = Encryption(cmd ↦ Key) // Command is the decryption of the Encrypted
Command x Key
   end

   event SAD_Receive_Unencrypted // Checking when receiving the Command if it is unencrypted, it will be
discarded
   any m sad cmd
   where
      @grd1: sad ∈ registered_SAD
      @grd2: cmd ∈ channel_command
      @grd3: m ∈ dom(unencrypted_commands)
      @grd4: unencrypted_commands(m) = cmd
   then
      @act1: channel_command ≔ channel_command \ {cmd}
   end

   event ATK_Eavesdrop
   any m
   where
      @grd1: m ∈ dom(unencrypted_commands)
   then
      @act1: attacker_knowledge ≔ attacker_knowledge ∪ {unencrypted_commands(m)}
   end


   event ATK_Tamper
   any m cmd
   where
      @grd1: m ∈ dom(unencrypted_commands)
      @grd2: cmd ∈ COMMAND
   then
      @act1: unencrypted_commands(m) ≔ cmd
   end

end
```

Case Study: Analysis of the Interaction Between AMI and Smart Meters

As stated before, this research models the functional requirements of the key access points of the IoT-enabled SG, which are SAS and AMI.

- The Advanced Metering Infrastructure (AMI); since AMI provides the technology to allow the exchange of information between Smart Meters and the Utility. It collects and analyses the electricity consumption by the consumers. It records and forwards the information including meter readings from Smart Meters to the Meter Data Management Systems (MDMS), and then to the Utility via ICT networks.
- the research analyses the interaction between AMI and Smart Meters as another part of the IoT-enabled Smart Grid to find out whether the resulting Formal Model of the interaction between SAS and SAD could be generalised as a template model for the interaction between AMI and Smart Meters or any other access points.

As presented in the system scenario in section 6.2.1, the Distribution substations distribute low-voltage to the Smart Meters (SM) through the Smart Metering network. Distribution substations distribute the electricity to a multitude of Smart Meters (in residential, commercial, and industrial premises). Multiple Smart Meters are connected to a single Utility, and a single Utility communicates with multiple Smart Meters. Also, Multiple Smart Meters are connected to a single Advanced Metering Infrastructure (AMI), and a single AMI communicates with multiple Smart Meters. For more explanation, The AMI provides the technology that allows the exchange of information between Smart Meters and the Utility, recording and forwarding the information from Smart Meters to the Meter Data Management System (MDMS) that manages information processing such as (Meter configuration information, Periodic meter readings, and On-demand meter readings) (NISTIR 7628, 2014).

The commissioning and registration processes are discussed before in Asset Management for the IoT-SAD devices and SAS. In the case of Smart Meters, enrolment is an additional process to the commissioning and registration processes in which the Advanced Metering Infrastructure (AMI) assigns a payment plan to the Smart Meter. As reported in (NISTIR 7628, 2014), There are two well-known pricing programs that Smart Metes can be enrolled in as a payment plan: the first is the (RTP) program that is (Real Time Pricing), and the second program is the (TOU) program that is (Time Of Use). The enrolment process will be described below and is represented in FUN-16 in Figure 6-6.

As explained before in Asset Management, the Utility manages a list of valid manufacturing ID numbers for the Smart Meters, and the Utility updates the AMI about the valid list of those Smart Meters. The commission process will be initiated when the “installer” powers on the Smart Meter and follows the manufacturer’s instructions. Thus, the valid installed Smart Meter that is powered “ON” sends admission requests to the AMI to join the network, as shown in FUN-14 in Figure D-2. Then, AMI registers the valid installed Smart Meter (FUN-15, Figure 6-6) involving authentication and authorization processes. The commissioning-including the admission request- and the registration processes are followed by the enrolment process (FUN-16, Figure 6-6) where the AMI enrols/disenrolls the Smart Meter to a payment plan that represents the pricing program assigned as discussed before in the Asset Management either it is the RTP program or the TOU program. Once the Smart Meter is registered, it is regulated and it will be assigned to the standby state until the AMI enrols the Smart Meter. Then, the Utility sends (Operate) Command to the Standby Smart Meter (FUN-17, Figure 6-6). Smart Meter executes (Operate) Command (FUN-18, Figure 6-6), and If the execution failed, the standby SM will be an unoperational SM. If the execution succeeded, the standby SM will be converted to operational SM, and can begin to exchange information with the AMI and the Utility, and can begin to send Monitoring Information, see FUN-19, in Figure 6-6. The Monitoring Information here could be:

1- The status of SM devices themselves whether it is a failure or Normal information. The Normal Monitoring Information means either Normal status for the SM or Consumption Information.
2- The execution responses for the received Commands such as (Success/Fail).
3- Consumption Information which is the meter readings per time.

AMI receives that Monitoring Information (FUN-20, in Figure 6-6) and then checks the monitoring information (FUN-23, in Figure 6-6) to decide the next appropriate state of the SM. There are three different cases when AMI receives the Monitoring Information from the operational SM:

1- If the received Monitoring Information is Normal, then AMI sends the Command to the operational SM, and the SM is still in the operational state (FUN-23.1, Figure 6-6).
2- If the received Monitoring Information is a Transient failure, then AMI converts SM to unoperational SM and sends back the appropriate Command to the SM as a response to the failure (FUN-23.2, Figure 6-6).
3- If the received Monitoring Information is a Permanent failure, then AMI deregisters SM, and the operational SM will be retired SM (FUN-23.3, Figure 6-6).

As stated in the case of Normal Monitoring Information received such as Consumption Information, AMI forwards it to the Utility, which in turn, sends Normal Command which is in this case the billing information based on the received Generation Pricing Information from the Market (FUN-22, Figure 6-6) (U.S. Department of energy, 2012; Marcelo *et al.*, 2013; NISTIR 7628, 2014). The Market varies significantly from state to state, region to region. There are no direct operational impacts, but there are financial security impacts interm of pricing information (NISTIR 7628, 2014).

The Smart Meter equipped with the In-Home Displays (IHDs) to display the billing information and the pricing signals to support the end-consumer to reduce rationalise consumption in peak times. This reduction will support The Utility, Generation Plants, Transmission, and Distribution in balancing supply and demand by rationalising production or diverting the power in response to Demand and Load (Marcelo *et al.*, 2013; NISTIR 7628, 2014). The Utility calculates the total Load Profile for all consumption information of all Smart Meters and sends the Load Profile Data to SAS in the Generation Plant for balancing demand and supply (FUN-31, Figure 6-6). Although the Smart Meters are considered a separate component from the end-user interfaces (In-Home Displays) in the system scenario and functionality use cases published by NIST, this research includes the end-user interfaces and applications combined in the Smart Meters.

So, Normal Command is the command required to run the operations of the IoT-enabled SG. Whereas, in case of transient failure, AMI converts SM to an unoperational state and will send a response command such as upgrading Commands, and troubleshooting including configuration or calibration (Marcelo *et al.*, 2013; NISTIR 7628, 2014). In other meaning, the appropriate Command will be sent back to the SM.

If the SM is in an unoperational state, it still sends Monitoring Information (FUN-25, Figure 6-6). Then, AMI receives Monitoring Information from unoperational SM (FUN-26 in Figure 6-6), and checks that Monitoring Information (FUN-27 in Figure 6-6). There are three different cases when AMI receives the Monitoring Information from the unoperational SM:

1- If the received Monitoring Information is Normal, then AMI converts the state of the device to either an operational state or, in some cases, to a standby state depending on the failure itself (FUN-27.1, FUN-27.2, Figure D-2).
2- If the received Monitoring Information is a Transient failure, then the uoperational SM still be in the unoperational state (FUN-27.3, Figure D-2).
3- If the received Monitoring Information is a Permanent failure, then AMI deregisters SM, and the unoperational SM will be retired SM (FUN-27.4, Figure D-2).

According to the SEC 40-SDMS-02 and IEC 62056 standards (DLMS User Association, 2019; Saudi Electricity Company, 2019), ***Expert#8 and Expert#10***, the following are the most possible conditions of Transient failure:

1- SM is in The rolling blackout status due to power off in response to overheating failure or payment failure.
2- SM is in repair due to malfunction.
3- SM is upgrading the firmware.
4- SM is finishing a previous Command.
5- SM is in a locked state due to exceeding a specific number of consecutive invalid login attempts, Unauthorised access, Elevation of privilege, or wrong/corrupted encryption key.

6- Connectivity issues that make the AMI can not communicate with the SM devices when:

- The communication modem becomes faulty.
- The semi-card of the SM, the communication service provider, or the communication network (such as Ethernet, Fiber Optics, or Wireless) has issues with connection signals.
- Misconfiguration screwups the modem which disconnects the modem from the network.

Then, the operational SM executes the Command successfully if it is permitted. If the Command is not permitted, the SM will not execute the Command and the execution result will fail, as shown in FUN-24 in Figure 6-6 (NISTIR 7628, 2014). The most possible conditions where the Command is not permitted to be executed by Smart Meter are the same as the as the conditions of the Transient failure aforementioned, as reported by ***Expert#8, Expert#10,*** the SEC 40-SDMS-02 standards, and IEC 62056 standard (DLMS User Association, 2019; Saudi Electricity Company, 2019).

As presented in FUN-23.3 and FUN-27.4, Figure 6-6, the AMI manages the de-registration process to remove the SM from the registered and valid SM and convert it to a retired SM, when its lifespan is expired or in case of permanent failure including Memory failure or in case the troubleshooting process is failed. Also, AMI tracks the state of all Smart Meters (registered, retired).

Conclusion of AMI and Smart Meters Case Study

In the previous section, the analysis of the AMI functionality and system scenario shows that the functional requirements for SAS and IoT-SAD apply as well to the AMI and Smart Meters access points; because both IoT-SAD devices and Smart Meters are IoT devices have the same IoT nature. Therefore, the Formal Functional Model for SAS and IoT-SAD can be generalised to the AMI and Smart Meters as a template model in which SAS/AMI are the controller and IoT-SAD/Smart Meters are the IoT devices. So, The modelling for the (SAS and IoT-SAD) part that is presented in black colour in Figure D-2 is applied to the modelling of the (AMI and Smart Meters) part that is presented in blue colour in Figure D-2.

Given that AMI is the second key access point from the security viewpoint. As a result, the Formal Functional Model developed for SAS and SAD also represents the Formal Functional Model for the whole IoT-enabled Smart Grid in the level of abstraction that followed in this research under the same modelling aspects stated in Chapter 6.

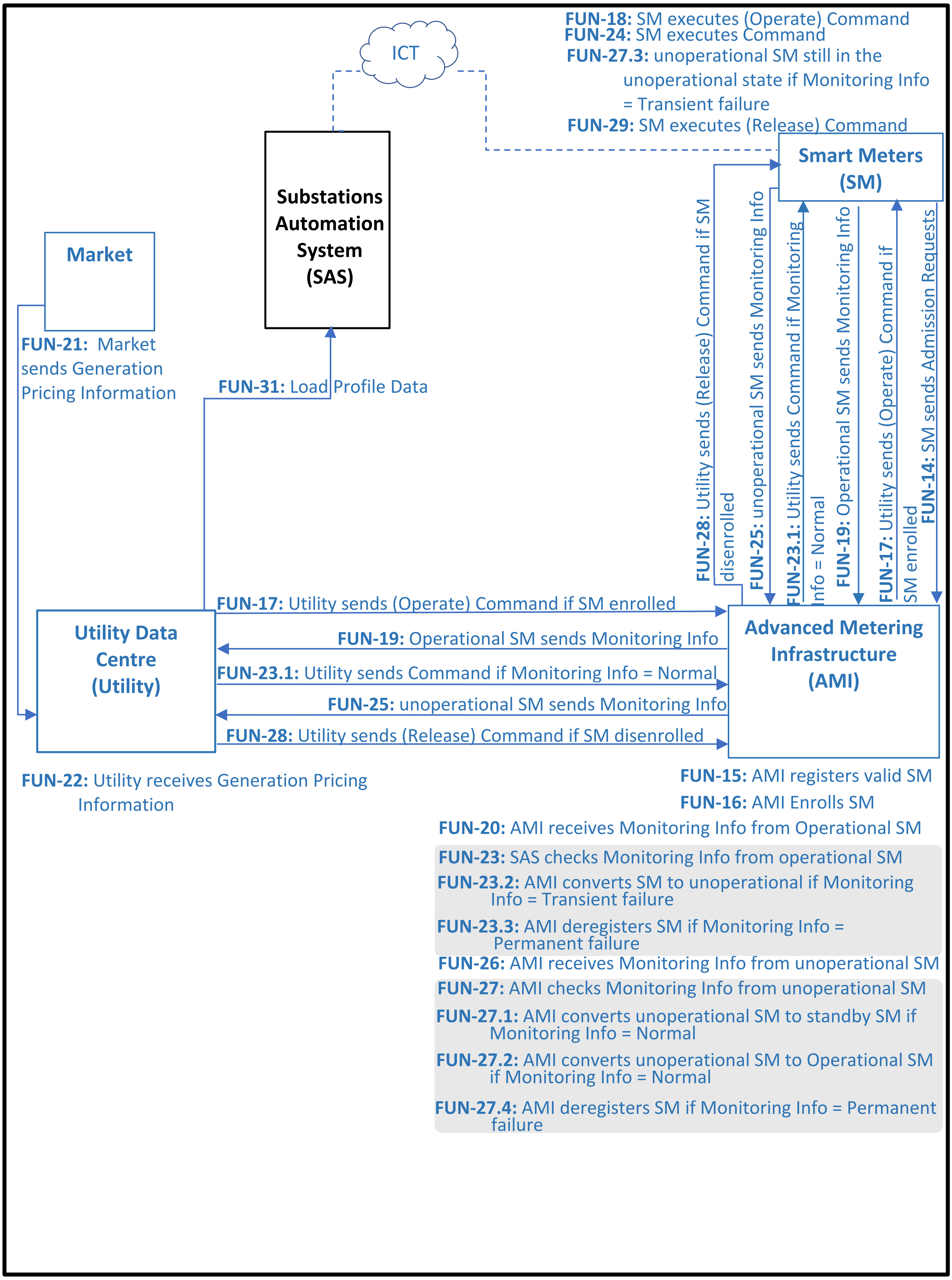

ICT
FUN-18: SM executes (Operate) Command
FUN-24: SM executes Command
FUN-27.3: unoperational SM still in the unoperational state if Monitoring Info = Transient failure
FUN-29: SM executes (Release) Command
Smart Meters (SM)
Substations Automation System (SAS)
Market
FUN-21: Market sends Generation Pricing Information
FUN-31: Load Profile Data
FUN-28: Utility sends (Release) Command if SM disenrolled
FUN-25: unoperational SM sends Monitoring Info
FUN-23.1: Utility sends Command if Monitoring Info = Normal
FUN-19: Operational SM sends Monitoring Info
FUN-17: Utility sends (Operate) Command if SM enrolled
FUN-14: SM sends Admission Requests
FUN-17: Utility sends (Operate) Command if SM enrolled
FUN-19: Operational SM sends Monitoring Info
FUN-23.1: Utility sends Command if Monitoring Info = Normal
FUN-25: unoperational SM sends Monitoring Info
FUN-28: Utility sends (Release) Command if SM disenrolled
Utility Data Centre (Utility)
Advanced Metering Infrastructure (AMI)
FUN-22: Utility receives Generation Pricing Information
FUN-15: AMI registers valid SM
FUN-16: AMI Enrolls SM
FUN-20: AMI receives Monitoring Info from Operational SM
FUN-23: SAS checks Monitoring Info from operational SM
FUN-23.2: AMI converts SM to unoperational if Monitoring Info = Transient failure
FUN-23.3: AMI deregisters SM if Monitoring Info = Permanent failure
FUN-26: AMI receives Monitoring Info from unoperational SM
FUN-27: AMI checks Monitoring Info from unoperational SM
FUN-27.1: AMI converts unoperational SM to standby SM if Monitoring Info = Normal
FUN-27.2: AMI converts unoperational SM to Operational SM if Monitoring Info = Normal
FUN-27.4: AMI deregisters SM if Monitoring Info = Permanent failure

Figure D-0-2: System block diagram for AMI and SM, showing access points and information flow

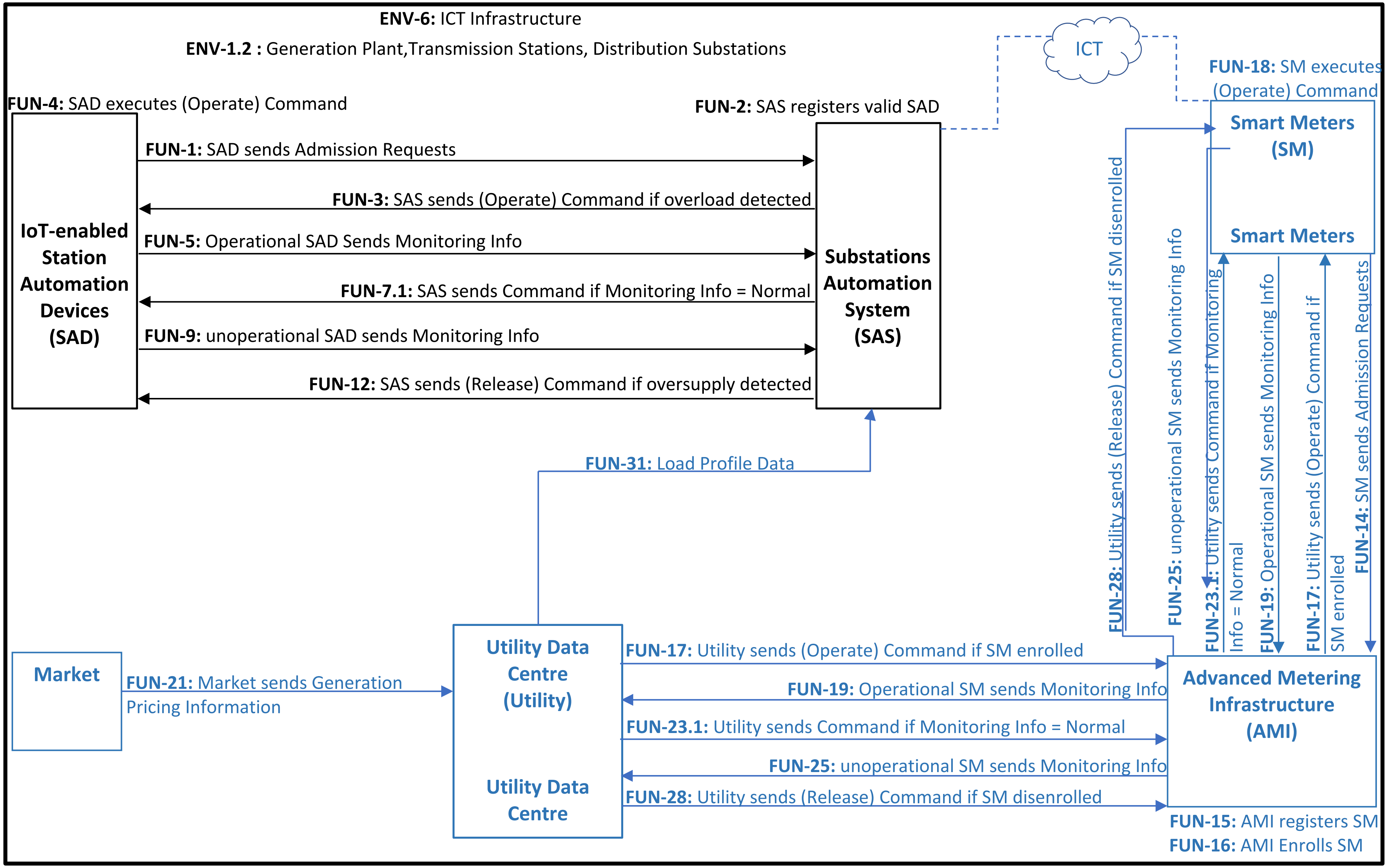

ENV-6: ICT Infrastructure
ENV-1.2 : Generation Plant,Transmission Stations, Distribution Substations
ICT
FUN-18: SM executes (Operate) Command
FUN-4: SAD executes (Operate) Command
FUN-2: SAS registers valid SAD
Smart Meters (SM)
Smart Meters
IoT-enabled Station Automation Devices (SAD)
FUN-1: SAD sends Admission Requests
FUN-3: SAS sends (Operate) Command if overload detected
FUN-5: Operational SAD Sends Monitoring Info
FUN-7.1: SAS sends Command if Monitoring Info = Normal
FUN-9: unoperational SAD sends Monitoring Info
FUN-12: SAS sends (Release) Command if oversupply detected
Substations Automation System (SAS)
FUN-28: Utility sends (Release) Command if SM disenrolled
FUN-25: unoperational SM sends Monitoring Info
FUN-23.1: Utility sends Command if Monitoring Info = Normal
FUN-19: Operational SM sends Monitoring Info
FUN-17: Utility sends (Operate) Command if SM enrolled
FUN-14: SM sends Admission Requests
FUN-31: Load Profile Data
Market
FUN-21: Market sends Generation Pricing Information
Utility Data Centre (Utility)
Utility Data Centre
FUN-17: Utility sends (Operate) Command if SM enrolled
FUN-19: Operational SM sends Monitoring Info
FUN-23.1: Utility sends Command if Monitoring Info = Normal
FUN-25: unoperational SM sends Monitoring Info
FUN-28: Utility sends (Release) Command if SM disenrolled
Advanced Metering Infrastructure (AMI)
FUN-15: AMI registers SM
FUN-16: AMI Enrolls SM

Figure 0-3: System block diagram for an IoT-enabled Smart Grid, showing access points and information flow

Advanced Security Controls with “Should” and “Could” Categories

Table D-1: Security controls with “Should” classification

Security Control	Status	Internet-based Security Threats from the STRIDE Analysis	Security Requirement	Code
The system has to have a secure session management for multiple users’ sessions for each request.	S	Spoofing	Authentication Confidentiality	Aun7
The system has to have an Anti-Spoofing algorithm	S	Eavesdropping/ Traffic Analysis/ Man In The Middle (MITM)		Aun8
Privileged Access Management (PAM)	S	Spoofing	Authorisation	Aur6
Principle of Least Privilege (POLP)	S			Aur7
Digital watermarking	S	Data Tampering Replay Attack SQL injection Malware injection False data injection	Integrity	In2
Secure Phasor Measurement Units (PMUs) installation	S			In4
Load profiling algorithms	S			In5
Intrusion Prevention Systems (IPS)	S	Denial Of Service	Availability	Av4
Web Application Firewall (WAF)	S			Av9
Anti-DDOS algorithm	S			Av10

Table D-2: Security controls with "Could" classification

Security Control	Status	Internet-Based Security Threats from the STRIDE Analysis	Security Requirement	Code
1. The access point has to have the Physically Unclonable Functions (PUF) which are lightweight hardware-based authentication	C	Spoofing Eavesdropping / Traffic Analysis/ Man In The Middle (MITM)	Authentication	Aun2
2. The system has to hash the information using the MAC-attached cryptographic function or the HORS signature.	C			Aun3
3. The System has to encrypt the data using the Attribute Certificates	C	Spoofing	Authorisation	Aur2
4. The System has to encrypt the data using the Attribute-Based Access Control System based on XACML (Extensible Access Control Markup Language)	C			Aur3
5. Anonymisation	C	Eavesdropping / Traffic Analysis/ Man In The Middle (MITM	Privacy Confidentiality	P1
6. Trusted aggregator	C			P2
7. Adaptive cumulative sum algorithm	C	Data Tampering Replay Attack SQL injection Malware injection False data injection	integrity	In3
8. Sequence numbers	C			In7
9. Query sanitisation	C			In8
10. Nonces	C			In9
11. Use multiple alternate frequency channels according to a hardcoded sequence	C	Denial Of Service Malware injection	Availability	Av1
12. Specification-based IDS	C			Av3
13. Quality of Services (QoS)	C			Av5
14. Operating system-independent Applications	C			Av7
15. Mutual Inspection technique	C	False data injection	Non-Repudiation	N1
16. Endpoint for Detection and Response EDR	C	Malware injection	all the above security requirements	Common 3

Transcripts for the Expert's Reviews

Full transcripts of the Expert's reviews are available in the Dataset on https://ePrint.soton.ac.uk with DOI: https://doi.org/10.5258/SOTON/D3060. These transcripts are encoded and comply with the Ethical Approval policy.

Appendix E Description of Security Controls and Justifications

This section reviews the security controls from the literature that could mitigate against the main internet-based threats (Reyzin and Reyzin, 2002; Farrell and Housley, 2002; Gegick and Barnum, 2005; Shirey, 2007; Sharma *et al.*, 2008; CISA, 2009; Mui and Frankl, 2010; Aravinthan *et al.*, 2011; Hutchins *et al.*, 2011; Jeon, 2011; Jung *et al.*, 2011; Lee *et al.*, 2011; Benoit, 2011; Nabeel *et al.*, 2012; Bhattarai *et al.*, 2012; Giani *et al.*, 2013; Xiao *et al.*, 2013; Komninos *et al.*, 2014; NISTIR 7628, 2014; Yu *et al.*, 2014; McCary and Xiao, 2015; Stouffer *et al.*, 2015; Machaka *et al.*, 2016; Tazi *et al.*, 2016; Mrabet *et al.*, 2018; Cawthra *et al.*, 2018; Barker *et al.*, 2019; Kimani *et al.*, 2019; Das and Zeadally, 2019; Ganguly *et al.*, 2019; Díaz Redondo *et al.*, 2020):

Keyed Cryptographic Hash Functions, Digital Signatures, and Random Number Generators

This technique is also called hash-based authentication codes. It is similar to cryptographic hash functions, but the keyed cryptographic hash function uses a private secret key, which is a Hash Message Authentication Code (HMAC). This uses both a hash function and a shared private key (Benoit, 2011). In HMAC, each communicated entity has its own public-private key pair and has agreed on a common hash. The sender hashes the message with the agreed hash function, such as SHA-1, and sends it with the plain message to the receiver accompanied by the HMAC code. On the receiver side, the hash of the received message is obtained and then compared with the received hash. If the two hashes match, the file has not been tampered with. The communicated entities can verify whether the message and HMAC they receive were sent from an authentic legitimate source. This can be done by using the private key, shared and known by only the sender and receiver. So, both the sender and receiver can compute the HMAC using the shared private key. Then if the results are the same the authentication is proven, because no one other than the sender could sign the message with their private key. The digital signature is similar to keyed cryptographic hash functions, however, digital signatures use asymmetric encryption. Random number generators are used in key generation for authentication purposes in cryptography, digital signature, and encryption schemes (CISA, 2009).

Physically Unclonable Function (PUF) Modules

PUF modules are functions embodied in a physical structure. They are inexpensive to manufacture yet impossible to replicate or clone, even if given the exact manufacturing process. A PUF is a circuit that uses manufacturing process variations to generate a unique digital fingerprint. PUFs can be used in meters or any other IoT devices and applications to achieve both strong hardware-based authentication and lightweight authentication, which is a low-cost security scheme (Suh and Devadas, 2007; Nabeel *et al.*, 2012).

MAC-Attached, and HORS-Signed Messages

A Message Authentication Code (MAC) algorithm defines a family of cryptographic functions that are parameterised by a symmetric key. Each of the functions can act on input data (known as a "message") of variable length to produce an output value of a specified length. The output value is called the MAC of the input message. An approved MAC algorithm is expected to satisfy the following property (for each of its supported security levels): it must be computationally infeasible to determine the (as yet unseen) MAC of a message without knowledge of the key, even if one has already seen the results of using that key to compute the MACs of other (different) messages.

A MAC algorithm can be used to provide data origin authentication and data integrity protection. In this recommendation, a MAC algorithm is used for key confirmation; the use of MAC algorithms for key derivation is addressed in SP 800-56C (Barker *et al.*, 2019).

HORS (Hash to Obtain Random Subset) is a hashed-based one-time signature proposed by Reyzin and Reyzin (2002). HORS improves the time overhead necessary for verifying and signing, and reduces the key and signature sizes. The security of HORS relies on one-way functions and the subset resilience of the hash function.

Secure Sockets Layer Certificates (SSL) and Transport Layer Security (TLS)

Both are standard technologies that exchange digital certificates to authenticate data transmitted between servers, systems, applications, and end-users. TLS is the updated and improved version of SSL (Thomas, 2000).

Automatic Lockouts

The Smart Grid information system enforces a limit on the number of consecutive invalid login attempts by a user in a given time. To avoid denial of service, automatic lockouts are released after a predetermined time. Both this time and number of tries are organisation-defined (NISTIR 7628, 2014).

Attribute-Based Encryption

This technique is a type of public-key encryption that provides access rights based on user, environment, action, or resource attributes. These attributes could be any information related to transmitted data such as the source of energy (solar, wind, fossil fuel), the type of consumer (individual, company, vehicle), and the access time. The decryption of a ciphertext is possible only if the set of attributes of the user key matches the attributes of the ciphertext (Yu *et al.*, 2014).

Attribute Certificates

An attribute certificate is a digital document containing attributes associated with the holder by an authority/issuer (Farrell and Housley, 2002).

Attribute-Based Access Control System Based on XACML

This is an open standard XML-based language that defines the security policies and access rights to information. XACML (Extensible Access Control Markup Language) is an Attribute-Based Access Control system, where attributes (set of characteristics) associated with a user, environment, action, or resource are given as inputs to the function that decides whether a user may access a given resource in a particular way (Jung *et al.*, 2011).

Role-Based Access Control and Allow/Block Listing

This is a cybersecurity strategy for providing access to resources or data on the basis of user roles, defining the allow/block lists by the administrator to allow or block identified entities access to a particular privilege and service (Komninos *et al.*, 2014; Stouffer *et al.*, 2015).

Symmetric and Asymmetric Algorithms and Public Key Infrastructure Certificate (PKI)

Both algorithms are cryptographic techniques. The symmetric algorithm is private key cryptography in which the communicated entities share a secret key to encrypt and decrypt messages, such as the Triple Data Encryption Standard (TDES) and Advanced Encryption Standard (AES). The asymmetric algorithm or Public Key Infrastructure certificate (PKI) uses a pair of keys. One is a public one that is known to all other entities and used by the sender to encrypt the message, and one is a private key that is kept secret and used to decrypt the message by the receiver. Rivest, Shamir, and Adleman (RSA), Digital Signature Algorithm (DSA), Elliptic Curve Digital Signature Algorithm (ECDSA), and Elliptic Curve Cryptography (ECC) are examples of asymmetric algorithms (Lozupone, 2018).

Anonymisation

A technique to remove the link between data and its producer, in which the energy utility can perform the required computations without attributing that data to a specific smart meter (Xu *et al.*, 2006; Komninos *et al.*, 2014).

Trusted Aggregators

Trusted entities aggregate the metering data and forward it to the energy utility, which can process the aggregated data without accessing the consumption information of each participant meter (Silva *et al.*, 2017).

Homomorphic Encryption

An encryption technique that performs computations on encrypted data and produces an encrypted output that could be decrypted as if the computations were processed on the unencrypted original data (Ogburn *et al.*, 2013).

Perturbation Models

A security technique that adds noise to sensitive private metering data before transmitting it to the energy utility, which receives the perturbed data and rebuilds the approximate data of the origin (Wang *et al.*, 2020).

Verifiable Computation Models

A security verification technique in which the smart meter is the prover that proves its identity to the energy utility (the verifier), using secret information (proof) without disclosing that secret information to the verifier. This is called a zero-knowledge proof. Then, the verifier confirms that the prover has the knowledge that they claim to possess (Komninos *et al.*, 2014; Paverd *et al.*, 2015).

Data obfuscation Techniques

Obfuscation or masking techniques aim to hide original data classified as sensitive personal data or sensitive commercial data, such as energy consumption, hidden by buffering the energy load (Komninos *et al.*, 2014; Tonyali *et al.*, 2016).

Cryptographic Hashing Functions and Session Keys

A hash function is applied by the sender to check the original data and attach it to that data. Likewise, the receiver uses the same hash function as the original data and compares the resulting hash to that attached to the received data. In the case of matching hashes, integrity is ensured to prove that the data have not been modified or tampered with (Kessler, 2019). Common hash functions are Cipher, SHA1 (secure hash algorithm) with 160-bit output, MD5 (message digest) with 128-bit output, and SHA256/SHA512 (secure hash algorithm) with 256/512-bit output. A session key is an encryption and decryption key that is randomly generated to ensure the security of a communication session between two entities. Session keys are sometimes called symmetric keys, because the same key is used for both encryption and decryption (Kessler, 2019).

Watermarking

An efficient technique that embeds digital data (a watermark) inside that original data to validate data integrity at a low cost against false data injection attacks. For instance, unique information about the consumer is a watermark embedded inside the meter readings that are to be sent from the meter to the energy utility via the networks. Then, the utility receives both the watermark and the watermarked data to correlate them and detect any attack (Bhattarai *et al.*, 2012).

Adaptive Cumulative Sum Algorithm

CUSUM is an algorithm that is used in many fields for the detection of abrupt variations. It is a sequential analysis technique used for monitoring change (Machaka *et al.*, 2016).

Secure Phasor Measurement Units (PMUs) Installation

Installing secure PMUs is a technique that can be used to defend against SG attacks. The measurements that are reported regularly from various phasor-measurement units at several locations about the current status of the SG could reveal any abnormalities and trigger immediate action to protect it (Giani *et al.*, 2013).

Load Profiling Algorithm

This algorithm maintains permission from the AMI server for a specific device. The AMI server, in turn, sends a previously created load profile of that device to the controller to make the comparison. In the event of a mismatch between the current profile and the previous profile, the controller disables this device (Aravinthan *et al.*, 2011).

Timestamps

This technique keeps track of the creation and modification time of a transaction, without the ability to alter it once it has been recorded (Farha *et al.*, 2020).

Sequence Numbers

This technique binds each message to a sequential serial number to identify the duplicated and replayed messages. If the sequence order is incorrect, this indicates that the data are compromised by an attack (Yan *et al.*, 2012).

Query Sanitisation

This is a cybersecurity measure of checking, cleaning, and filtering data inputs from users, APIs, and web services (Mui and Frankl, 2010).

Nonces

A security technique that adds unique or non-repeating numbers to make each exchanged message unique (Xiao *et al.*, 2013).

Frequency-Hopping Spread-Spectrum

This uses multiple alternate frequency channels according to a hardcoded sequence, and is a method for transmitting radio signals by rapidly changing the carrier frequency between many distinct frequencies over a large spectral band. The changes are controlled by a code known to both communicated entities (Aravinthan *et al.*, 2011).

Frequency Quorum Rendezvous Between Connected Nodes

This is a scheme for a fast and resilient key establishment that allows each node to build a hopping sequence independently during the key establishment phase. This scheme increases the probability that a pair of nodes meet within a bounded segment of time, during which they share a common key for future communication (Lee *et al.*, 2011).

Anomaly Intrusion Detection Systems (IDS)

An anomaly-based IDS method that recognises deviations from normality by building a model of normal system behaviour, whereby any deviation from normal is identified as an intrusion (Scarfone and Mell, 2011; Santos *et al.*, 2018).

Specification-Based IDS

Specification-based IDS detects attacks using a set of constraints (rules) defining the correct operation of a program or protocol (Scarfone and Mell, 2011; Santos *et al.*, 2018).

Intrusion Prevention System (IPS)

This form of network security works to prevent identified threats. It monitors the network to prevent threats (Scarfone and Mell, 2011).

Quality of Service (QoS)

This is a set of technologies that work on a network to guarantee its ability to run high-priority applications and traffic under limited network capacity. QoS technologies accomplish this by providing differentiated handling and capacity allocation to specific flows in network traffic (Jeon, 2011).

Load Balancing

This is the process of distributing a set of tasks or traffic efficiently over a set of resources. Load balancing can support availability (Sharma *et al.*, 2008).

Air Gap

This is a network security measure employed to ensure that a secure network is physically isolated from unsecured networks, such as the public internet (Shirey, 2007).

Data DIODE

This is referred to as a unidirectional gateway, a deterministic one-way boundary device, or a unidirectional network. It is a network appliance or device that allows data to travel in a single direction (Stouffer *et al.*, 2015).

Web Application Firewalls (WAF), Network Firewalls, and Next-Generation Firewalls (NGFW)

A WAF protects web applications by targeting Hypertext Transfer Protocol (HTTP) traffic. This differs from a standard firewall, which provides a barrier between external and internal network traffic. A network firewall protects a secured local-area network from unauthorised access to prevent attacks. Traditional network firewalls mitigate against or prevent unauthorised access to private networks. Firewall policies define the traffic allowed onto the network; any other access attempts are blocked. Examples of the network traffic that this helps to prevent are unauthorised users and attacks from users or devices in less secure zones.

A WAF specifically targets application traffic. It protects HTTP and Hypertext Transfer Protocol Secure (HTTPS) traffic and applications in internet-facing zones of the network. This secures businesses against threats such as Cross-Site Scripting (XSS) attacks, Distributed Denial-Of-Service (DDoS) attacks, and SQL injection attacks (Stouffer *et al.*, 2015; Cisco, 2020). The NGFW provides several advanced tools that allow the integration of other features such as (intrusion prevention systems, application firewalls, proxy servers, URL filtering, and network packet inspection tools) to defend the network against malicious activity (Cawthra *et al.*, 2018; Cisco, 2020).

Redundancy

Redundancy is an operational requirement of the data centre that refers to the duplication of certain components or functions of a system so that, if they fail or need to be taken down for maintenance, others can take over. With redundancy, a switch may be need to be flipped to move from one server to another, for example (Stouffer *et al.*, 2015). Redundancy provides multiple protected instances of critical resources.

Endpoint Detection and Response (EDR)

Endpoint Detection and Response perform behavioural analytics on endpoint events from Symantec Endpoint Protection to identify potentially malicious behaviour. It can sandbox the impacted endpoints, prioritise risks, and provide tailored remediation guides (Cawthra *et al.*, 2018).

Secure Session Management

This is a process of securing and managing multiple users' sessions for each request (Gutzmann, 2001).

Privileged Access Management (PAM)

Privileged access includes those credentials with permission to access systems that is higher than that of standard users. Privileged access accounts often allow greater visibility of resources stored on systems and may allow modification of configuration settings or installation of software components (Cawthra *et al.*, 2018).

Principle Of Least Privilege (POLP)

This states that a subject should be given only those privileges needed for it to complete its task. If a subject does not need an access right, the subject should not have that right (Gegick and Barnum, 2005).

Justifications for each control

Authentication (Aun)

1. **Keyed cryptographic hash functions (HMAC), digital signatures, and Random numbers generators (Aun1)**

Justification: These controls ensure the authenticity and integrity of messages by using cryptographic techniques to verify the sender's identity and message content. HMAC provides a way to check the authenticity and integrity of a message, while digital signatures uniquely identify the sender and ensure non-repudiation. Random number generators are used to create unpredictable values, which are essential for secure cryptographic operations.

2. **Physically Unclonable Functions (PUF) (Aun2)**

Justification: PUFs leverage the inherent physical variations in hardware to generate unique identifiers that are nearly impossible to replicate. This control ensures strong authentication at the hardware level, preventing spoofing and impersonation.

3. **MAC-attached, and HORS-signed messages (Aun3)**

Justification: Message Authentication Codes (MAC) and Hash to Obtain Random Subset (HORS) signatures ensure the integrity and authenticity of messages. They protect against tampering and impersonation by providing a secure method of verifying the origin and content of messages.

4. **Secure Sockets Layer Certificates (SSL Certificates) and Transport Layer Security (TLS) (Aun4)**

Justification: SSL and TLS certificates establish encrypted connections between clients and servers, ensuring that the data transmitted is secure from eavesdropping and tampering. They provide strong mutual authentication and protect the integrity and confidentiality of data in transit.

5. **Multi-factor authentication mechanism (Aun5)**

Justification: Multi-factor authentication (MFA) requires multiple forms of verification (e.g., something you know, something you have, and something you are), significantly reducing the risk of unauthorized access through stolen credentials alone.

6. **Automatic lockouts (Aun6)**

Justification: Automatic lockouts help prevent brute force attacks by temporarily disabling accounts after a certain number of failed login attempts. This control ensures that repeated unauthorized attempts to gain access are mitigated.

Authorisation (Aur)

7. **Attribute-Based Encryption (Aur1)**

Justification: Attribute-Based Encryption (ABE) allows access to be controlled based on user attributes. It ensures that only users with the required attributes can decrypt and access certain data, providing fine-grained access control.

8. **Attribute Certificates (Aur2)**

Justification: Attribute certificates store user attributes and permissions separately from the main identity certificate. This separation allows for more flexible and scalable management of user permissions, ensuring that access rights are appropriately enforced.

9. **Attribute-Based Access Control System based on XACML (Extensible Access Control Markup Language) (Aur3)**

Justification: XACML provides a standardized way to define and enforce access control policies. Attribute-Based Access Control (ABAC) systems using XACML can evaluate multiple attributes and conditions to determine access permissions, offering granular and dynamic access control.

10. **Role-Based Access Control and allow/block listing (Aur4)**

Justification: Role-Based Access Control (RBAC) assigns permissions based on user roles, simplifying the management of access rights. Allow/block listing enhances security by explicitly defining which users or processes are permitted or denied access to resources.

Confidentiality (C)

11. **Symmetric and asymmetric algorithms and Public Key Infrastructure certificate (PKI) (C1)**

Justification: These cryptographic algorithms ensure that data remains confidential by encrypting it, making it unreadable to unauthorized users. PKI supports secure key exchange and digital signatures, enhancing overall data security.

Privacy (P)

12. **Anonymisation (P1)**

Justification: Anonymisation techniques remove personally identifiable information from data sets, protecting individuals' privacy by ensuring that data cannot be traced back to specific individuals.

13. **Trusted aggregators (P2)**

Justification: Trusted aggregators collect and process data from multiple sources while ensuring that individual data points remain anonymous and private. They play a critical role in maintaining data privacy in distributed systems.

14. **Homomorphic encryption (P3)**

Justification: Homomorphic encryption allows computations to be performed on encrypted data without decrypting it. This ensures that sensitive data remains confidential even during processing, preserving privacy.

15. **Perturbation models (P4)**

Justification: Perturbation models add noise to data, protecting privacy by making it difficult to identify individual data points. This technique is commonly used in data mining and statistical analysis to protect sensitive information.

16. **Verifiable computation models, and zero-knowledge proof systems (P5)**

Justification: These models allow one party to prove to another that a computation is correct without revealing any underlying data. Zero-knowledge proofs ensure that data remains confidential while providing assurance of the correctness of computations.

17. **Data obfuscation techniques (P6)**

Justification: Data obfuscation transforms data into a format that is difficult to understand without additional information. This technique helps protect sensitive information from unauthorized access and analysis.

Integrity (In)

18. **Cryptographic hashing functions and session keys (In1)**

Justification: Hashing functions ensure data integrity by producing a unique hash value for each data set, which changes if the data is altered. Session keys provide secure communication channels, ensuring data is not tampered with during transmission.

19. **Digital watermarking (In2)**

Justification: Digital watermarking embeds a unique identifier into data, which can be used to verify the integrity and authenticity of the data. This helps detect unauthorized modifications and distribution.

20. **Automated patch management for flaw remediation (In3)**

Justification: Automated patch management ensures that security patches are applied promptly, reducing vulnerabilities that could be exploited to compromise data integrity.

21. **Adaptive cumulative sum algorithm (In4)**

Justification: This algorithm detects changes in data patterns, identifying potential integrity breaches. It is useful for monitoring and maintaining the integrity of large data sets.

22. **Secure Phasor Measurement Units (PMUs) installation (In5)**

Justification: PMUs provide real-time monitoring of electrical systems, ensuring the integrity of measurements and data used for system operations.

23. **Load profiling algorithms (In6)**

Justification: These algorithms monitor and analyze usage patterns to detect anomalies that could indicate data integrity issues.

24. **Timestamps (In7)**

Justification: Timestamps provide a record of when data was created, modified, or accessed, helping to ensure the integrity of data by tracking its history.

25. **Sequence numbers (In8)**

Justification: Sequence numbers ensure that data packets are processed in the correct order, preventing data loss or duplication that could compromise integrity.

26. **Query sanitisation (In9)**

Justification: Query sanitization ensures that input data is properly validated and cleaned, preventing SQL injection and other attacks that could compromise data integrity.

27. **Nonces (In10)**

Justification: Nonces are unique values used once in a cryptographic communication to prevent replay attacks, ensuring the integrity of each communication session.

Availability (Av)

28. **Use multiple alternate frequency channels according to a hardcoded sequence (Av1)**

Justification: This control ensures availability by using multiple communication channels, reducing the risk of single points of failure and jamming attacks.

29. **Frequency quorum rendezvous between connected nodes (Av2)**

Justification: Ensures reliable communication between nodes by agreeing on a common frequency, maintaining availability even in the presence of interference.

30. **Anomaly Intrusion Detection Systems (IDS) (Av3)**

Justification: IDS detect and respond to unusual activity that may indicate an attack, helping to maintain the availability of systems by mitigating potential threats.

31. **Specification-based IDS (Av4)**

Justification: This type of IDS uses predefined models to detect deviations from normal behavior, providing a robust method for ensuring system availability.

32. **Intrusion Prevention Systems (IPS) (Av5)**

Justification: IPS actively prevent attacks by blocking malicious traffic, ensuring that systems remain available and operational.

33. **Quality of Services (QoS) (Av6)**

Justification: QoS mechanisms manage network traffic to ensure that critical services receive the bandwidth and resources they need, maintaining availability.

34. **Load balancing (Av7)**

Justification: Distributes network traffic and workloads across multiple servers, preventing any single server from becoming a bottleneck and ensuring continuous availability.

35. **Operating system-independent applications (Av8)**

Justification: Ensures that applications can run on multiple operating systems, reducing dependency on any single system and enhancing availability.

Non-repudiation (N)

36. **Mutual Inspection technique (N1)**

Justification: Ensures that both parties in a transaction verify each other's actions, preventing either party from denying their involvement.

37. **Unique keys and digital signatures (N2)**

Justification: Provide a secure method to verify the authenticity and integrity of messages, ensuring that actions can be attributed to the correct source.

38. **Transaction log (N3)**

Justification: Maintains a record of all transactions, providing an audit trail that can be used to verify actions and ensure accountability.