Security of Critical Cyber-Physical Systems: Fundamentals and Optimization

AbdelRahman Eldosouky

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

Doctor of Philosophy
in
Electrical Engineering

Walid Saad, Chair
Jeffrey H. Reed
Ryan M. Gerdes
Hesham A. Rakha
Anil Kumar S. Vullikanti

May 03, 2019
Blacksburg, Virginia

Keywords: Cyber-Physical Systems Security, Critical Infrastructure Resilience, Unmanned Aerial Vehicles, Game Theory, Moving Target Defense, Contract Theory, Internet of Things

Security of Critical Cyber-Physical Systems: Fundamentals and Optimization

AbdelRahman A. Eldosouky

ABSTRACT

Cyber-physical systems (CPSs) are systems that integrate physical elements with a cyber layer that enables sensing, monitoring, and processing the data from the physical components. Examples of CPSs include autonomous vehicles, unmanned aerial vehicles (UAVs), smart grids, and the Internet of Things (IoT). In particular, many critical infrastructure (CI) that are vital to our modern day cities and communities, are CPSs. This wide range of CPSs domains represents a cornerstone of smart cities in which various CPSs are connected to provide efficient services. However, this level of connectivity has brought forward new security challenges and has left CPSs vulnerable to many cyber-physical attacks and disruptive events that can utilize the cyber layer to cause damage to both cyber and physical components. Addressing these security and operation challenges requires developing new security solutions to prevent and mitigate the effects of cyber and physical attacks as well as improving the CPSs response in face of disruptive events, which is known as the CPS resilience.

To this end, the primary goal of this dissertation is to develop novel analytical tools that can be used to study, analyze, and optimize the resilience and security of critical CPSs. In particular, this dissertation presents a number of key contributions that pertain to the security and the resilience of multiple CPSs that include power systems, the Internet of Things (IoT), UAVs, and transportation networks. First, a mathematical framework is proposed to analyze and mitigate the effects of GPS spoofing attacks against UAVs. The proposed framework uses system dynamics to model the optimal routes which UAVs can follow in normal operations and under GPS spoofing attacks. A countermeasure mechanism, built on the premise of cooperative localization, is then developed to mitigate the effects of these GPS spoofing attacks. To practically deploy the proposed defense mechanism, a dynamic Stackelberg game is formulated to model the interactions between a GPS spoofer and a drone operator. The equilibrium strategies of the game are analytically characterized and studied through a novel, computationally efficient algorithm. Simulation results show that, when combined with the Stackelberg strategies, the proposed defense mechanism will outperform baseline strategy selection techniques in terms of reducing the possibility of UAV capture. Next, a game-theoretic framework is developed to model a novel moving target defense (MTD) mechanism that enables CPSs to randomize their configurations to proactive deter impending attacks. By adopting an MTD approach, a CPS can enhance its security against potential attacks by increasing the uncertainty on the attacker. The equilibrium of the developed single-controller, stochastic MTD game is then analyzed. Simulation results show that the proposed framework can significantly improve the overall utility of the defender. Third, the concept of MTD is coupled with new cryptographic algorithms for enhancing the security of an mHealth Internet of Things (IoT) system. In particular, using a combination of theory and implementation, a framework is introduced to enable the IoT devices to update their cryptographic keys locally to eliminate the risk of being revealed while they are shared.

Considering the resilience of CPSs, a novel framework for analyzing the component- and system-

level resilience of CIs is proposed. This framework brings together new ideas from Bayesian networks and contract theory – a Nobel prize winning theory – to define a concrete system-level resilience index for CIs and to optimize the allocation of resources, such as redundant components, monitoring devices, or UAVs to help those CIs improve their resilience. In particular, the developed resilience index is able to account for the effect of CI components on the its probability of failure. Meanwhile, using contract theory, a comprehensive resource allocation framework is proposed enabling the system operator to optimally allocate resources to each individual CI based on its economic contribution to the entire system. Simulation results show that the system operator can economically benefit from allocating the resources while dams can have a significant improvement in their resilience indices. Subsequently, the developed contract-theoretic framework is extended to account for cases of asymmetric information in which the system operator has only partial information about the CIs being in some vulnerability and criticality levels. Under such asymmetry, it is shown that the proposed approach maximizes the system operator's utility while ensuring that no CI has an incentive to ask for another contract. Next, a proof-of-concept framework is introduced to analyze and improve the resilience of transportation networks against flooding. The effect of flooding on road capacities and on the free-flow travel time, is considered for different rain intensities and roads preparedness. Meanwhile, the total system's travel time before and after flooding is evaluated using the concept of a Wardrop equilibrium. To this end, a proactive mechanism is developed to reduce the system's travel time, after flooding, by shifting capacities (available lanes) between same road sides. In a nutshell, this dissertation provides a suite of analytical techniques that allow the optimization of security and resilience across multiple CPSs.

Security of Critical Cyber-Physical Systems: Fundamentals and Optimization

AbdelRahman A. Eldosouky

GENERAL AUDIENCE ABSTRACT

Cyber-physical systems (CPSs) have recently been used in many application domains because of their ability to integrate physical elements with a cyber layer allowing for sensing, monitoring, and remote controlling. This pervasive use of CPSs in different applications has brought forward new security challenges and threats. Malicious attacks can now leverage the connectivity of the cyber layer to launch remote attacks and cause damage to the physical components. Taking these threats into consideration, it became imperative to ensure the security of CPSs.

Given that many CPSs provide critical services, for instance many critical infrastructure (CI) are CPSs such as smart girds and nuclear reactors; it is then inevitable to ensure that these critical CPSs can maintain proper operation. One key measure of the CPS's functionality, is resilience which evaluates the ability of a CPS to deliver its designated service under potentially disruptive situations. In general, resilience measures a CPS's ability to adapt or rapidly recover from disruptive events. Therefore, it is crucial for CPSs to be resilient in face of potential failures.

To this end, the central goal of this dissertation is to develop novel analytical frameworks that can evaluate and improve security and resilience of CPSs. In these frameworks, cross-disciplinary tools are used from game theory, contract theory, and optimization to develop robust analytical solutions for security and resilience problems. In particular, these frameworks led to the following key contributions in cyber security: developing an analytical framework to mitigate the effects of GPS spoofing attacks against UAVs, introducing a game-theoretic moving target defense (MTD) framework to improve the cyber security, and securing data privacy in m-health Internet of Things (IoT) networks using a MTD cryptographic framework. In addition, the dissertation led to the following contributions in CI resilience: developing a general framework using Bayesian Networks to evaluate and improve the resilience of CIs against their components failure, introducing a contract-theoretic model to allocate resources to multiple connected CIs under complete and asymmetric information scenarios, providing a proactive plan to improve the resilience of transportation networks against flooding, and, finally, developing an environment-aware framework to deploy UAVs in disaster-areas.

To my wife, Ehsan,
and my kids, Omar, Adel, and Ali.

Acknowledgments

First of all, I praise Allah, the almighty, for all his blessings and for giving me the strength and the perseverance to accomplish my PhD work including this dissertation.

I would like to thank everyone who has helped or supported me during my PhD program. First and foremost, I owe my deepest gratitude to my Ph.D. advisor, Dr. Walid Saad. Without his help, guidance, and patience through my ups and downs, I could not make it to finish this dissertation. I would like to thank him for the amount of time and effort he has dedicated to make my Ph.D. experience productive and for allowing me to benefit from his knowledge and experience. I am thankful for his priceless advice and for his invaluable mentorship that guided me through my PhD journey and gave my the confidence to explore new research directions that allowed me to finish the work done in this dissertation. I would like to thank the members of my Ph.D. advisory committee, Dr. Jeffrey H. Reed, Dr. Ryan M. Gerdes, Dr. Hesham A. Rakha, and Dr. Anil Kumar S. Vullikanti, for their valuable comments and guidance which were tremendously helpful in improving the quality of this dissertation.

I am also grateful to all my friends and colleagues at the NEWS labs (formerly NetSciWiS), those who have graduated and those who are still on track. I was fortunate to be surrounded by them and work with all of them. I really appreciate the friendly atmosphere in our group that made my PhD time enjoyable. I would like also to extend this gratefulness to all my fiends and colleagues at Wireless@VT.

I would like to thank my wife, Ehsan, and my kids Omar, Adel, and Ali for their endless love and support. I cannot find the words to express my immense gratitude to my wife, Ehsan. Indeed, without her support, I could have never been able to finish this dissertation. Thanks for always believing in me and for being my source of strength. I would also like to thank my parents. Without their encouragement, I would not have started my PhD program. Their unconditional love and support have been really helpful and gave me the energy to finish my PhD work. I am also heartily thankful to my brother and sister for their emotional support.

Finally, I want to thank all my friends whom I met in Blacksburg who made me feel at home with their warm feelings and who were always available when I needed them. My thanks continue to all my friends in Egypt who kept supporting and encouraging me during the course of my PhD.

Contents

List of Figures

List of Tables

Chapter 1

Motivation, Background, and Contributions

In the era of the Internet of Things (IoT), cyber-physical systems(CPSs) have emerged as a revolutionary technology that transforms the way humans interact with engineered systems. CPSs use a cyber layer to integrate sensing, computation, data processing, communication, and control into physical objects and infrastructure. This gives the ability to the traditional physical systems to connect to the Internet and to communicate with each other; allowing them to be more sustainable, resilient, and efficient [2]. Examples of CPSs include unmanned aerial vehicles, smart grids, the Internet of Things (IoT), smart cities, and many other critical infrastructure (CI) systems. Critical infrastructure (CI) system are the systems that are vital to modern day cities and communities such as power grids and transportation systems. In the United States, the Department of Homeland Security classifies CIs into sixteen sectors that include energy production, financial services, communications, nuclear reactors, transportation systems, water supply, and financial services [3]. While this classification is not general and each country determines its own sectors, protecting and maintaining the service of CIs is considered a national security in most countries. Owing to this essential role of CIs, protecting and securing CI proper operation has recently attracted significant attention [4–18]. Besides CIs that are critical by nature, other critical CPSs include systems that provide time-critical services. For instance, drone-delivery systems [19] can be used to deliver medications, special equipment, or goods to natural disaster sites as well as remote areas.

However, the widespread use of CPSs, in different applications, exposes them to a plethora of security threats that include cyber, physical, and cyber-physical attacks [20]. Different types of attacks can target CPSs causing damage to the both cyber and physical components. In particular, the cyber layer brings forward new security threats and vulnerabilities that can jeopardize the whole system security. These security threats arise from the attacker's ability to launch remote and coordinated attacks simultaneously from the cyber layer towards the physical components. In a typical CPS, an attacker can exert disruptions and damages to the system that range from service interruptions, manipulating data, taking control of the system, to life-threatening attacks, in which the securities of humans, societies, or a whole nation can be negatively affected [21].

Taking these possible threats into consideration and given the vital role of the critical CPSs, it

became crucial to develop novel security solutions for the CPSs. The main goal of these security solutions should be to help the CPSs to maintain proper operations under possible attacks and expected failures. Moreover, suitable measures need also to be defined to evaluate the functionality and the ability of these critical CPSs to deliver its designated service, under potentially disruptive events or attacks. Of these measures, *reliability and resilience* are two of the most widely adopted measures in literature. The importance of studying reliability and resilience for critical CPSs stems from the fact that they are prone to many disruptive events such as natural disasters, hazardous conditions, subversive attacks, aging, or even inadequate maintenance [22]. Thus, it is crucial for critical CPSs to operate reliably and to be resilient in face of potential attacks and disruptions. Both reliability and resilience can be used to evaluate the functionality of a Critical CPS, however, in practice, there is a significant difference between reliability and resilience. Reliability is a term that describes the frequency or the likelihood of a system's failure [23]. Different numeric metrics, that comply with the aforementioned definition, are used to evaluate infrastructures' reliability and help to improve it. For instance, IEEE defines reliability metrics for power distribution systems as a relation between the effect of service interruption and the number of customers [24]. In transportation networks, reliability can be defined as the probability that a trip, from a certain origin to a certain destination, will arrive within a given period of time. Arriving after the defined time period is considered a failure [25]. Similar reliability metrics are defined for different critical infrastructures. Resilience on the other hand, does not have a common definition like reliability, rather it has multiple different definitions [26]. Most of these definitions assume a change in or a corruption to the system's normal functionality that affects the system's performance. A general definition of resilience, given by the a DHS advisory council, is the ability of an infrastructure to adapt to or rapidly recover from a potentially disruptive event [27]. Lacking a standard definition, resilience metrics do not have a consistent approach [27] and hence improvement techniques can vary widely. In [28], a general framework for analyzing and quantifying the resilience was introduced which defined four properties for the resilience: robustness, redundancy, resourcefulness, and rapidity. Resilience is then quantified using four interrelated dimensions: technical, organizational, social, and economic. The technical dimension considers the physical system and the relation between its physical components. Organizational dimension considers the ability of the organizations that manage the infrastructures to take actions and apply failure-related functions to improve their infrastructures' resilience. The social dimension considers the effect of disasters on communities, and finally the economic dimension deals with the direct or indirect losses due to failures. Based on this framework, many studies have either adopted or can be categorized to assess the resilience using the properties and the dimensions defined in this framework.

Studying the resilience of critical CPSs is, the, considered an integrated part CPSs security and protection. This is due to the fact that, failing to protect critical CPSs against attacks will result in increasing the CPSs probability of failure and reducing the time in which the CPSs is providing its intended service which will result in a poor resilient performance of the CPSs. Typically, CPSs protection can be achieved by applying security mechanisms and anticipating possible attacks to mitigate their effects. In [8] CIs are studied as the nation's key assets and the presidential executive orders for protecting CIs are discussed. It was shown that it is almost impossible to study the physical security apart from the cyber security. *Physical security* according to [8], means protecting

CIs from damage caused by physical attacks or physical forces such as explosions and fire while *cyber security* means protecting both the physical and cyber components from cyber attacks such as data manipulation and operational failure. Therefore, joint *cyber-physical security* is essential for critical CPSs and CIs which will include jointly applying physical defensive strategies, e.g., guards and fences, along with cyber defensive strategies such as firewalls and anti-viruses.

The pressing necessity for critical CPSs protection is expected to continue to rise with the move towards smart cities [29]. In a smart city, it will be imperative to monitor and incorporate the data collected from its CPSs to provide better services for citizens and to help prepare preventive security and maintenance plans [30]. This, in turn, requires using smart computing technologies and smart monitoring systems to make the critical CPSs more intelligent, efficient, and more interconnected [31] which can come at a cost to CPSs security, as CPSs will become more vulnerable to cyber attacks, which in turn can affect their physical components [32–34]. As decisions in smart cities are taken based on the conditions of their CPSs that are retrieved from the monitoring systems, e.g., sensors, cyber attacks such as jamming, data injections, or eavesdropping can have a great effect on the CPSs behavior and, hence, on its physical components. Therefore, most of CPSs protection work in literature focus on the cyber protection of the CPSs. This is because of the wide range of cyber attacks that can be performed against the CPSs especially after the increased dependability on the Internet of Things (IoT) to connect the cyber components of the CPSs to the Internet [35] and [36]. The IoT brought new challenges in protecting the critical CPSs both of the cyber and physical components as discussed later.

In the following, some of the CPSs security approaches are covered along with other work for quantifying the resilience. In this chapter, we first present the main approaches of protecting and securing the CPSs. Then, we will focus on the challenges and the relevant existing literature of CPSs resilience, followed by our contributions.

1.1 Security of Cyber-Physical Systems

Securing critical CPSs is imperative to maintaining their functionality and their proper functionality. In this section, we discuss the major aspects of CPSs security and some of the approaches that were developed to secure the CPSs.

1.1.1 Major CPS Security Threats

In this section, we discuss the major cyber attacks and threats that target CPSs. In general, attackers can exploit one or more major components of the system to perform their attacks [37]. For instance, sensors and actuators are two main components that can be targeted at the perception layer. On the other hand, data communication channels are the target of attacks at the transmission layer. Other attacks that can exploit components at more than one layer include:

- *Eavesdropping*: In this type of attacks, an attacker intercepts the transmitted data. The attacker can benefit from this attack by obtaining information about the control system within the CPS or obtaining data from the sensors that monitor the CPS. Although eavesdropping is a passive attack, i.e., the attacker does not interfere with the system operation, it can lead to more serious attacks as the attacker reveals sensitive information about the CPS. In other cases, eavesdropping as a standalone attack can be serious when the intercepted data concerns users' privacy as in medical systems [38].

- *Spoofing / Impersonating*: In this type of attacks, an attacker pretends to be a legitimate part of the system, e.g., a sensor sending its readings then it attempts to be intervene with the system operation. The attacker may use this to gain access to information or inject false data that can affect the CPS functionality. An example of this is Stuxnet worm [39], in which Stuxnet was impersonating the normal behavior of a programmable logic controller (PLC).

- *Man-in-the-Middle*: In this type of attacks, the attacker sends counterfeit signals to controllers or actuators to initiate harmful actions to the CPSs. The attacker, for example, can then control some functions of the system. This type of attacks is similar to false data injection, in which the attacker injects crafted data into the CPS either at the sensors or the controllers. The goal of the attacker is to force the physical system into a potentially unsafe state by either manipulating the controllers directly or through the false sensors measurements. For example, a man-in-the-middle attack was initiated against address resolution protocol (ARP) in supervisory control and data acquisition (SCADA) systems [40].

- *Denial of Service*: This type of attacks can cause serious damage to the CPSs if the attacker was able to block certain functionalities within the CPS. The attacker can flood the controllers or data sinks with many false requests or measurements, that can, in turn, cause buffer overflow withing the system and cause it to quit normal operation states. In [41], a framework was introduced to address the problem of denial of service attacks against communication channels in CPSs. The proposed security framework depends on intrusion detection systems and robust control and is applied on a case study of power systems.

- *Replay*: In this type of attacks, an attacker gains access to a legit node in the system and monitor carefully its outputs. The attacker can, then, creates output signals similar to the original signals but with altered information. One goal of these attacks is to cause the CPSs to reach instability state. This attack is usually performed with other sophisticated attacks in order to cause serious damage to the system. In those attacks, stealthy data injection is performed so the attacker appears as a trusted node to the CPS [42].

Besides these attacks, there are also other common attacks that target the CPSs. Of these attacks, communication attacks can be seen as a real threat in CPSs. This is because, CPSs depend mainly on the cyber layer which involves the communication between the different components. Jamming attacks [43] can block the sensors wireless signals from reaching the controllers causing disruptions in the controllers' view of the systems status. Thus, the controllers can make unpredictable actions

causing harm to the CPS. Routing protocol attacks can also compromise the CPSs behavior. In this type of attacks, specific routers can be targeted by injecting bogus routes, and, hence, all communication signals in the entire parts of the CPS will be disrupted. This type of attacks usually applies to large scale CPSs such as the smart grid [44]. Finally, selective forwarding attacks can affect some nodes in the system causing them to forward only selected packets to the relays or the controllers. This type of attacks can cause similar effects as jamming attacks.

Sophisticated attackers can perform one or more of the previous attacks simultaneously or sequentially. Here, we discuss two forms of these attacks.

- *Coordinated Attacks*: This type of attacks represent combining several attacks simultaneously on different parts of a CPS. For example, the authors in [45] discussed the possibility of performing coordinated attacks against power systems. The attacks are coordinated in the sense that multiple nodes are being attacked at the same time using both physical and cyber attacks. Physical attacks are meant to cause line outages, while cyber attacks can manipulate the topology preserving and the load redistribution within the power system. These combined attacks can potentially exasperate outages to trigger cascading failures by leading to undetectable line outages.
- *Persistent Attacks*: In this type of attacks, an attacker performs one or more attacks sequentially against a specific part of the CPS. The primary objective of the attacker is to cause impairment failure to the whole system by choosing a critical point to attack [46]. The basic idea of this types of attacks is to weaken some point in the system over time, so it reaches a failure state, and, hence, affects the whole system.

After discussing the major cyber-physical attacks against CPSs, we then explore the attacks that target some critical CPSs.

1.1.2 Cyber-physical Attacks against Critical CPSs

Due to the key role they play in modern societies, critical CPSs have been target to different types of disruptive attacks. While there are common attacks that target the cyber layer as discussed earlier, many other attacks vary widely based on the type of the critical CPS. In the following, we will discuss some types of critical CPSs and the attacks that can target them.

- *The Internet of Things (IoT)*: The Internet of Things (IoT) is seen as a large-scale ecosystem that will integrate a heterogeneous mix of devices, sensors, and wearable devices. The IoT provides a general framework that can be exploited by a variety of smart services and technologies. However, it poses various security challenges to the systems connected to the Internet as these systems will be prone to remote cyber attacks. In [47], the different components of a CPS are evaluated based on their sensitivity to attacks, when a CPS is connected to the Internet. For instance, communication layers are prone to many attacks

and they are highly sensitive to integrity, authenticity, and confidentiality attacks. Sensors, on the other hand, are more sensitive to privacy and regulation attacks while actuators are the least affected by the cyber attacks among CPS components. Another challenge in securing IoT devices against cyber-physical attacks, is the limited resources nature of the IoT devices. Many IoT have limited power because they are battery-operated. This requires lightweight security protocols to be developed that cope with the limited resources. For example, traditional encryption techniques cannot be used with the IoT devices due to their high computational requirements. Instead, encryption techniques that use shorter keys and require less computations are used.

- *Unmanned Aerial Vehicles (UAVs)*: UAVs or as popularly known, drones, have recently been widely used in many applications such as providing communications in emergency situations, delivering goods, and for monitoring services. This wide use of UAVs was due to the advancements in their design that makes them efficient and reliable in many situations where other types of transportation are not suitable. For instance, UAVs can help provide aerial services in remote areas or hard to reach regions such as mountains or valleys. In addition, UAVs can contribute to many critical applications such delivering medical supplies to war zones or providing the essential communication services to disaster-sites. This wide use of UAVs raised concerns about their security and resilience facing cyber-physical attacks. Examples of cyber-physical attacks targeting UAVs are false data injection, GPS spoofing, GPS jamming, hijacking the controller's signals, and deactivation attacks. In [48], different UAVs' vulnerabilities to cyber-physical attacks have been discussed. In particular, potential cyber and physical threats that can arise from the use of UAVs are presented. To detect the malicious UAVs, many techniques were studied that depend on the radio signals emitted from the UAVs, acoustic sensors, and radars.

- *Smart Grid*: The smart grid has recently been widely studied in literature. The smart grid is basically an electrical power grid that utilizes smart devices such smart meters and smart generators. These smart devices are connected using a network so they can exchange data. The smart grid can then integrate the actions of the connected users with the smart devices in order to provide an efficient and more reliable service. Due to the necessity of having a continuous electricity service, smart grids are becoming among the critical CPSs. The main cyber-physical threats that target smart grids was presented in [44]. As an example of physical attacks is meter bypassing which can cause instability to the grid due to the wrong measurements. On other hand, there are many cyber attack that can affect the smart grid such as malware spreading, compromising communication equipment, injecting false information on price and meter data, and eavesdropping attacks. In [49], the authors discussed the main weak points that make power grids vulnerable to different cyber and physical attacks by estimating the attack potential impacts on the grid. A layered risk evaluation model is presented that assess the risks in the control, transmission, and distribution layers.

- *Critical Infrastructure*: Many critical infrastructure represent CPSs such as transportation networks, power grids, water distribution, gas production, nuclear reactors, and dams. Due

to the wide range of types and applications, cyber-physical attacks vary widely among CIs. More details are discussed later about specific CIs types and their threats.

As discussed in this section, critical CPSs face many security threats that exploit vulnerabilities in each type of these systems. As such, many security solutions have been proposed in literature to avoid these threats and mitigate their effects based on the type of the critical CPSs. In the next sections, we discuss and explore the current security solutions that have been proposed or are currently being used to protect critical CPSs.

1.1.3 CPS Security solutions

To address the various security attacks and threats that target critical CPSs, many research works have focused on introducing new security solutions to CPSs. In [50], different CPS security solutions were discussed which fall into three categories: safety, security, and sustainability. Different requirements are listed for each category. For instance, to ensure CPS safety, security solutions are required to take into consideration characterization of the interactions between different cyber and physical components. This, in turn, requires understanding the dynamic nature of the CPSs and its different states over time. To ensure the security of CPSs, the coupling between the cyber and physical layer needs to be accounted for. This involves creating a mapping between the cyber components and their connected points in the physical environment. By doing so, each connected nodes will be protected together. Finally, to ensure sustainability, the relation between the CPSs and their surrounding environments needs to be considered in such security solutions.

In [51], a context-aware security framework is proposed for CPSs. The framework ensures three main aspects, in order to achieve the cyber-physical security, that are sensing security, cyber security, and control security. The main goal of sensing security is to ensure that available information is trusted, e.g., no sensor reading were manipulated. In cyber security part, communication, network, and software security need to be achieved which involves avoiding the common cyber attacks against CPSs, as discussed earlier. Finally, control security can be divided into feedback security and actuators security which ensure that the actuators can perform the correct actions while the controllers make the correct decisions.

In general, different security detection, mitigation, and prevention techniques can be used to thwart the cyber attacks against CPSs. Another promising techniques to improve a system's security is the so-called moving target defense (MTD) [52], which is discussed in more details next.

1.1.4 Moving Target Defense

Moving target defense (MTD) techniques [52] is one of the effective ways to thwart cyber attacks. MTDs is based on continuously randomizing the system's configuration (e.g., IP addresses and cryptographic keys) to increase the attacker's uncertainty and cost for performing a successful at-

tack. Applying MTDs in a system, e.g., a CI cyber network, requires meeting several challenges that range from optimizing the randomization to balancing the costs and the benefits of the randomization [53–61].

Recently, MTD has attracted significant attention [52–55]. First, in [52], the authors defined five domains in which MTD techniques could be applied against cyber attacks in critical systems. The domains were defined to be networks, platforms, runtime environments, software, and data. These domains capture the essential components of a cyber system which can be found in most systems. The work in [54] defined a higher class called system agility in which MTD can be considers as a subclass. The system agility is defined as any reasoned modification to a system or environment in response to a functional, performance, or security need. The author discussed the parameters that controls the timing and the ability to employ changes as MTD to improve the security of a system. They also considered the measurement of the effectiveness of different MTD approaches. In [55], the authors propose a foundation for defining the theory of MTD. They defined key problems related to MTD such as selecting next valid configuration of the system, configuration space (all valid reconfigurations), and the timing problem to apply MTD. They also defined the MTD Entropy Hypothesis, which measure the effectiveness of an MTD system as a relation in the system's entropy.

The methods of MTD were applied in resource-constrained distributed devices, e.g., wireless sensor networks and IoT [56, 57, 62, 63]. In [62], the authors proposed to add a new layer of security to the IoT. In this layer, frequently changing IPv6 addresses is applied as a MTD technique. The authors in [56], proposed two different reconfigurations for wireless sensor networks at different architectural layers. The first is applied at, their newly defined security layer, by dynamically changing the data encryption techniques where each node can choose its own encryption technique, from a pre-defined list of techniques. The second reconfiguration is to applied at the physical layer by changing the node's firmware. Changing the node's firmware was shown to incur more cost as it requires the nodes to be off for some time. In [57], MTD is used to defend against selective jamming attacks to protect network parts from being isolated by an attacker. Finally, in MTD techniques, there is a concept of attack surface [63] which represents the points that could be attacked. The defender's goal is to change the attack surface so as to harden the attacker's mission.

Next, we discuss the other aspect of CPS security, which is the resilience. In particular, the CPSs resilience is explored and its relation to CPS security.

1.2 Resilience of Cyber-Physical Critical Systems

In this section, we present the main relation between resilience and cyber-physical security of critical systems, in particular, critical infrastructure.

1.2.1 Critical CPSs Resilience and Reliability

In the previous section, we discussed the importance of evaluating and improving the resilience and reliability of critical CPSs as they are prone to multiple disruptive events that can affect or stop their operation. Natural disasters such as floods, hurricanes, and earthquakes have great effect on CI. Other non-attack factors include aging, inappropriate maintenance and components failure. To evaluate the ability of a CI to perform under these factors, resilience and reliability are two key measures that can used [23] and [64]. In [23], reliability describes the likelihood of a system to fail while resilience describes the rapidity of a system to recover from a failure. In terms of the probabilities of success and failure, reliability is the probability of a system to be in a success state, while the resilience is the probability to return to a success state once being in a failure state.

In [64], the author gave definitions to robustness, resilience and reliability in the context of power systems. Reliability is defined as the ability of a system (or a CI) to retain its intended function under given conditions when it is subject to internal or external failures. Reliability is calculated as a probability over a given time period. Resilience, on the other hand, is defined as the ability of a system (or a CI) to degrade its function by changing its structure (components) when it is subject to perturbations and to quickly recover to its full state after the perturbations are ceased. Usually, resilience is studied in light of a class of events and a CI is said to be resilient to these events.

In this work, we focus on the resilience, of critical CPSs, as a proactive approach that can be used to enhance their operation under disruptive events. Resilience, in general, can be improved by altering the current system's configuration (components). Resilience is believed to be more critical (than reliability) as it studies the CPS performance under failure events that can prevent it from performing its intended functionality. Next, we will discuss the main aspects of of resilience as related to security, then, we will study in details the approaches of CI resilience and their challenges in literature.

1.2.2 Resilience and Cyber-Physical Security

In the previous section, we explained how resilience determines the CPS response to failure events. As cyber attacks are meant to cause both cyber and physical failures to the targeted CPS, it is important for CPSs to withstand cyber attacks and to be more resilient to the cyber-physical attacks that target both components. Cyber security and resilience are now considered as two integrated parts of CPSs protection according to the Department of Homeland Security [15]. In [65], the authors listed cyber-physical resilience as one of the challenges in making cyber-physical systems fault-free. In [16], the authors studied the resilience of cyber-physical systems. They proposed a control design framework in which the system stochastically switches between its structure states to improve the resilience. The authors in [66] stated two properties that are essential for the resilience of smart grids. The first pertains to the physical part where the system itself should withstand disruptive events and the second is to ensure the correctness of sensors readings in order to have an accurate state estimation. The relation between security and resilience is represented in

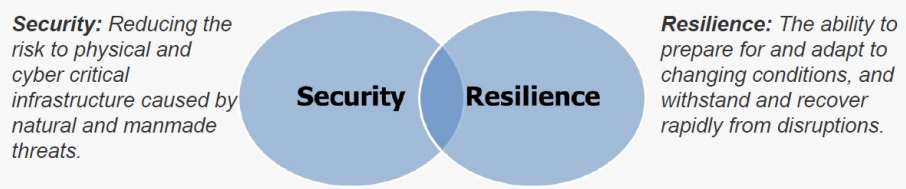


Figure 1.1: Relation between Resilience and Security.

Figure 1.1 from [10], where it is clear that both security and resilience have a common area where they affect each other and where the work in this area can consider both resilience and security.

This focuses on cyber security is enhanced by inheriting the IoT technologies in the CI of the smart cities. IoT was recently used in many smart cities applications such as infrastructure management, healthcare and transportation [67–70]. For instance, in transportation systems, IoT devices monitor and control bridges and railways to predict future failures. Some recent approaches proposed to install IoT vehicular sensors to monitor the car's conditions such as speed, tailgating distance, acceleration, and frequency of lane changes to better study the traffic flow on roads and predict any traffic jams. In health care, IoT enabled devices allow installing body and health sensors on patients' homes to get real time data on a remote location such as a clinic or a hospital. These devices can also be controlled from a distance, e.g., from a hospital to provide urgent care according to the patient's condition.

These advances in using IoT in smart cities and their intelligent CI brought forward some concerns about the CI cyber security. CI, using IoT devices, became vulnerable to cyber attacks such as false data injection, jamming, and eavesdropping [35, 36, 68]. Some other attacks target CI physical components through the cyber components [71, 72]. In [71], the authors studied the problem of false data injection on smart grids and how this can affect the power generation in the grid. In [72], the authors studied the interdependency between gas, power, and water as CI and proposed a mechanism to defend against cyber attacks that tries to change the sensors readings in order to affect the whole system's operation. Furthermore, the specific nature of IoT and having a massive number of heterogeneous devices complicates the security defensive mechanisms and makes it hard to use the conventional mechanisms. As IoT devices are usually battery-operated and cannot have regular processing capabilities, they are considered to be limited in resources. These limited resources make it hard for the IoT devices to run complex security algorithms and , hence, special security techniques are used such as lightweight encryption. Next, we discuss different protections approaches that are designed specifically for CIs.

1.2.3 Critical Infrastructure Protection

In this section we focus on some approaches for CI protection which mainly consider the physical security part. CI protection has recently attracted significant attention [4–7, 14, 73], particularly following recent terrorist and malicious attacks. One of the effective mechanism in enhancing the

physical security is allocating resources within the CI to protect its weak or vulnerable points. In [73], the authors proposed a bi-level optimization solution to allocate resources among some vulnerable CIs to limit the impact of disruptive events on the CI. This solution considers an imaginary interdictor that tries to reduce the system efficiency as much as possible by hitting a number of the unprotected CI, while the system planner decides which CIs to support, by allocating protective resources, so that the system achieves the maximum operation in case of interdiction. Although, the solution was proven to be useful in protection planning, it assumed all the CI parameters and the number of possible interdiction are known which might not be true for every CI. In [14], that authors pointed out that investing in specific CI upgrades to avoid some risk events is not the optimal way for protecting CI as these risk events may or may not occur. The proposed a more economic approach by simulating real-life scenarios and implementing intelligent prediction approaches to mimic the surprising events that can face CI. The work in [4] focused on CI control systems by presenting a system to secure their functions and management tasks. The system is based on anomaly-based intrusion detection to secure the CIs in an automatic way with a little or even no involvement from the users. Similarly, the authors in [5] dealt with CI control systems by exposing and analyzing the vulnerabilities of these systems. They built their approach on protecting the programmable logic controllers, that represent the basic point of CI control systems, against threats and attacks. In [6], the authors proposed a risk-aware robotic sensor network and applied it to CI protection. The work in [7] studied the vulnerabilities and protection challenges that face CIs and proposed a collaborative game theoretic solution for CI protection. Although these works proposed solutions to the CI protection problem, they did not address the problem of studying CI components or how to effectively allocate resources within CIs to achieve the protection.

In summary, we have discussed the importance of improving CIs resilience to be able to withstand disruptive events and the main approaches in literature for cyber-physical security as an integrated part of CI resilience. Despite the attention that CI resilience has in literature, there are several technical challenges that must be also taken into account when addressing the CI resilience. Next, we will discuss these challenges in detail.

1.3 Security and Resilience Challenges of Critical CPSs

As discussed in the previous sections, it is crucial for critical CPSs to improve their resilience and to apply security solutions, however, effectively designing security mechanisms and resilience improvement techniques has many key challenges.

The first challenge is that critical CPSs cut across multiple domains which makes it difficult to develop generic security solutions. While some traditional security mechanism can still be used in CPSs, each CPS requires security mechanism tailored to its nature. For instance, the problem of GPS spoofing in UAVs [74] is different from spoofing other sensors in the smart grid. This requires handling the information differently, and, to develop security solutions suitable for the application. The effect of different CPS domains is more clear when studying the resilience, for instance, the

effect of flooding on smart transportation networks is very different from its effect on dams or telecommunications. The same applies to earthquakes where it has different effects according to the CPS domain. Therefore, considering a resilience evaluation or improvement technique that can cover multiple CPSs or CIs domains is a challenge.

The interdependency between different critical CPS, from the same type or different types creates new challenges in studying the security and resilience. Usually, the interdependency is not straightforward and the failure propagation cannot be fully predicted even for the same attacks or disruptive events. For instance, the failure of dams functionality, due to cyber attacks or natural disasters, can affect other CPSs such as other dams on the river downstream or if there is water leakage due to breaching, mostly every CI in the proximity area will be affected. Interdependency is believed to be the hardest challenge to address when considering the security and resilience of CPSs.

The variability of attack types and disruptive events is another challenge for CIs. As discussed in [64], the resilience is usually measured according to some disruptive event. There are typically a huge number of disruptive events that can be considered for every CI like natural disasters, components failure, inappropriate maintenance, and aging. Natural disasters, yet, can be earthquakes, flooding, tornado and so on. With all these disruptive events and natural disasters having different effects on the CI, the CI response should be different which makes it more challenging for CI resilience.

The next two challenges concerns the resilience, and in particular CI resilience. In literature, resilience does not have a common definition, moreover, there are multiple resilience perspectives in literature. In particular, different classifications for resilience exist in literature, and each piece of work can cover the resilience from one approach. For instance, some approaches consider the resilience in terms of the failure time before which the CI can return to its normal operation. Another approach considers the probability of failure where reducing the probability of failure means an improvement of the CI resilience. Other classifications of resilience include technical and organizational resilience while others consider the community resilience of CI failure. All these different approaches for resilience make it hard to develop an improvement technique that covers one or more of these approaches.

Finally, considering the resilience of individual CIs is very different from considering the resilience of a whole system of multiple CIs. For example, the resilience of the smart grid can be improved through installing redundant generators or transformers. However, the resilience of a single power plant is studied in light of its internal generators or components. Both approaches are valid to improve the resilience of the smart grid, however, they follow very different approaches. This concept of considering the whole system versus its individual units resilience creates another challenge for approaching the CI resilience.

Figure 1.2 summarizes the security and resilience challenges of CPSs. Next, we explain, in details, each of the challenges in Figure 1.2. The challenges are discussed from both the security point of view and the resilience point of view.

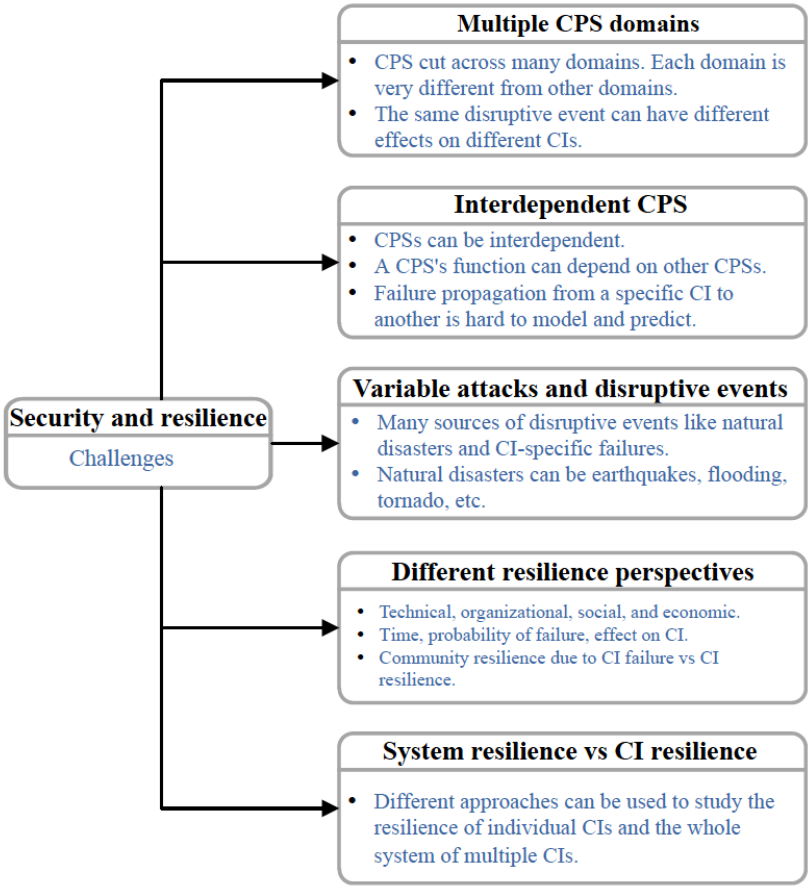


Figure 1.2: Challenges of evaluating and improving CI resilience.

1.3.1 Multiple CPS Domains

Security: As discussed earlier, critical CPSs can be found in many applications that are different in architecture and physical components. Hence, the security requirements of UAVs should be different from that of IoT devices. Each of those critical CPSs is subject to different attacks, for example, GPS spoofing attacks are common attacks that target UAVs, while eavesdropping and false data injection can target IoT devices. Computational capabilities are another difference between different CPSs that affects the nature of the security solutions adopted by each CPS. Hence, application-specific security solutions need to be developed based on the type of CPS and the expected attacks.

Resilience: As discussed earlier, resilience improvement techniques are typically infrastructure-specific due to the wide range of CI domains. In [75], the authors considered the resilience of transportation networks. The resilience is evaluated as the weighted average number of reliable passageways between two cities in the network graph. This approach of evaluating the resilience over a network graph cannot be applied to, for example, petrochemical CIs. The authors in [76]

considered the resilience of petrochemical CIs by the failure cost where the cost is considered for both the direct cost from the decreased productivity and the cost required for recovery activities. The proposed approach evaluated the resilience based on the cost and the time the system will be affected after disruptive events. These approaches are different from studying the resilience in power grids. In [77], the authors proposed a framework to improve the resilience of the power grid by installing redundant components such as generators. It is clear that the every CI type requires a specific resilience approach which poses many challenges for assessing and developing resilience improvement techniques for different-type interdependent CIs.

1.3.2 Interdependent CPSs

Security: Interdependency between different CPSs represent a serious concern for CPSs because of the cascading failure problem. In [78], the problem of cascading failure between two different networks representing a CPS, is discussed. In particular, the failures in one network's nodes can lead to potential failures in the other network's nodes. This problem is of particular concern in large scale systems such as the smart grid, where failures to one part of the grid can cause extra load on the other parts, and, hence, cascading failures.

Resilience: CPS interdependency poses many challenges on studying CI resilience due to the unpredictable effects or failures. In [18], the authors studied the resilience of the power delivery system and telecommunication system under a hurricane. The dependency was modeled as the role of power delivery systems in post-event (hurricane) telecommunications system recovery. While this represents a valid representation of interdependency, it did not cover failure of both CIs at the same time due to the disruptive event. The author in [11], studied the resilience of interdependent CIs and concluded a series of resilience improvement strategies of the interdependent CISs from three resilience capacities which are resistant, absorptive, and restorative capacity. He stated that these approaches require corresponding modeling and simulation approaches to study the interdependent CIs, which was not cover in his study. hence, studying the resilience od interdependent CIs remains an open challenge for resilience works.

1.3.3 Variable attacks and Disruptive Events

Security: As discussed earlier, different attacks can target the same CPS. In addition, one attack can cause different effects on different CPSs. For example, false data injection attacks on the smart grid can alter the meters' reading causing disrupts to the consumption measurements around the smart grid. The same data injection attacks on an m-health IoT devices can cause serious problems and threaten the patients life in case other medical devices took actions based on these measurements. Therefore, defending against data injection attacks in the smart grid required different security solutions than securing the m-health IoT devices against data injection attacks.

Resilience: Disruptive events is another challenge in improving the CI resilience as the resilience

is usually studied according to a specific disruptive event. In [79], the author listed the variable disruptive events that can affect various types of dams. It was shown that not all the disruptive events can have effect on all types of dams. In [80], the authors compared between flooding and earthquake effects on dams and what are the expected failures due to each disaster. The same concept applies to transportation networks where the effect of earthquakes is mostly studied for bridges [81] as they are the most transportation network element prone to earthquakes. Flooding, on the other hand, is studied for the effect on cars speed and travel time over the transportation network [82]. It is clear that different disruptive events have multiple effects on the same CI.

1.3.4 Different Resilience Perspectives

Resilience: As discussed earlier, resilience has multiple definitions that are typically application-dependent [26]. A general definition, given by the Department of Homeland Security (DHS), is the ability of a CI to adapt to or rapidly recover from a potentially disruptive event [27]. A general framework for analyzing and quantifying the resilience was introduced in [28]. The authors proposed four properties for the resilience: robustness, redundancy, resourcefulness, and rapidity. Resilience is then quantified using four interrelated dimensions: technical, organizational, social, and economic. The technical dimension considers the physical system and the relation between its physical components. Organizational dimension considers the ability of the organizations that manage the infrastructures to take actions and apply failure related functions to improve their infrastructures' resilience. The social dimension considers the effect of disasters on communities, and finally the economic dimension deals with the direct or indirect losses due to failures. Based on this framework, many studies have either adopted or can be categorized to assess the resilience using the properties and the dimensions defined in this framework.

Addressing the resilience of CIs from an organizational dimension evaluates the CI resilience based on satisfying a number of requirements. For example, the DHS recently contributed in developing an organizational resilience index, named the resilience measurement index (RMI) [83]. RMI is an indicator that determines to which degree the CI is prepared to possible failures and the extent to which recovery mechanisms and mitigation measures are installed within the CI. RMI can be used to compare the resilience levels, of CIs, however, it does not capture the effect of specific failure events on the infrastructure.

Economic resilience is similar in that it evaluates the economic losses of the resilience and how to reduce them [28] but it does not consider the effect of specific disruptive events on the CI. The social or community resilience studies the effect of CI failure on the community and how the people response can hinder or aggravate the failure losses. To improve the social resilience, plans need to be prepared to be used directly after disasters as it is believed that the first hours or days after the disaster have the highest effect [17].

The last, and the most important, dimension of CI resilience is the technical (physical). This can be considered as the most adopted dimension in literature [23, 84, 85]. In [84], the authors proposed a multi-stage framework to improve CI resilience. The resilience is evaluated based on the *time*

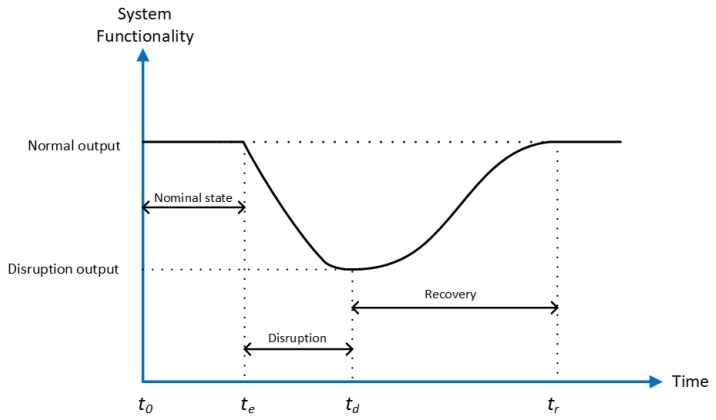


Figure 1.3: Generic system performance in case of a disruptive event.

effect of the disruptive events on the CI. The improvement process aims to reduce the time the CI spends in the recovery process. This improvement process depends on resources availability where basic actions are taken in case of limited resources while hardening components, using redundant components, or ensuring rapid recovery are the key elements for resilience improvement if resources are available. In [85], the authors present a framework to improve CI resilience based on the *effect* of disruptive events on the CI. The framework identifies the main system components that can be repaired to improve the resilience. Resilience is defined as the cumulative system functionality that has been restored at a given time. Figure 1.3 shows a generic CI performance under a disruptive event. Resilience approaches that target to reduce the *effect* on the system work on reducing the gap between the normal output and disruption output, while *time* improving approaches work on decrease the recovery time t_r which is measured after an event occurs at t_e and the maximum disruption at t_d. Finally, CI resilience can be defined as the probability of returning to a success state after the system fails [23]. The proposed framework defines the system reliability as the probability of being in a success state.

While the previous works provide detailed studies on CI resilience, most of the works that considered the technical dimension of the resilience abstract the cyber-physical components of the system making it hard to study the effect of CI's cyber components on the proposed resilience definitions. Furthermore, studying the resilience from an organizational dimension, while studies an individual CI in details, it lacks the analytical analysis of improving this resilience.

1.3.5 System Resilience and CI Resilience

Resilience: The last challenge in resilience is whether the resilience should be considers for a whole system of multiple CIs or for the individual CIs. This challenge is clear in power grids where an approach is to model the system as a graph with nodes representing the components, power plants, and links represent their connectivity. The whole system's resilience can be improved by approaches like shedding power from some areas while maintaining it in critical buildings or

areas. Another approach is asking some power plants to increase their production to compensate the lost power from one plant. On the other hand, studying individual power plants as the main focus of resilience means improving the operation conditions of the generation unit itself. The later approach may give more expensive and slower results but it is believed to give a more effective solution on the long run.

After discussing the challenges of developing security and resilience solutions to CPSs, we notice that the security solutions have to be application-specific due to the nature of the attacks and the architecture of different CPSs. Resilience, on the other hand, have more common features and provide a promising research direction to develop new approaches. In the following, we discuss the limitations od the current works in literature, while focusing on CI resilience, then, we present our key contributions to both CPSs security and resilience.

1.4 Limitations of Existing Works

As discussed in Section 1.2 and 1.3, in order to evaluate and improve the resilience of CIs, many technical challenges such as managing cyber security, dealing with multiple CI domains, and multiple disruptive events need to be taken into account. While the previous studies presented in Section 1.3 have addressed some of these challenges, the literature lacks comprehensive studies on evaluating and optimizing the performance of CIs under multiple disruptive events and considering the interdependency on a system of multiple CIs. In particular, several resilience-related problems such as evaluating the CI resilience in terms of both its cyber and physical components, providing realistic approaches to improve the resilience of both individual CIs and the whole system of multiple CIs are only marginally studied in the prior art. Moreover, the existing prior art has not provided comprehensive analytical frameworks that can handle all of these variables. In summary, the main limitations of the previous studies on CI resilience are as follows:

- Despite the notable number of works studying CI resilience, cyber-physical components of CIs are mostly abstracted within the CI. Thus, these effect of these components on the CI resilience is mostly not taken into consideration. This is more problematic in CIs with both cyber and physical components as these components can affect each other while the approach does not handle this interaction.
- Individual CIs are also abstracted within large-scale cyber-physical systems. For example, a work that studies the resilience of the power grid will consider power generators as just nodes in the power grid. The resilience is then measured for the entire system and not for individual system's components, i.e., individual CIs. This hinders the process of improving individual infrastructure's resilience as the failure events, for this CI, are not addressed and, hence, future failure events will have the same effect for each individual CI.
- One of the main resilience improvement techniques, as discussed earlier, is allocating resources within the system to improve the resilience. These resource allocation methods

lacks the procedure of allocating resources within each individual CI.

- Resilience studies that consider technical resilience dimension, while being useful in addressing the disruptive events and their effect on the CI, they do not provide detailed information about the CI like organizational dimension of the resilience.
- Most of the work in literature focus on the empirical implementation of MTD. While this can be effective to improve the security, it does not help to study the drawbacks of applying MTD on the entire system. For example, applying MTD incurs cost that needs to be evaluated to determine the effectiveness of the used MTD mechanism.
- The challenges of connecting CIs to the IoT are relatively new and did not gain enough attention in the literature.

In summary, despite the existence of a notable body of work on CI resilience, this existing literature lacks a comprehensive framework that can overcome all the aforementioned limitations. A general framework is therefore needed to evaluate the resilience of different CIs and to help in designing general resilience improvement techniques.

1.5 Summary of Contributions

The main contribution of this dissertation is to provide an in-depth analyses and to develop novel security defense mechanisms in specific critical CPSs problem. In addition, the dissertation provides an in-depth understanding and analyses of the resilience of critical CPSs. Both the security and resilience are studied in various application domains. Towards achieving these goals, this dissertation will develop novel defense mechanisms to mitigate specific cyber-physical security problems in UAVs, to improve the CPS cyber security through MTD, and to define a framework for IoT connected CPSs. In addition, we develop analytical foundations for evaluating and improving the resilience of CI. In particular, we propose various frameworks for evaluating different CI resilience and allocating resources withing CIs to improve their resilience. By using the proposed frameworks, the resilience of CIs can be optimized in terms of each CI probability of failure. This is performed while taking into consideration the cyber and physical components of each CI. The frameworks also covers both individual CIs and their economic contribution to a system of multiple CI in the resilience improvement process. To enable these contributions, this dissertation will weave together notions from contract theory, game theory, optimization techniques, and moving target defense. Indeed, using such advanced mathematical tools, this dissertation develops in-depth analytical foundations and efficient algorithms to mitigate the effects of cyber attacks against specific CPS and to evaluate, improve, and allocate resource within CI to make it more cyber-physical resilient. In summary, our contributions are given as follows:

1.5.1 Mitigating the Effects of GPS Spoofing Attacks Against UAVs

In Chapter 2, the problem of mitigating the effects of GPS spoofing against UAVs, is studied. In particular, UAVs are prone to capture attacks via GPS spoofing in which an attacker manipulates a UAV's global positioning system (GPS) signals in order to capture it. Given the anticipated widespread deployment of UAVs for various purposes, it is imperative to develop new security solutions against such attacks. In this chapter, a mathematical framework is introduced for analyzing and mitigating the effects of GPS spoofing attacks on UAVs. In particular, system dynamics are used to model the optimal routes that the UAVs will adopt to reach their destinations. The GPS spoofer's effect on each UAV's route is also captured by the model. To this end, the spoofer's optimal imposed locations on the UAVs, are analytically derived; allowing the UAVs to predict their traveling routes under attack. Then, a countermeasure mechanism is developed to mitigate the effect of the GPS spoofing attack. The countermeasure is built on the premise of cooperative localization, in which a UAV can determine its location using nearby UAVs instead of the possibly compromised GPS locations.

To better utilize the proposed defense mechanism, a dynamic Stackelberg game is formulated to model the interactions between a GPS spoofer and a drone operator. In particular, the drone operator acts as the leader that determines its optimal strategy in light of the spoofer's expected response strategy. The equilibrium strategies of the game are then analytically characterized and studied through a novel proposed algorithm. Simulation results show that, when combined with the Stackelberg strategies, the proposed defense mechanism will outperform baseline strategy selection techniques in terms of reducing the possibility of UAV capture.

1.5.2 Moving Target Defense for CI Cyber Security

In Chapter 3, a novel approach for implementing MTD techniques that can be used to randomize cryptographic techniques and keys in wireless networks is proposed. The wireless network can represent the cyber components in any CI. The wireless network consists of a central data sink, the base station (BS), that can receive data and control the other nodes or sensors. The goal for defining such a problem is the lack of analytical techniques, in literature, that enable one to quantify the benefits and tradeoffs of MTDs.

In particular, the problem is formulated as a stochastic game in which a BS, acting as a defender seeks to strategically change its cryptographic techniques and keys in an effort to deter an attacker that is trying to eavesdrop on the data. The game is shown to exhibit a single-controller property in which only one player, the defender, controls the state of the game. For this game, the existence and properties of the Nash equilibrium are studied, in the presence of a defense cost for using MTD. Then, a practical algorithm for deriving the equilibrium MTD strategies is derived. Our results show that the proposed game-theoretic MTD framework can significantly improve the overall utility of the defender, while enabling effective randomization over cryptographic techniques.

1.5.3 Cyber Security of IoT Connected CI

In Chapter 4, we consider a health system as one of the CI systems. A smart health system connected to the IoT represents what is known as m-Health services which allow monitoring the health status of patients while providing the ability for a rapid response in emergency cases. Connecting healthcare services to the IoT brings forward new security threats and vulnerabilities that can jeopardize the patients' private data. A novel security framework for m-Health IoT security is proposed using the concept of moving target defense (MTD). MTD allows the m-Health system to dynamically change its cryptographic keys to increase uncertainty on an attacker and secure the data. In the proposed scheme, the devices update their keys locally to eliminate the risk of revealing new keys while they are being shared with a gateway. A practical implementation is proposed based on bitslicing LED, a lightweight encryption cipher, to improve the performance of decrypting multiple packets at the same time. LED bitsliced implementation was tested on an ARM Cortex-A53 and was shown to consume half of the processor's instructions compared to the conventional implementation. The effect of applying MTD on the number of processor's instructions is evaluated and shown to be bounded.

1.5.4 CI Resilience Evaluation

In Chapter 5, the problem of optimizing and managing the resilience of CIs is studied. In particular, a comprehensive two-fold framework is proposed to improve CI resilience by considering both the individual CIs and their collective contribution to an entire system of multiple CIs. To this end, a novel analytical resilience index is proposed to measure the effect of each CI's physical or cyber components on its probability of failure. In particular, a Markov chain defining each CI's performance state and a Bayesian network modeling the probability of failure are introduced to infer each CI's resilience index. The introduced resilience index considers the analytical domain of the resilience. It quantifies the resilience of individual infrastructures and gives insights about improving this CI resilience in a similar manner to organizational resilience indices which consider each CI in depth. The framework is general enough to be used with CIs from different domains and it can handle multiple disruptive events through capturing their effect on the CI components.

We also introduced an algorithm, using the Bayesian network, to prioritize each CI components based on their effect on the CI probability of failure, and hence its resilience. Each CI can measure the improvement in its resilience due to fixing its components. Fixing, here, means repairing, replacing, or monitoring the components of interest based on each CI type.

1.5.5 Resource Allocation within multiple CI

Complete Information Scenario

As we discussed earlier, resource allocation is one of the effective techniques to improve CI resilience. Resources, which can be physical or cyber, are used to fix or replace components. In Chapter 5, we propose a novel approach to allocate resources in a system of multiple CIs in order to maximize the system's resilience. In particular, a comprehensive resource allocation framework, based on the tools of contract theory, is proposed enabling the system operator to optimally allocate resources, such as, redundant components or monitoring devices to each individual CI based on its economic contribution to the entire system. A contract can be seen as an agreement between the system operator and CI using which the system operator allocates resources and gets rewards in return. In this framework, the system operator is assumed to have complete information about each CI within the system. That enables the system operator to get the best benefit for each unit of resources it will allocate.

The optimal solution of the contract-based resilience resource allocation problem is analytically derived using dynamic programming. The proposed framework is then evaluated using a case study pertaining to hydropower dams and their interdependence to the power grid. Our results, within the case study, show that the system operator can economically benefit from allocating the resources while dams have a 60% average improvement over their initial resilience indices.

Asymmetric Information Scenario

In Chapter 6, we study the problem of allocating resources among multiple CIs when the system operator does not have complete information, rather it has asymmetric information about the CIs. A control center (CC), representing the system operator, is used to design contracts and offer them to infrastructures' owners. Contracts are designed in a way to maximize the CC's benefit and motivate each infrastructure to accept a contract and obtain proper resources for its protection. Infrastructures are defined by both vulnerability levels and criticality levels which are unknown to the CC. The CC is assumed to have the probability distribution of its belief about each CI being in each of the criticality and vulnerability levels. Therefore, each infrastructure can claim that it is the most vulnerable or critical to gain more resources. A novel mechanism is developed to handle such an asymmetric information while providing the optimal contract that motivates each infrastructure to reveal its actual type. The necessary and sufficient conditions for such resource allocation contracts under asymmetric information are derived. Our results show that the proposed contract-theoretic approach maximizes the CC's utility while ensuring that no infrastructure has an incentive to ask for another contract, despite the lack of exact information at the CC

1.5.6 Transportation Networks Resilience against Flooding

In Chapter 7, the problem of studying transportation networks' resilience under flooding is addressed. The effect of flooding on roads capacities and on their free-flow travel times, is studied taking into consideration the rain intensity and roads preparedness. An analytical framework, based on the concepts of Wardrop equilibrium, is formulated to model the total system travel times, for fixed demands, before and after flooding. A novel capacity shift mechanism is introduced in which roads' capacities, defined by the available lanes, can be either totally or partially shifted between same road sides. The problem of shifting capacities is formulated as bi-level optimization in which the upper level solves for the a possible capacity shift patterns, and the lower level is used to provide feedback about users' travel times. This feedback is, then, used to produce a better capacity shift pattern. The basic idea behind capacity shifts is to increase the total travel time on roads with smaller demands while decreasing the travel time for other roads. Simulation results have shown that the total system travel time can be slightly decreases when using the proposed solution.

1.5.7 UAVs for Communication Resilience

In Chapter 8, a software-based simulator for the deployment of base station-equipped unmanned aerial vehicles (UAVs) in a cellular network is proposed. To this end, the Google Earth Engine platform and its included image processing functions are used to collect geospatial data and to identify obstacles that can disrupt the line-of-sight (LoS) communications between UAVs and ground users. Given such geographical information, three environment-aware optimal UAV deployment scenarios are investigated using the developed simulator. In the first scenario, the positions of UAVs are optimized such that the number of ground users covered by UAVs is maximized. In the second scenario, the minimum number of UAVs needed to provide full coverage for all ground users is determined. Finally, given the load requirements of the ground users, the total flight time (i.e., energy) that the UAVs need to completely serve the ground users is minimized. Simulation results using a real area of the Virginia Tech campus show that the proposed environment-aware drone deployment framework with Google Earth input significantly enhances the network performance in terms of coverage and energy consumption, compared to classical deployment approaches that do not exploit geographical information. In particular, the results show that the proposed approach yields a coverage enhancement by a factor of 2, and a 65% improvement in energy-efficiency. The results have also shown the existence of an optimal number of drones that leads to a maximum wireless coverage performance.

1.6 List of Publications

As a byproduct of the above contributions, thus far, this dissertation has led to the following key publications:

1.6.1 Journal Publications

1. **A. Eldosouky**, W. Saad, and N. Mandayam, “Resilient Critical Infrastructure: Bayesian Network Analysis and Contract-Based Optimization," submitted for a journal publication.

2. **A. Eldosouky**, A. Ferdowsi, and W. Saad, “Drones in Distress: A Game-Theoretic Countermeasure for Protecting UAVs Against GPS Spoofing," submitted for a journal publication.

1.6.2 Conference Publications

1. **A. Eldosouky**, W. Saad, C. Kamhoua, and K. Kwiat, “Contract-Theoretic Resource Allocation for Critical Infrastructure Protection," in *Proc. of IEEE Global Communications Conference (GLOBECOM)*, San Diego, CA, USA, December 2015.

2. **A. Eldosouky**, W. Saad, and D. Niyat “Single Controller Stochastic Games for Optimized Moving Target Defense," in *Proc. of IEEE International Conference on Communications (ICC)*, Kuala Lumpur, Malaysia, May 2016.

3. **A. Eldosouky** and W. Saad, “On the Cybersecurity of m-Health IoT Systems with LED Bitslice Implementation," in *Proc. of the IEEE International Conference on Consumer Electronics (ICCE)*, Las Vegas, NV, USA, January 2018.

4. **A. Eldosouky** and W. Saad, “On the Resilience of Transportation Systems to Flooding in Coastal Cities: A Game-Theoretic Perspective," extended abstract, in *Proc. of Resilience Week - Students Competition Track*, Wilmington, DE, USA, September 2017.

5. A. French, M. Mozaffari, **A. Eldosouky**, and W. Saad, “Environment aware deployment of wireless drones base stations with Google Earth simulator,” in *Proceedings of UNAGI’19 - Workshop on Unmanned aerial vehicle Applications in the Smart City*, Kyoto,Japan, Mar 2019.

6. **A. Eldosouky**, A. Ferdowsi, and W. Saad, “Deceiving GPS Spoofers: A Game Theoretic Countermeasure for Wireless Drones", extended abstract, in *Proc. IEEE CNS 2019*, Washington, D.C. USA, June 2019.

Chapter 2

Drones in Distress: A Game-Theoretic Countermeasure for Protecting UAVs Against GPS Spoofing

2.1 Background, Related Works, and Contributions

As we discussed in Chapter 1, unmanned aerial vehicles (UAVs), popularly known as drones, have been recently adopted in many Internet of Things (IoT) systems to provide services such as telecommunications, delivery, surveillance, and medical services [13, 86–88]. Due to their ability to hover and their high-mobility capability without being restricted to specific routes, UAVs can provide services in hard-to-reach locations such as natural disaster sites. Considering their ease of deployment, UAVs can play a major rule in time-critical systems [19] and to provide urgent Internet and communication services when necessary [89].

However, the widespread use of UAVs in different applications exposes them to a plethora of security threats that include cyber, physical, and cyber-physical attacks [90]. Examples include cyber attacks such as, false data injection and physical attacks such as targeting the UAVs using firearms or hunting rifles. Cyber-physical attacks, on the other hand, represent a category of sophisticated attacks that aim at causing both cyber and physical damage to the UAV such as GPS spoofing, GPS jamming, hijacking the connection between a UAV and its controller, and thwarting delivery drones.

Among these attacks, GPS spoofing is seen as one of the most imminent threats as it is practical and can be easily performed against UAVs [91]. In GPS spoofing attacks, an attacker transmits fake GPS signals to a UAV with slightly higher power than the authentic GPS signals, so as to mislead the UAV into thinking it is in another location. Hence, the attacker can use this technique to send the UAV to another, predetermined, location where it can be captured, thus executing a capture via GPS spoofing attack [92]. The authors in [92] discussed two types of GPS spoofing attacks known

as covert and overt attacks. In a covert attack, the attacker wants to avoid triggering some spoofing detection techniques such as jamming-to-noise ratio and frequency unlock monitoring within the GPS receiver. This requires the attacker to be capable of accurately monitoring the target UAV and to transmit its spoofing signals with specific powers and frequencies. The attacker may also be forced to limit the changes it can impose on a UAV. In contrast, in an overt attack, the attacker can impose any location on the UAV with the risk of being detected.

2.1.1 Related Works

Different techniques have been proposed in literature to defend against GPS spoofing attacks with a focus on attack detection [93–97]. In [93], different techniques are discussed that can enable a UAV's receiver to detect the spoofing attacks. This includes allowing the receiver to observe the received signal strength and compare it to the expected signal strength over time. It can also monitor the identification codes of GPS satellites or keep checking the time intervals to see if they are constant. While these techniques can help to detect basic attackers, they fail against sophisticated attacks in which the attacker monitors the target object accurately [94]. In [95], the authors proposed a method to detect GPS spoofing attacks by using two GPS receivers and checking their cross-correlations. This method was tested against several spoofing attacks and was shown to successfully detect attacks, however, it has low probability of differentiating spoofed from authentic GPS signals and cannot detect the spoofing when the signals are weak.

Meanwhile, the authors in [98] introduced a spoofing-free (SPREE) GPS receiver that is capable of detecting various types of spoofing attacks. The basic idea behind the spoofing-free receiver is to track not only the GPS signal that results in the strongest correlation but also to track weaker correlation peaks for the same GPS satellite. This was achieved by allocating more than one channel to the same satellite, which requires no hardware modifications to commercial GPS receivers, but requires changing the way in which channels are allocated within the receiver. SPREE was shown to detect spoofing signals that impose a location more than 1 km away from the true location. The value drops to 400 meters in case of covert attacks. However, if the attacker offsets the true locations by less than these limiting distances, SPREE will not be able to detect the attack. The distances at which SPREE is effective are, in fact, too large to be used with real-world UAVs given the average trip lengths of UAVs. For example, if a UAV is traveling for 1 km, there should not be a sudden change of 400 m in its location. This will immediately indicate an unrealistic change in the location. In practice, a spoofer performing a covert attack will need to offset the true location by much lower values, which will not be detected by SPREE. Another limitation of using SPREE, is the assumption that an attacker cannot precisely determine the GPS receiver's location so that it will not be able to completely cancel out the authentic GPS signals. This assumption becomes questionable in case of UAVs as it was shown in [92] that a spoofer can accurately monitor and track the target UAV and, hence, predict its trajectory. In this case, the attacker might be able to completely spoof the authentic GPS signals making SPREE unable to determine the real locations.

Other techniques to thwart GPS spoofing such as receiver autonomous integrity monitoring, signal

to interference ratio, Doppler shift detection are discussed in [96]. However, all of these techniques can be avoided by highly capable adversaries that can carefully generate GPS counterfeit signals to avoid triggering these detection schemes. In [99], automatic gain control is used within the GPS receiver to detect and flag potential spoofing attacks within a low computational complexity framework. Finally, the work in [97] proposed a technique that allows UAVs to detect GPS spoofers by using an independent ground infrastructure that continuously analyzes the contents and times of arrival of the estimated UAV positions. The proposed technique was shown to be effective in detecting the spoofing attacks in less than two seconds and to determine the spoofer's location after 15 minutes of monitoring time, with an accuracy of up 150 meters.

Other works in literature have studied the use of multiple receivers to detect GPS spoofing attacks [100–102]. In [100], the authors demonstrated the ability of using a dual antenna receiver in detecting GPS spoofing attacks. Their proposed technique depends on observing the carrier differences between the different antennas referenced to the same oscillator. Under the proposed configuration, an attacker will need to use an additional transmitting antenna for every additional receiver antenna which complicates the attacker's mission. In [101], multiple independent GPS receivers were used to detect GPS spoofing attacks. The proposed technique depends on fixing the distances between the receivers and then measuring the distances between the receivers' reported locations. Under authentic GPS signals, the measured distances will be similar to previously fixed distances. However, under a GPS spoofing attack, these distances will be very close to zero, as all the receivers are spoofed with the same fake location. Finally, in [102], multiple receivers are used to authenticate the GPS signals based on the correlation with the military GPS signal, without the need to decrypt it. One GPS receiver uses the other receivers, referred to as cross-check receivers, to determine whether its GPS signals are authentic. The proposed technique was shown to be effective even when the cross-check receivers are spoofed with some probability. The technique was tested with stationary and moving GPS receivers and was shown to effectively detect the spoofing attacks.

However, one limitation of these existing GPS spoofing detection techniques, i.e. [93–97, 99–102], is that they do not provide an approach to determine the real location of the UAV, after detecting the attack. Thus, if a UAV is attacked while following a route towards a specific destination, the best it can do is to recognize the attack and to stop using the altered GPS signals. However, the UAV will not be able to determine its real location, and, hence, it will not resume its motion towards the specified destination. Indeed, these prior works are mostly focused on detection techniques and do not provide any attack mitigation or defense mechanisms (beyond discarding GPS signals altogether).

2.1.2 Contributions

The main contribution of this chapter is, thus, a general framework for UAVs to mitigate the effect of capture attacks via GPS spoofing. Unlike the prior works [93–97, 99], our framework can both allow the UAVs to detect the GPS spoofing attacks and to determine their real locations. This

will enable the UAVs to avoid being captured and to resume their previous routes and fulfill their missions.

In the proposed framework, we use system dynamics to model a UAV's motion between its origin and destination. For this model, we derive the optimal UAV's controller that allows UAV to travel on the shortest path between any given two locations. This model also captures the effect of a GPS spoofer's on the UAV's traveling path. In particular, we analytically derive the optimal location, under covert attack, that a GPS spoofer can impose on a UAV to lead it to the attacker's desired destination where it can be physically captured. We, then, introduce a defense mechanism built on the technique of cooperative localization [103], which enables a UAV, traveling within a group of proximate UAVs, to determine its location using the real locations of neighboring UAVs and their relative distances. The mechanism also allows the identification of which UAV is being attacked.

Subsequently, we model the interactions between a GPS spoofer and a group of UAVs using game theory [38]. In particular, we formulate a dynamic Stackelberg game in which the drone operator is the leader that selects its strategy first, and then, the spoofer responds by selecting its strategy. A strategy, here, represents a set of actions taken over all the time steps. We, then, propose an efficient technique to solve the formulated dynamic Stackelberg game. Using this technique, we analytically derive the Stackelberg strategies for the game. Finally, through simulations, we show that drone operator can effectively use the proposed defense mechanism to protect the UAVs from being captured and minimize the attacker's effect on the UAVs' optimal routes.

In summary, our contributions include

- We propose a general framework, using realistic system dynamics, to model a UAV's traveling path between any two locations. This model takes into account the effect of a possible GPS spoofer on the UAV's traveling path.
- We analytically derive the attacker's optimal imposed location, on a UAV. This imposed location ensures that attack remains covert while maximizing the attacker's benefit from imposing a different location on the UAV.
- We propose a new defense mechanism built on the technique of cooperative localization to help UAVs to determine their real location, under GPS spoofing attack, using neighbor UAVs' real locations and their relevant distances.
- We then formulate a dynamic Stackelberg game to model the interaction between the drone operator and the GPS spoofer. This game formulation allows the drone operator to effectively use the proposed defense mechanism.
- We introduce a novel computationally-efficient approach to solve the formulated game and we analytically derive the Stackelberg equilibrium strategies for the game.
- We show, through simulations, that the derived Stackelberg strategies outperform other strategy selection techniques by reducing the UAVs possibility of being captured. The defense

mechanism is also shown to mitigate the effects of GPS spoofing attacks on the UAVs' deflections from their planned routes.

The rest of this chapter is organized as follows. The UAV's system dynamics model is presented in Section 2.2. The attacker's model and the optimal imposed locations are derived in Section 2.3. The proposed defense mechanism and the Stackelberg dynamic game with its equilibrium solutions are formulated in Section 2.4. Numerical results are presented and analyzed in Section 2.5. Finally, conclusions are drawn in Section 2.6.

2.2 System Model

Consider a set $\mathscr{N}$ of N UAVs performing a common mission, e.g., a drone delivery system responsible for delivering goods within a certain geographic area. Each UAV is typically equipped with a GPS receiver, a means of wireless communication, and other application-specific sensors. As it was shown in [104], a GPS spoofing attack cannot affect the altitude of UAVs and, thus, we use a two-dimensional (2D) coordination system to specify their locations. Let the location of UAV i at time t be $\boldsymbol{x}_i(t) = [x_i(t), y_i(t)]^T$, where $i \in \mathscr{N}$. Similarly, the source locations, O_i, will be given as $\boldsymbol{x}_{O_i} = [x_{O_i}, y_{O_i}]^T$, and each destination's location is $\boldsymbol{x}_{d_i} = [x_{d_i}, y_{d_i}]^T$. Destinations are assumed to be fixed and not time dependent. The goal of each UAV is to minimize the transportation cost and, hence, it chooses the shortest path from its source O_i towards its destination d_i.

In our model, we consider an adversary located along the traveling paths of the UAVs whose goal is to spoof the GPS signals of any of the UAVs in order to send it to another location where it can be captured. We consider a capable GPS spoofer that can spoof from a distance (in the order of hundreds of meters) without the need to be co-located with the UAV's GPS receiver [92].

Prior to developing the threat model, we first use system dynamics to model the UAV's motion between its source location, the origin, and destination. This model is needed to better understand the impact of the attack on the UAV's mobility. In order for each UAV to minimize its travel time, each UAV will follow the shortest path between its current location and its destination which essentially consists of the straight line connecting the two locations in 2D space. Let the location of UAV i after a time duration Δ be $\boldsymbol{x}_i(t+\Delta)$. Let $\boldsymbol{v}_i(t) = [v_{x_i}(t), v_{y_i}(t)]^T$ be the UAV's velocity at the beginning of time step t. The UAV's velocity at the end of the time step can be represented in terms of the UAV's acceleration as follows:

$$\boldsymbol{v}_i(t+\Delta) = \boldsymbol{v}_i(t) + \Delta \cdot \boldsymbol{g}_i(t), \tag{2.1}$$

where $\boldsymbol{g}_i(t) = [g_{x_i}(t), g_{y_i}(t)]^T$ is the acceleration of UAV i during the time step starting at t and T is the duration of the time step.

The next location can then be represented using both the velocity and acceleration as follows:

$$\boldsymbol{x}_i(t+\Delta) = \boldsymbol{x}_i(t) + \Delta \cdot \boldsymbol{v}_i(t) + \frac{\Delta^2}{2} \cdot \boldsymbol{g}_i(t). \tag{2.2}$$

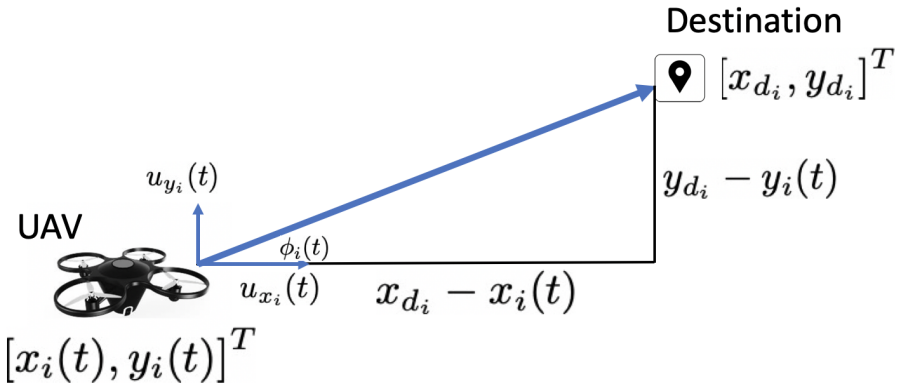


Figure 2.1: UAV traveling model between two locations.

Since the force needed to move the UAV between two locations is proportional to both the UAV's acceleration and weight, and the UAV's weight is constant, this force can then be related directly to the acceleration. Let $\boldsymbol{u}_i(t) = [u_{x_i}(t), u_{y_i}(t)]^T$ be the force needed to move the UAV between any two locations. This force will be proportional to the distance difference between the current and the next locations, i.e., the UAV must accelerate more in the direction with a larger distance difference. Figure 2.1 shows the UAV's traveling model with the force components in each direction. In Figure 2.1, the distance difference in the x direction is more than the difference in the y direction, and, hence, the force component in the x direction will be greater than that of the y direction.

In order for each UAV to minimize its travel time, each UAV will need to find the optimal force to move between any two locations given that the maximum allowable force is $u^{\max}$. Let ϕ_i be the angle between the UAV movement route and the positive x direction which can be calculated as:

$$\phi_i(t) = \arctan\left(\frac{y_{d_i} - y_i(t)}{x_{d_i} - x_i(t)}\right) = \arctan(\gamma). \tag{2.3}$$

The force components in both x and y directions can then be given by:

$$\begin{aligned} u^*_{x_i}(t) &= u^{\max} \cdot \cos(\phi_i(t)), \\ u^*_{y_i}(t) &= u^{\max} \cdot \sin(\phi_i(t)). \end{aligned} \tag{2.4}$$

These values represent the optimal controller, i.e., the optimal force that each UAV can use to move between any two locations. Note that, if $\gamma > 1$ then $\sin(\phi_i(t)) > \cos(\phi_i(t))$ and in this case $u^*_{y_i}(t) > u^*_{x_i}(t)$, and vice verse. Substitute the optimal controller into (2.2), the UAV's next location can then be given, in terms of the optimal controller, as:

$$\begin{aligned} x_i(t+\Delta) &= x_i(t) + \Delta \cdot v_{x_i}(t) + c \cdot \Delta^2 \cdot u^*_{x_i}(t), \\ y_i(t+\Delta) &= y_i(t) + \Delta \cdot v_{y_i}(t) + c \cdot \Delta^2 \cdot u^*_{y_i}(t), \end{aligned} \tag{2.5}$$

where $c = \frac{m}{2}$ is a constant and m is the UAV's weight. Next we will discuss the effect of a GPS spoofer on the UAV's route by deriving both the optimal locations for an attacker to impose on a UAV and the manipulated routes under attack.

2.3 UAV Traveling Model under GPS Spoofing Attack

In our model, the GPS spoofer seeks to take control of the UAV's GPS antennas and then transmit tailored GPS signals to convince the UAV's navigation system that it is in a different location. The spoofer can perform either an overt attack or a covert attack. In the overt attack, the spoofer makes no effort to hide its attack, it transmits its fake signals with higher power than the authentic GPS signals. The covert attack, on the other hand, requires an accurate tracking of the target UAV and the transmission of fake GPS signals with specific power requirements to avoid being immediately detected by the UAV. Here, we consider a spoofer that wants to keep its attack covert by adjusting the transmission power of the counterfeit GPS signals to avoid being detected. Practical values for such power requirements can be found in [92]. In addition, the attacker will be limited to the changes it can impose on the UAV's location, each time, so that these imposed locations do not trigger the fault detectors within the UAV [104]. The distance between the current and imposed location is known as the instance drifted distance [105].

Let e_{max} be the instance drifted distance that limits the spoofer's attack. Let $\hat{\boldsymbol{x}}_i(t) = [\hat{x}_i(t), \hat{y}_i(t)]^T$ be the attacker's imposed location on UAV i. Let $\boldsymbol{E}_i(t) = [e_{x_i}(t), e_{y_i}(t)]^T$ be a vector whose individual elements represent the distance difference between the UAV's actual location and the attacker's imposed location. Then, we must have:

$$\|\boldsymbol{E}_i(t)\|_2 = \|\boldsymbol{x}_i(t) - \hat{\boldsymbol{x}}_i(t)\|_2 \leq e_{\text{max}}, \tag{2.6}$$

which represents a circle of radius e_{max} around the UAV's current location.

Note that, imposing an attacker-desired location on a UAV does not actually change the UAV's location, instead, it changes the UAV's belief about its location. This means that the UAV will still be in its real location but its navigation system will believe that it is in a different location. The UAV will then need to find a new optimal controller, i.e., new force components to move from its imposed location to its final destination. In this case, there will be two routes as shown in Figure 2.2. Here, the upper route is the fake route which the UAV believes it is traveling on. This route starts from the attacker's imposed location towards the UAV's real destination. However, the UAV will actually travel on the lower path towards the attacker's desired destination.

Let $\boldsymbol{x}^a_{d_i} = [x^a_{d_i}, y^a_{d_i}]^T$ be the attacker's desired destination for UAV i. The attacker's imposed location, $\hat{\boldsymbol{x}}_i(t)$, at each time step, needs to be calculated in order for the UAV to move towards $\boldsymbol{x}^a_{d_i}$. This can be achieved by satisfying the condition in the following lemma.

Lemma 1. *The attacker's imposed location needs to satisfy $\hat{\gamma} = \gamma^a$, where $\hat{\gamma} = \left(\frac{y_{d_i} - \hat{y}_i(t)}{x_{d_i} - \hat{x}_i(t)}\right)$ and $\gamma^a = \left(\frac{y^a_{d_i} - y_i(t)}{x^a_{d_i} - x_i(t)}\right)$.*

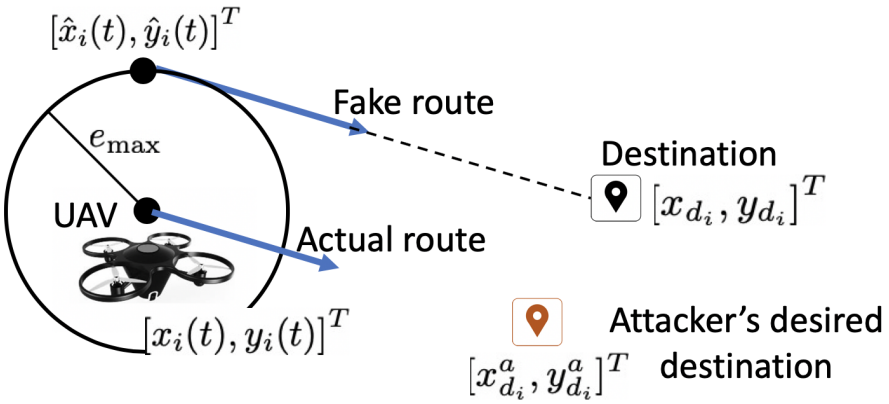


Figure 2.2: UAV actual and fake routes.

Proof. Under an attack, the UAV will believe it is traveling from the attacker's imposed location, on the fake route in Figure 2.2. The attacker should then select this imposed location $[\hat{x}_i(t), \hat{y}_i(t)]$ such that the actual route leads the UAV to the attacker's desired destination. Since, the UAV travels on the shortest path between any two locations, the actual route will represent a straight line that is parallel to the route which the UAV believes it is traveling on, i.e., the fake route.

From Figure 2.2, the fake route, can be defined by the two points $[\hat{x}_i(t), \hat{y}_i(t)]$ and $[x_{d_i}, y_{d_i}]$. Similarly, the actual route, can be defined by the two points $[x_i(t), y_i(t)]$ and $[x^a_{d_i}, y^a_{d_i}]$. For these routes to be parallel, the slopes of both routes need to be equal, i.e., $\hat{\gamma} = \gamma^a$. □

Note that, under overt attack, according to Lemma 1, the attacker can impose, theoretically, any location on the UAV that will lead the UAV to follow a path towards the attacker's desired destination. However, under a covert attack, the imposed location will be limited by (2.6). This imposes constraints on the attacker when choosing the imposed location as there may be multiple or no points inside the circle, in (2.6), that satisfy Lemma 1. When no such points exist, the best option for the attacker is to force the UAV to move in a direction as close as possible to the line connecting the real location and the attacker's desired destination, i.e., a direction that minimizes the difference $|\hat{\gamma} - \gamma^a|$ in Lemma 1. If there are more than one point that satisfy Lemma 1, then the best for the attacker is to choose the furthest point from the UAV's real destination as this gives more flexibility for the attacker in changing the imposed locations in the future time steps. Thus the attacker's optimal imposed location can be given by the solution of the following constrained-optimization problem.

$$\begin{aligned} & \min_{\boldsymbol{x}^a_i(t)} |\hat{\gamma} - \gamma^a|, \\ \text{s. t.} \quad & x^a_i(t) = \underset{\boldsymbol{x}^a_i(t)}{\operatorname{argmax}} \left\| \boldsymbol{x}_{d_i} - \boldsymbol{x}^a_i(t) \right\|_2, \\ & \left\| \boldsymbol{E}_i(t) \right\|_2 = \left\| \boldsymbol{x}_i(t) - \boldsymbol{x}^a_i(t) \right\|_2 \leq e_{\max}. \end{aligned} \tag{2.7}$$

In the following theorem, we analytically derive the attacker's imposed location, under covert attack.

Theorem 1. *Let* $s_i(t) \triangleq (d_i(t)+a_i(t)+l_i)(d_i(t)+a_i(t)-l_i)(d_i(t)-a_i(t)+l_i)(a_i(t)+l_i-d_i(t))$ *where*

$$d_i(t) \triangleq \sqrt{(x_i(t)-x_{d_i})^2+(y_i(t)-y_{d_i})^2}, \tag{2.8}$$

$$a_i(t) \triangleq \sqrt{(x_i(t)-x^a_{d_i})^2+(y_i(t)-y^a_{d_i})^2}, \tag{2.9}$$

$$l_i \triangleq \sqrt{(x^a_{d_i}-x_{d_i})^2+(y^a_{d_i}-y_{d_i})^2}. \tag{2.10}$$

Then, the attacker's imposed location is the solution for the following set of equations:

$$\hat{y}_i(t)-y_{d_i} = \frac{\hat{y}_i(t)-y_{d_i}}{\hat{x}_i(t)-x_{d_i}}(\hat{x}_i(t)-x_{d_i}) \tag{2.11}$$

$$(x_i(t)-\hat{x}_i(t))^2+(y_i(t)-\hat{y}_i(t))^2 = e^2_{max}, \tag{2.12}$$

if $e_{max} > \frac{1}{2a_i(t)}\sqrt{s_i(t)}$*, or the following set of equations:*

$$(\hat{x}_i(t)-\hat{x}_{d_i}(t))^2+(\hat{x}_i(t)-\hat{y}_{d_i}(t))^2 = e^2_{max}+d^2(t) \tag{2.13}$$

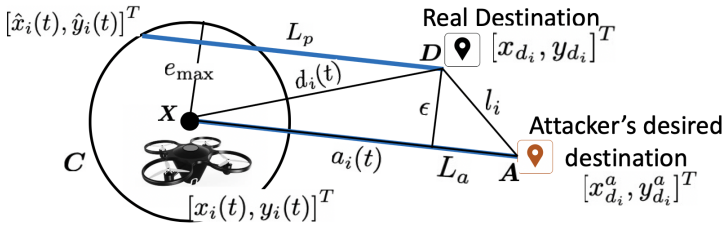
$$(x_i(t)-\hat{x}_i(t))^2+(y_i(t)-\hat{y}_i(t))^2 = e^2_{max} \tag{2.14}$$

if $e_{max} \leq \frac{1}{2a_i(t)}\sqrt{s_i(t)}$.

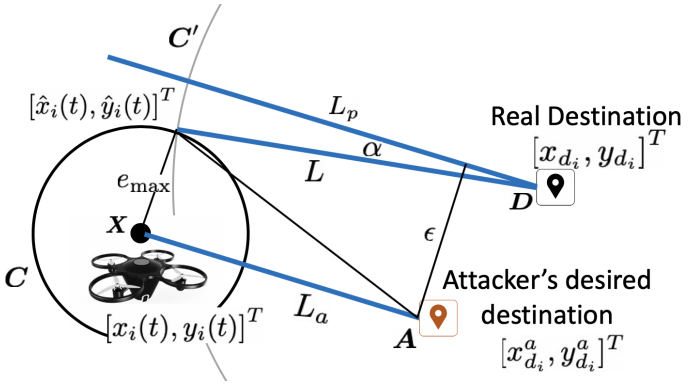
Proof. In Figure 2.3, we use a geometrical representation for the problem to help clarify our proof. Let L_a be the line connecting the UAV's real location to the attacker's desired destination. This line represents the attacker's ideal route for the UAV to travel on. Let L_p be the line parallel to L_a and passes through the UAV's real destination and ε be the distance between these two lines. There are then two cases for line L_p.

Case 1: if the line L_p touches or intersects with the circle, formed by the constraint, then the point or the set of points of the intersection will represent a solution for the first objective function. In this case, the difference $|\gamma^a-\hat{\gamma}|$ will be 0, which is the minimum possible value. This case will happen if $e_{\max} \geq \varepsilon$. Thus, next, we find the value of ε using the known values of $d_i(t)$, $a_i(t)$, and l_i. To this end, we find the area of triangle ADX, $s_{ADX}(t)$, using two ways: 1) $s_{ADX}(t) = \frac{\varepsilon \cdot a_i(t)}{2}$ and 2) $s_{ADX}(t) = \sqrt{s_i(t)}/4$, using Heron's formula [106]. Hence, we will have $\varepsilon = \frac{1}{2a_i(t)}\sqrt{s_i(t)}$. Therefore, if $e_{\max} \geq \frac{1}{2a_i(t)}\sqrt{s_i(t)}$, then the attacker's optimal choice is the intersection of L_p with circle C where L_p can be given by (2.11) and C can be given by (2.12).

As this intersection may consist of more than one point, let $\mathscr{S}$ represents the solution set so far. The optimal solution for the problem in (2.7) can then be found by solving the second optimization problem in the first constraint, i.e., $x^a_i(t) = \text{argmax}_{\boldsymbol{x}^a_i(t)} \|\boldsymbol{x}_{d_i}-\boldsymbol{x}^a_i(t)\|_2$ [107]. If $\mathscr{S}$ has only one point, then this point will be the solution to the first constraint, which is a point on the circle



(a) Case 1



(b) Case 2

Figure 2.3: Determining the attacker's imposed location.

perimeter. However, if $\mathscr{S}$ has multiple points, the solution will be the point on the circle's perimeter on the opposite side from the UAV's real destination. In either cases, this solution point will, then, be the attacker's imposed location.

Case 2: This case represents a more general case when L_p does not intersect the circle formed by the constraint, i.e., $e_{\max} > \frac{1}{2a_i(t)}\sqrt{s_i(t)}$. The solution to the objective function, in this case, will not lie on L_p, instead it will lie on another line L that passes through the UAV's real destination and intersects the circle at one point. This line L should make the smallest angle α with the line L_p, and, hence, it minimizes the objective function. Thus, the optimal imposed location by the attacker, in this case, is the intersection of circle C and a circle C' with a radius of $\sqrt{e^2_{\max} + d^2_i(t)}$ with its center at the actual desired destination of the UAV. Circle C can be represented formally as in (2.13). Note that the two circles will always intersect in two points, however, only one of them will minimize the objective function and, hence, this point will also be a solution for the maximization problem in the first constraint. □

In the second case of Theorem 1, the attacker's imposed location will not lead the UAV directly to the attacker's desired destination. Consequently, the attacker might need to impose more than one location on the UAV along its perceived route. Each new imposed location can be calculated from Theorem 1 with respect to the UAV's new location. Once the attacker can lead the UAV towards its desired destination, the imposed location, according to Theorem 1, will be the furthest point in the circle that maintains the same direction. Next, we study the UAV's manipulated route due to

the attacker's imposed location.

Consider the UAV's next location under an attack. Similar to (2.4), the UAV needs to compute the force components in both directions. The UAV thinks it is at the attacker's imposed location, $\hat{\boldsymbol{x}}_i(t)$, so it calculates the required force to move from $\hat{\boldsymbol{x}}_i(t)$ to its desired destination $\boldsymbol{x}_{d_i}$. Let ϕ_i^a be the angle between the x direction and the line connecting the UAV's imposed location to its real destination. The value of ϕ_i^a can then be given as:

$$\phi^a(t) = \arctan\left(\frac{y_{d_i} - \hat{y}_i(t)}{x_{d_i} - \hat{x}_i(t)}\right) = \arctan(\hat{\gamma}). \tag{2.15}$$

Therefore, the force components in both x and y directions can then be given as:

$$\begin{aligned} u_{x_i}^a(t) &= u^{\max} \cdot \cos(\phi_i^a(t)), \\ u_{y_i}^a(t) &= u^{\max} \cdot \sin(\phi_i^a(t)). \end{aligned} \tag{2.16}$$

The UAV will then use the optimal controller in (2.16) to move towards its real destination. However, the UAV will actually move from its real location not its perceived location as shown in Figure 2.2. Let $\boldsymbol{x}_i^a(t+1) = [x_i^a(t+1), y_i^a(t+1)]^T$ be the UAV's next location under attack. It can then be given, according to (2.5), as:

$$\begin{aligned} x_i^a(t+1) &= x_i(t) + T \cdot v_{x_i}(t) + c \cdot \Delta^2 \cdot u_{x_i}^a(t), \\ y_i^a(t+1) &= y_i(t) + T \cdot v_{y_i}(t) + c \cdot \Delta^2 \cdot u_{y_i}^a(t). \end{aligned} \tag{2.17}$$

Note that, following the route calculated by (2.17) may not guarantee the attacker to eventually lead the UAV to the attacker's desired destination. Achieving this depends on multiple parameters such as the UAV's current location and the locations of its real destination and the attacker's desired destination. In general, the attacker should choose its desired destination to satisfy the following condition.

Proposition 1. *Under a covert attack, the attacker's desired destination should be located on the same side as the UAV's real destination, in terms of the direction with the largest difference between the UAV's current location and its real destination.*

Proof. We start by making two assumptions. First, the UAV is considered to reach its destination if it is within a distance $e_{\max}$ from its real destination. We also assume that the distance between the UAV's current location and its real destination is greater than $e_{\max}$, i.e., the UAV did not reach its destination yet.

According to (2.5) and (2.17), the UAV travels towards its real destination from its perceived location. Therefore, in order for the UAV to reach a destination in the opposite side from its real destination, the UAV needs to change its direction in the direction with the longest difference from the UAV's current location. Assume without loss of generality that the difference in the x direction,

between the UAV's perceived location and its real destination, is bigger than the difference in the y direction. Then for the UAV, to change its x direction, the value of $u^a_{x_i}(t)$ needs to flips its sign in (2.17). Comparing the optimal controller, in x direction, in both (2.16) and (2.4) and assuming they have opposite signs:

$$\cos(\arctan\Big(\frac{y_{d_i}-\hat{y}_i(t)}{x_{d_i}-\hat{x}_i(t)}\Big)) = -\cos(\arctan\Big(\frac{y_{d_i}-y_i(t)}{x_{d_i}-x_i(t)}\Big))$$

For this condition to hold, $x_{d_i}-\hat{x}_i(t)$ needs to have a different sign from $x_{d_i}-x_i(t)$. i.e., $\hat{x}_i(t)-x_i(t) = (x_{d_i}-x_i(t))+(x_{d_i}-\hat{x}_i(t))$

However, under a covert attack, the imposed location is limited by (2.6), i.e., $|\hat{x}_i(t)-x_i(t)| \leq e_{\text{max}}|$. Since, it is assumed that the UAV's real location is more than e_{max} away from its real destination, i.e, $x_{d_i}-x_i(t) \geq e_{\text{max}}$, then the condition for changing the direction cannot hold. In this case, the attacker cannot impose a location that forces the UAV to change its x direction. Therefore, the attacker's desired destination cannot be in the opposite x direction from the UAV's real destination. □

Next, we will discuss the defense mechanism against the considered GPS spoofing attack.

2.4 GPS Spoofing Countermeasure

2.4.1 Defense Mechanism for Mitigating Spoofing Attacks

We propose a defense mechanism built on the concept of cooperative localization [103] which is a framework that enables a UAV to determine its real location in a 2D coordinate system using the locations of three other UAVs. Each UAV is assumed to have a means of measuring its relative distances to the other, neighboring UAVs by inter-UAV range measurements. In cooperative localization, a UAV chooses any three neighboring UAVs, to update its location, given that the selected UAVs are non-collinear. Following this, the UAV can accurately determine its 2D location. While the cooperative localization mechanism in [103] can help a UAV to determine its location, it was proposed to be used in case of GPS signals loss and cannot be used, directly, in case of GPS spoofing attack due to the different nature of the problem.

Under a covert GPS spoofing attack, a UAV cannot trust its GPS location nor the locations of other UAVs. Choosing a neighboring UAV for the cooperative localization mechanism will involve a risk as this UAV might itself be under attack. To overcome this limitation, we propose a defense mechanism based on the fact that a GPS spoofing attacker can target only one UAV at a time, as discussed earlier. In our proposed mechanism, a UAV will use the locations of four neighboring UAVs, instead of three, to determine its real location by identifying the UAV under attack and eliminating it from the calculations. The proposed mechanism has the same requirements of

cooperative localization, i.e., the UAVs are non-collinear, a UAV can request other UAVs' locations through inter-UAV communications, and each UAV needs to be able to measure its relative distances to its neighboring UAVs.

Due to the fact that determining a 2D location requires only three UAVs, the fourth UAV will be used to check the results as follows. A UAV will calculate its location using all the permutations of three UAVs formed from the selected four UAVs. Let any UAV and its selected four neighbor UAVs represent a set given by $\mathscr{F} = \{F_i\}$, where F_i represent a UAV and $i \in \{1, \ldots, 5\}$. Assume UAV F_1 wants to calculate its location, let its location calculated from the GPS signals be $\tilde{\boldsymbol{x}}_1$. The UAV, F_1, cannot determine at this point if this location is real or a spoofed location. The UAV will then calculate its location four more times using all the groups formed of three UAVs out of the selected four UAVs. For example, $\tilde{\boldsymbol{x}}_2$ can be calculated using UAVs F_2, F_3, and F_4, $\tilde{\boldsymbol{x}}_3$ can be calculated using UAVs F_2, F_3, and F_5, and so on for $\tilde{\boldsymbol{x}}_4$ and $\tilde{\boldsymbol{x}}_5$.

The UAV can then determine its real location according to the following cases:

- If there is no attack, the value of $\tilde{\boldsymbol{x}}_1$ will equal all the other values, i.e., $\tilde{\boldsymbol{x}}_i, i = 2, \ldots, 5$, will all be the same.
- If UAV F_1 is under attack, then all the values $\tilde{\boldsymbol{x}}_i, i = 2, \ldots, 5$, will be equal but their value will not equal $\tilde{\boldsymbol{x}}_1$. In this case, the real location of UAV F_1 is the value calculated from its neighboring UAVs.
- If another UAV, rather than UAV F_1, is under attack, then the value of $\tilde{\boldsymbol{x}}_1$ will equal only one of the four other values. The other three values will be the same and the UAV that contributed to calculating these values will be the one under attack.

Note that the technique used in [102] was shown to require four different cross-check receivers, to detect the GPS spoofing attack, when $15\% - 25\%$ of the cross-check receivers are unreliable. Comparing these findings to our proposed defense mechanism, the same number of UAVs, i.e., four cross-check receivers, will be required to detect a single attack, i.e., 25% of the cross-check receivers are unreliable. However, our defense mechanism can not only detect attacks but it can also determine the real locations.

We summarize our defense mechanism steps in the flow chart shown in Figure 2.4. Note that, following our approach in Figure 2.4, a UAV can determine its real location and identify which UAV is under attack. However, the UAV under attack will have to also execute the same procedure to determine its real location. One approach is to allow all the UAVs to continuously use the proposed defense mechanism along their travel paths. However, this might be challenging to do in long routes as the energy consumption due to exchanging communication messages, measuring distances to other UAVs, and calculating the locations will be significant compared to the UAVs' limited power.

Given that a GPS spoofer can target only one UAV at a time, we next propose a new approach to regulate the use of our proposed defense mechanism among the UAVs through studying the

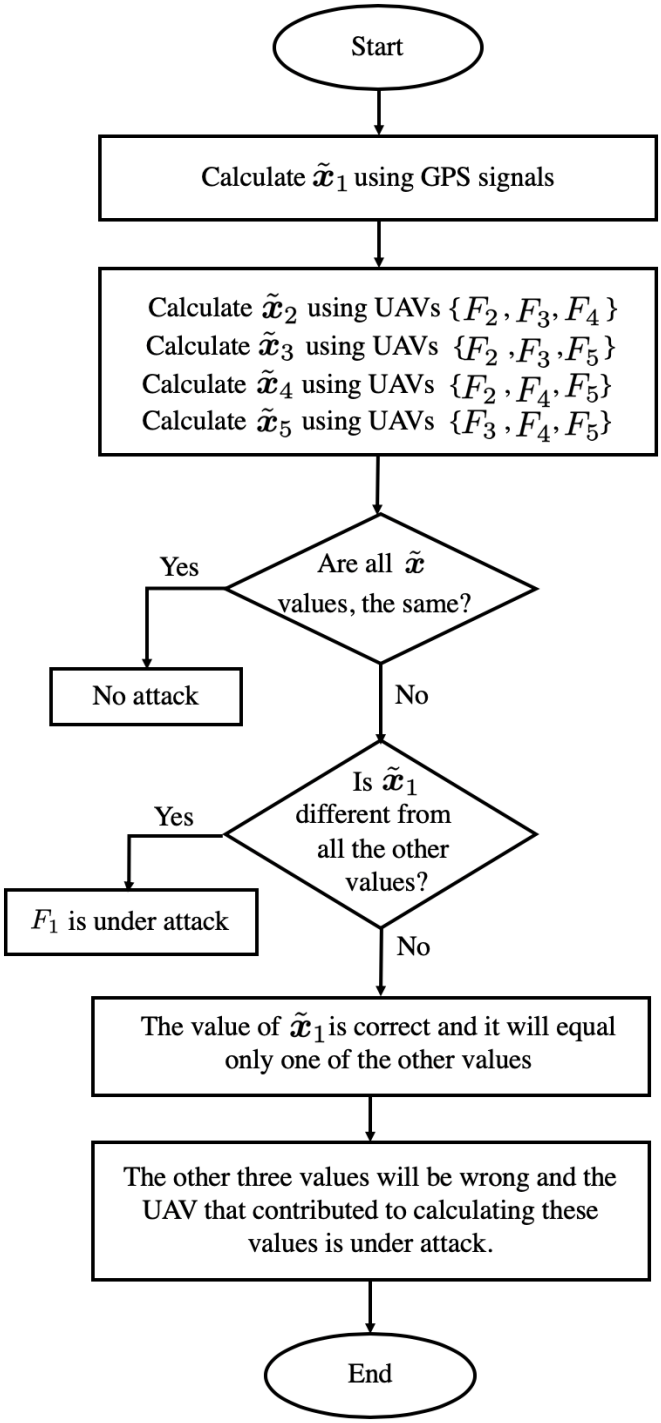


Figure 2.4: Flowchart for the defense mechanism.

interactions between the GPS spoofer and a group of UAVs, managed by their operator. The drone operator wants to regulate the use of the defense mechanism, in an energy-efficient manner, by determining when each UAV can use it along its travel path, to avoid being captured. On the other hand, the GPS spoofer wants to take control of the UAVs to send them to other destinations where they can be captured. In doing so, both the operator and the spoofer will be affected by each others' actions. Therefore, we propose to use game theory [108] to model these interactions. In our game, the drone operator is the defender and the GPS spoofer is the attacker. As the attack and defense mechanisms are applied along each UAV's travel paths, the game will be time-depended, and, hence, a dynamic game model is appropriate.

2.4.2 Dynamic Stackelberg Game Formulation

We formulate a dynamic game in which every player, i.e., the attacker and the defender chooses its actions at every time step. Since the spoofer needs to monitor the targeted UAV to generate tailored spoofing signals, the spoofer will be able to attack only one UAV at any given time step. At each time, the attacker can choose one action out of set $\mathscr{Z}^a$ which represents the choice of one UAV to attack. Similarly, the drone operator can choose one UAV to use the defense mechanism defined in Section 2.4 to update its location, given that the spoofer can attack only one UAV at a time. Let $\mathscr{Z}^d$ be the set from which the defender is choosing its actions, i.e., a UAV to apply the defense mechanism.

Note that, in the proposed defense mechanism, each UAV needs the locations of four neighboring UAVs to determine its location. Therefore, each five UAVs can be seen as a separate group in which one UAV is applying the defense mechanism using the locations of the other four UAVs. Without loss of generality, we assume the total number of UAVs is a multiple of five, with each five, closely traveling, UAVs forming a group. For larger systems, each group of five UAVs can form and coordinate together and, hence, our approach can be applied locally to each group. Hence, hereinafter, we consider a game in which the drone operator is protecting only one group, because the solution can be easily extended to the case of multiple groups of UAVs.

The actions of each player, when taken at a time step, will affect the next locations of the UAVs. If a UAV is applying the defense mechanism, then it can accurately determine its location whether there is an attack or not. On the other hand, if a UAV is dependent on the GPS signals, it will be affected by the attacker's actions and its next location will depend on whether it is attacked or not. Here, we assume that each spoofing attack is successful in that the attacker will gain control of the UAV's GPS receivers and impose its desired location on the UAV's GPS. Let $z_i^a(t), i = 1, \ldots, 5$ be a variable indicating whether the attacker has chosen to attack UAV i, where $z_i^a(t) = 1$ means the UAV i is being attacked at time step t, and $z_i^a(t) = 0$ otherwise. Similarly, let $z_i^d(t), i = 1, \ldots, 5$, be a variable indicating whether the defender has chosen to protect UAV i, where $z_i^d(t) = 1$ means the UAV i is applying, at time step t, the defense mechanism, i.e., being protected, and $z_i^d(t) = 0$ otherwise. Each UAV's next location can then be given by:

$$\boldsymbol{x}_i(t + \Delta, z_i^d(t), z_i^a(t)) = z_i^d(t) \cdot \boldsymbol{x}_i(t) + \big(1 - z_i^d(t)\big)\big[z_i^a(t) \cdot$$

$$\boldsymbol{x}_i^a(t) + (1 - z_i^a(t)) \cdot \boldsymbol{x}_i(t)\big], \tag{2.18}$$

where $\boldsymbol{x}_i(t)$ and $\boldsymbol{x}_i^a(t)$ are given by (2.5) and (2.17), respectively. Equation (2.18) can be rearranged as:

$$\begin{aligned}\boldsymbol{x}_i(t+\Delta, z_i^d(t), z_i^a(t)) =& \Big(1 - z_i^a(t) + z_i^a(t) \cdot z_i^d(t)\Big) \cdot \boldsymbol{x}_i(t) \\ &+ \Big(z_i^a(t) - z_i^a(t) \cdot z_i^d(t)\Big) \cdot \boldsymbol{x}_i^a(t).\end{aligned} \tag{2.19}$$

From Theorem 1, we can observe that the attacker's imposed location $\boldsymbol{x}_i^a(t)$ can be accurately calculated from the UAV's current location $\boldsymbol{x}_i(t)$, the UAV's real destination, and the attacker's desired destination. However, as the UAV's real destination and the attacker's desired destination are constants, the attacker's imposed location, at any given time step, can be given as a function of the UAV's real location. Therefore, the location in (2.19) can be given as a function in the UAV's current location and both player's actions, i.e., $\boldsymbol{x}_i(t+\Delta, z_i^d(t), z_i^a(t)) = f\big(\boldsymbol{x}_i(t), z_i^d(t), z_i^a(t)\big)$.

Next, we define the outcomes (utilities) for both players due to their interactions. Since the objective for each player is to move each UAV to its own desired destination, each player will take actions to minimize the distance between the current UAV's location and the player's desired destination. Thus, we define the utility function for the attacker, at each time step, as follows:

$$U^a(t, z_i^d(t), z_i^a(t)) = \sum_{i=1}^{5} \left\|\boldsymbol{x}_{d_i}^a - \boldsymbol{x}_i(t+\Delta, z_i^d(t), z_i^a(t))\right\|_2^2. \tag{2.20}$$

Similarly, the defender's utility, at each time step, can be given by:

$$U^d(t, z_i^d(t), z_i^a(t)) = \sum_{i=1}^{5} \left\|\boldsymbol{x}_{d_i} - \boldsymbol{x}_i(t+\Delta, z_i^d(t), z_i^a(t))\right\|_2^2. \tag{2.21}$$

Now, consider the players' actions and utilities over all time steps. Assume the maximum possible number of time steps is τ, which is determined by the maximum time that any UAV can travel based on its fuel or battery. This number is known to the defender but the attacker does not need to know this number. From Proposition 1, the GPS spoofer will not be able to change the UAV's direction and, thus, once a UAV passes beyond the attacker's desired destination, the attacker will no more consider it when choosing its actions. Therefore, the game is considered to end for the attacker when all the UAVs pass beyond the attacker's desired destinations.

Consider the players' strategies which are defined as the players' actions taken at each time step t. Let $\boldsymbol{\beta}^a$ be an attacker's strategy defined by $\boldsymbol{\beta}^a = \{z_i^a(1), \ldots, z_i^a(\tau)\}$, and let $\mathscr{A}$ be the set of all the attacker's possible strategies. Similarly, let $\boldsymbol{\beta}^d$ be a defender's strategy defined by $\boldsymbol{\beta}^d = \{z_i^d(1), \ldots, z_i^a(\tau)\}$, and let $\mathscr{D}$ be the set of all the defender's possible strategies. The attacker's accumulated utility will then be:

$$J^a(\boldsymbol{\beta}^d, \boldsymbol{\beta}^a) = \sum_{t=1}^{\tau} U^a(t\Delta, z_i^d(t), Z_i^a(t))$$

$$= \sum_{t=1}^{\tau} \sum_{i=1}^{5} \left\| \boldsymbol{x}_{d_i}^a - \boldsymbol{x}_i(t\Delta, z_i^d(t), z_i^a(t)) \right\|_2^2 . \tag{2.22}$$

Similarly, the defender's accumulated utility will be given by:

$$\begin{aligned} J^d(\boldsymbol{\beta}^d, \boldsymbol{\beta}^a) &= \sum_{t=1}^{\tau} U^d(t\Delta, z_i^d(t), Z_i^a(t)) \\ &= \sum_{t=1}^{\tau} \sum_{i=1}^{5} \left\| \boldsymbol{x}_{d_i} - \boldsymbol{x}_i(t\Delta, z_i^d(t), z_i^a(t)) \right\|_2^2 . \end{aligned} \tag{2.23}$$

To solve this dynamic game, we propose to use the dynamic Stackelberg game model [109]. In Stackelberg games, one player, the leader, acts first by selecting its strategy and, then, the other player, the follower, can respond by selecting its strategy. In our game formulation, the drone operator will act as the leader as it can choose which UAVs to protect in advance and the attacker can observe this selection and responds by choosing which UAVs to attack.

Now, we can formally formulate a dynamic Stackelberg game Ξ described by the tuple $\left\langle \mathscr{M}, \mathscr{A}, \mathscr{D}, J^a, J^d, \tau \right\rangle$ where $\mathscr{M}$ is the set of the two players: the defender and the attacker, and the rest of the parameters as defined earlier. Based on the utility functions, the game is non-zero sum. This means, every player will try to minimize its utility and the sum of the utilities will not equal zero. Moreover, each player seeks to follow a strategy that minimizes its utility function given the other player's strategy. Next, we study our approach of finding the optimal strategies, for each player, under the formulated game.

2.4.3 Stackelberg Game Solution

The most commonly adopted solution for Stackelberg dynamic games is known as the Stackelberg equilibrium strategy concept [109]. This solution is given by a pair of strategies $(\boldsymbol{\beta}^{a*}, \boldsymbol{\beta}^{d*})$ defined as follows.

Definition 1. *The* Stackelberg equilibrium strategies*, when the defender is the leader, are derived as follow. Let $r : \mathscr{D} \rightarrow \mathscr{A}$ be a mapping between the defender's strategies and the attacker's strategies, such that:*

$$J^a(\boldsymbol{\beta}^d, r(\boldsymbol{\beta}^d)) \leq J^a(\boldsymbol{\beta}^d, \boldsymbol{\beta}^a), \forall \boldsymbol{\beta}^a \in \mathscr{A}, \tag{2.24}$$

and the set:

$$R^a = \{(\boldsymbol{\beta}^d, \boldsymbol{\beta}^a) \in \mathscr{D} \times \mathscr{A} : \boldsymbol{\beta}^a = r(\boldsymbol{\beta}^d), \forall \boldsymbol{\beta}^d \in \mathscr{D}\}, \tag{2.25}$$

is the reaction set for the attacker when the defender is the leader. The Stackelberg equilibrium strategies $(\boldsymbol{\beta}^{a*}, \boldsymbol{\beta}^{d*}) \in R^a$ *of the game should then satisfy:*

$$J^d(\boldsymbol{\beta}^{a*}, \boldsymbol{\beta}^{d*}) \leq J^d(\boldsymbol{\beta}^d, \boldsymbol{\beta}^a), \forall (\boldsymbol{\beta}^d, \boldsymbol{\beta}^a) \in R^a. \tag{2.26}$$

Note that, solving for the Stackelberg equilibrium strategies that satisfy (2.26) depends on the information available for each player, at each time step [110]. According to [110], dynamic games can be solved using open-loop strategies, closed-loop strategies, or feedback strategies. In the formulated game, each player selects a strategy that minimizes its utility which involves taking actions, at each time step to control the UAVs' locations. In doing so, both players can observe the initial locations of the UAVs as well as their subsequent locations up to the current time step. This type of information coincides with the notion of *closed-loop perfect information* [110], and, thus, we use closed-loop Stackelberg strategies to solve the formulated game. Note that the equilibrium strategies should satisfy (2.26) irrespective of the type of the solution.

In our formulated game, the cost functions in (2.22) and (2.23) will ensure the existence of the Stackelberg solution, under closed-loop perfect information [110]. However, this solution might not be unique, as there might be multiple strategies that yield the same utilities for the players. Solving for closed-loop strategies, in general, is challenging, especially when the number of time steps is large. In the formulated game, the number of available actions for each player, at every time step, equals 5 which is the number of the UAVs. As a strategy is a combination of τ different actions, there will be 5^τ different strategies available for each player. The solution follows by calculating the attacker's response for each of 5^τ different defender's strategies, which involves testing all the attacker's 5^τ strategies per a defender's strategy. Finally, the defender selects the pair of strategies that minimize its utility. The complexity of this solution approach will then be $O(5^{2\tau})$, which is exponential in terms of the number of time steps. This, in fact, might not be feasible when the value of τ is large, as is the case in the UAVs' traveling model. To this end, we propose a computationally efficient solution of the game as shown in the next theorem.

Theorem 2. *The solution of the closed-loop dynamic Stackelberg game Ξ is equivalent to solving the static Stackelberg equilibrium at each individual time step.*

Proof. We begin the proof by investigating the solution of the closed-loop dynamic Stackelberg game which is the pair of strategies from (2.25) that satisfy (2.26). Our proof will show that the same reaction set in (2.25) can be achieved by considering the solution of the static Stackelberg game at each time step. Note that, the reaction set in (2.25) is a combination of the attacker's reactions to every single defender's strategy calculated from (2.24). In the following, we will show the solution when $\tau = 2$ and then generalize it to any number of time steps.

When $\tau = 2$, the attacker's cost function in (2.22) can be written as:

$$J^a(\boldsymbol{\beta}^d, \boldsymbol{\beta}^a) = \sum_{i=1}^{5} \left\| \boldsymbol{x}_{d_i}^a - \boldsymbol{x}_i(\Delta, z_i^d(1), z_i^a(1)) \right\|_2^2 + \sum_{i=1}^{5} \left\| \boldsymbol{x}_{d_i}^a - \boldsymbol{x}_i(2\Delta, z_i^d(2), z_i^a(2)) \right\|_2^2. \tag{2.27}$$

In the dynamic Stackelberg game, the attacker will select a strategy $\boldsymbol{\beta}^a = \{z_i^a(1), z_i^a(2)\}$ in response to every $\boldsymbol{\beta}^d$ that minimizes its utility. We can rewrite (2.27) by substituting the values from

(2.19):

$$
\begin{aligned}
J^a(\boldsymbol{\beta}^d, \boldsymbol{\beta}^a) = \sum_{i=1}^{5} \Big\| \boldsymbol{x}^a_{d_i} - \big(1 - z^a_i(1) + z^d_i(1) \cdot z^a_i(1)\big) \cdot \boldsymbol{x}_i(\Delta) \\
- \big(z^a_i(1) - z^a_i(1) \cdot z^d_i(1)\big) \cdot \boldsymbol{x}^a_i(\Delta) \Big\|_2^2 \\
+ \sum_{i=1}^{5} \Big\| \boldsymbol{x}^a_{d_i} - \big(1 - z^a_i(2) + z^d_i(2) \cdot z^a_i(2)\big) \cdot \boldsymbol{x}_i(2\Delta) \\
- \big(z^a_i(2) - z^a_i(2) \cdot z^d_i(2)\big) \cdot \boldsymbol{x}^a_i(2\Delta) \Big\|_2^2 .
\end{aligned} \tag{2.28}
$$

Now, consider that the defender chooses a specific strategy $\boldsymbol{\beta}^d = \{z^d_j(1) = 1, z^d_k(2) = 1\}$ where $j, k \in \mathscr{N}$. This means in the first time step $z^d_j(1) = 1$ and all the remaining actions will be be zero. Similarly, in the second time step $z^d_k(2) = 1$ and all the remaining actions will be be zero. The previous cost function can then be written as:

$$
\begin{aligned}
J^a(\boldsymbol{\beta}^d, \boldsymbol{\beta}^a) = \sum_{i=1, i\neq j}^{5} \Big\| \boldsymbol{x}^a_{d_i} - \big(1 - z^a_i(1)\big) \cdot \boldsymbol{x}_i(\Delta) - z^a_i(1) \cdot \boldsymbol{x}^a_i(\Delta) \Big\|_2^2 \\
+ \sum_{i=1, i\neq k}^{5} \Big\| \boldsymbol{x}^a_{d_i} - \big(1 - z^a_i(2)\big) \cdot \boldsymbol{x}_i(2\Delta) - z^a_i(2) \cdot \boldsymbol{x}^a_i(2\Delta) \Big\|_2^2 \\
+ \Big\| \boldsymbol{x}^a_{d_j} - \boldsymbol{x}_j(\Delta) \Big\|_2^2 + \Big\| \boldsymbol{x}^a_{d_k} - \boldsymbol{x}_k(2\Delta) \Big\|_2^2 .
\end{aligned} \tag{2.29}
$$

Now consider the attacker's response to $\boldsymbol{\beta}^d = \{z^d_j(1) = 1, z^d_k(2) = 1\}$. Let $\boldsymbol{\beta}^a = \{z^a_m(1) = 1, Z^a_n(2) = 1\}$ where $m, n \in \mathscr{N}$ is the attacker's response that achieves the minimum cost in (2.24). Similar to the defender's actions, in this case $z^a_m(1) = 1$ and $z^a_n(1) = 1$ and all the other attacker's actions will be zero. Now, rewrite the cost in (2.29) with respect to $\boldsymbol{\beta}^a = \{z^a_m(1) = 1, z^a_n(2) = 1\}$:

$$
\begin{aligned}
J^a(\boldsymbol{\beta}^d, \boldsymbol{\beta}^a) = \sum_{i=1, i\neq m,n}^{5} \sum_{t=1}^{2} \Big\| \boldsymbol{x}^a_{d_i} - \boldsymbol{x}_i(t\Delta) \Big\|_2^2 \\
+ \Big\| \boldsymbol{x}^a_{d_m} - \boldsymbol{x}^a_m(\Delta) \Big\|_2^2 + \Big\| \boldsymbol{x}^a_{d_m} - \boldsymbol{x}_m(2\Delta) \Big\|_2^2 \\
+ \Big\| \boldsymbol{x}^a_{d_n} - \boldsymbol{x}_n(\Delta) \Big\|_2^2 + \Big\| \boldsymbol{x}^a_{d_n} - \boldsymbol{x}^a_n(2\Delta) \Big\|_2^2 .
\end{aligned} \tag{2.30}
$$

As the cost in (2.30) represents the minimum cost in response to $\boldsymbol{\beta}^a = \{z^a_m(1) = 1, z^a_n(2) = 1\}$, the attacker cannot achieve a better cost by changing its strategy. This minimum cost was achieved

by attacking UAV m, at the first time step without affecting the other UAVs, and attacking UAV n, on the second time step, without affecting the other UAVs. This is because the attacker affects only one UAV, at a time step, and the remaining UAVs travel towards their real destinations. Note that, the attacker's choice at the second time step, i.e, UAV n is independent from its choice at the first time step. After the first time step, UAV n reached its real destination, and yet, this was the best for the attacker at the second time step. Since the action at the second time step is independent from the action at the first time step and it depends only on the new UAVs' locations after the first time step, the attacker will have the same reaction set if faced by the defender's actions sequentially instead of the whole strategy.

This finding can be extended to any number of time steps as the attacker's action, at a time step, will affect only one UAV and its actions in the following time steps will be based on the new UAVs' locations whether they were attacked or not at the previous time step. In other words, when faced by a strategy, the attacker cannot achieve a better outcome than responding at each time step independently. Considering this fact, the defender can determine the reaction set in (2.25) sequentially by solving each time step individually. After determining the complete reaction set, the Stackelberg strategies can be achieved from (2.26).

□

From Theorem 2, we can infer that the complexity of obtaining the solution will be reduced to determining the attacker's response and the defender's Stackelberg action at each time step. Thus, the complexity of the game will be reduced to $O(5^2\tau)$, which is linear in terms of the number of time steps. Note that, as discussed earlier, the solution of the formulated game is non-unique. Theorem 2 will then allow obtaining one of the Stackelberg equilibrium strategies.

2.5 Simulation Results and Analysis

For our results, we consider the case of one GPS spoofer and one group of five UAVs to better highlight the outcome of the dynamic game. The analysis will can apply to multiple groups of UAVs with an expected better outcome for the UAVs due to the decreased probability of attacking a single UAV. Note that, the following results are obtained through simulations in which the proposed analytical models are simulated using MATLAB. However, to further enhance the accuracy of the results, real experiments will be required to capture the UAVs' behaviors in a real environment and to examine the effect of other parameters abstracted in the simulations as discussed in Chapter 9.

First, in Figure 2.5, we show a visual output of the proposed game. The UAVs' starting locations, real destinations, and the attacker's desired destinations are all shown in this figure. The points A_i, $i = 1, \ldots, 5$ are the attacker's desired destinations for each UAV and the points D_i, $i = 1, \ldots, 5$ are the real destinations for each UAV. The UAVs update their locations every 50 meters and the value of e_{max} is assumed to be 50 m as well. Note that the maximum value of e_{max} that keeps the attack covert depends on the fault detector used within the UAV [104] and it can range from few

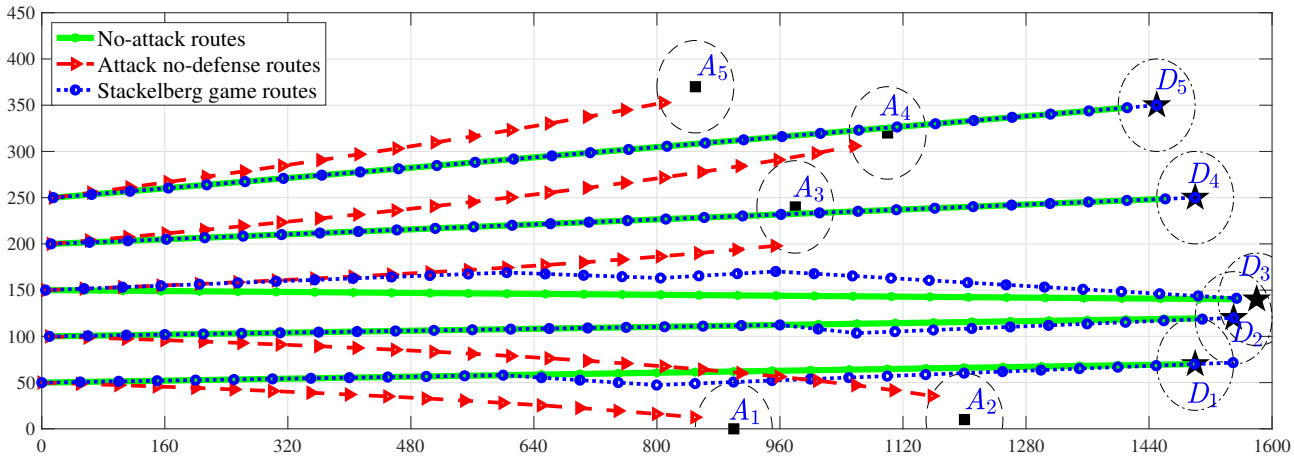


Figure 2.5: UAVs routes under no attack, no defense, and under the proposed Stackelberg game solution.

meters up to 90 m for different detector settings.

Figure 2.5 shows the different routes that each UAV can follow. The straight lines connecting each UAV's starting point to its destination, D_i, represent the shortest paths that each UAV will follow if there is no attack. These routes result from calculating the UAVs' locations according to (2.5) at each time step. On the other hand, the routes from the UAVs' starting points to the attacker's desired destinations, A_i, are the routes resulting from following the attacker's imposed locations at each time step, i.e., the locations in (2.17). Note that, these routes, unlike the shortest paths, are not straight lines as they are composed of multiple short segments each of which is the UAV path after perceiving the attacker's imposed location. In some of these paths, the attacker imposes multiple locations along the path causing the route to deviate towards the attacker's desired destinations. The UAVs can follow these routes only if there is an attack while the defense mechanism is not used. Finally, Figure 2.5 also shows the routes resulting from the proposed Stackelberg game solution. These routes are bounded by the previous two routes and may coincide with parts of these routes. Every change in these routes represents a change in the attacker's response action, and, hence a change in the UAV under attack. In the following, we will study how the routes resulting from the Stackelberg game compare to the previous two sets of routes, i.e., routes under no attack and routes under attack while no defense is used.

Next, we study the effect of GPS spoofing attacks on the UAVs' traveling routes.

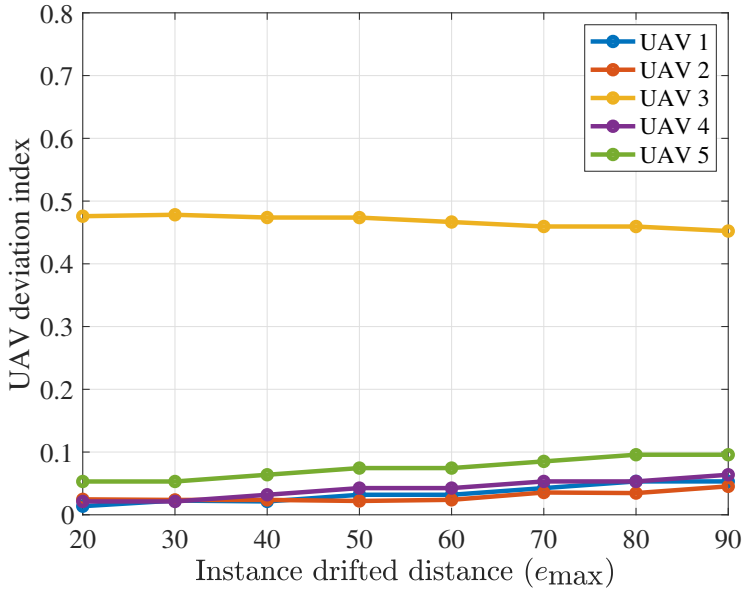


Figure 2.6: UAVs deviation index as a relation in the instance drifted distance.

2.5.1 UAVs Deviation due to Spoofing Attacks

To study the attacker's effect on the UAVs' traveling routes, we define a deviation metric, for each UAV, to compare the UAV's route resulting from the Stackelberg game with both the no-attack route and the attack no-defense route shown in Figure 2.5. Let $\boldsymbol{x}_i^r(t)$ be UAV i's location on the expected route under no attack and let $\boldsymbol{x}_i^f(t)$ represent its locations on the attacker's desired route. We then define $\theta_i(t)$ to be the deviation of UAV i at time step t given by:

$$\theta_i(t) = 1 - \frac{\left\| \boldsymbol{x}_i(t) - \boldsymbol{x}_i^f(t) \right\|_2^2}{\left\| \boldsymbol{x}_i^r(t) - \boldsymbol{x}_i^f(t) \right\|_2^2}, \tag{2.31}$$

such that when a UAV is traveling on a no-attack route, the value of $\theta_i(t)$ will equal 0. Similarly, if a UAV is traveling on the attacker's desired route, the value of $\theta_i(t)$ will equal 1. Any other value in between the two routes, the value of θ will be $0 < \theta < 1$. We can, then, define the deviation index Θ_i for UAV i as the average of its deviation over all time steps. Note that the deviation index can capture how far each UAV has traveled from its planned route towards the attacker's desired route. However, a higher deviation index does not necessary mean that this specific UAV will be captured by the attacker. It merely means that the attacker has disrupted the UAV's original route.

In Figure 2.6, we study the effect of the instance drifted distance, $e_{\max}$, on the UAVs' deviation indices. To better highlight the effect of $e_{\max}$, we allowed the UAVs to update their locations frequently by setting the update distance to 15 m. We notice that, as $e_{\max}$ increases, the attacker will be able to induce bigger changes to the UAVs' locations causing them to deviate more from the

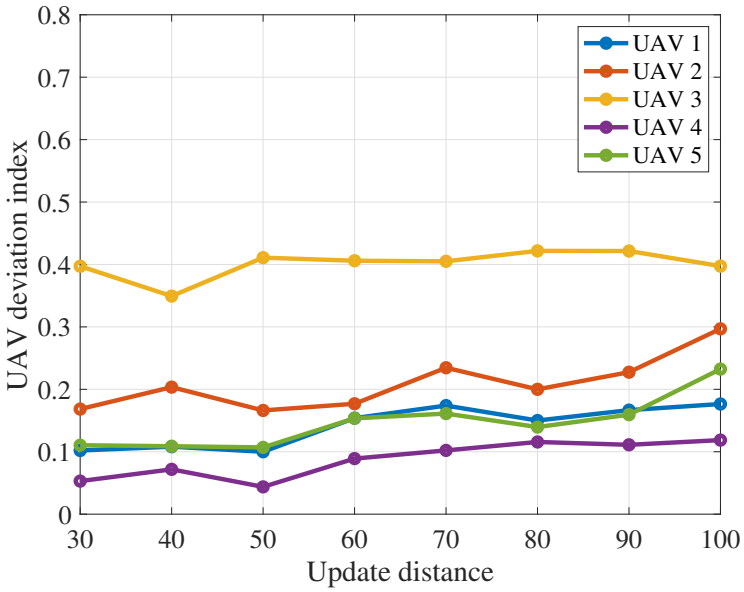


Figure 2.7: UAVs deviation index as a relation in the update distance.

planned routes hence increasing their deviation index. For instance, when $e_{\max} = 20$ m, some of the UAVs have almost zero deviation from their planned routes. On the other hand, when $e_{\max} = 90$ m, most UAVs have a slight deviation from their planned routes. We also note, from Figure 2.6, that UAV 3, has much higher deviation than the other UAVs. This happens as UAV 3 affects the attacker's utility the most while having a smaller effect on the defender's utility. Thus, UAV 3 is chosen by the attacker, at most time steps, as its response action while the defender chooses other UAVs to protect. Note that the players' utilities, and, hence, their chosen actions (UAVs) depend on the UAVs' current locations and both the real and the attacker's desired destinations.

Next, we study the effect of the update distance, i.e., the distance at which the UAVs apply the defense mechanism, on the UAVs' deviation indices. As the UAVs are traveling using $u^{\max}$, the update distance will indicate the frequency of updating the UAVs' locations. In Figure 2.7, we study different update distances on the deviation index. In this case, we set the value of $e_{\max}$ to 60 m. From Figure 2.7, we can see that, when the update distance increases, i.e., the less frequent the UAVs apply the defense mechanism, the more they deviate from their planned routes. For instance, when the update distance is set to 30 m, the average deviation index of all UAVs is 0.17 compared to 0.25 when the update distance is 100. This happens as the UAVs will travel more towards the attacker's destinations before they update their locations, and, move towards their real destinations. We also note that changing the update distance can change the effect of the UAVs on the players' utilities, and, hence on their actions. For instance, when the update distance is 40 m, UAV 2 is attacked more than when the update distance is 30 m, and, hence, it has a higher deviation index. For the same update values, UAV 3 has a lower deviation index when the update distance is 40 m compared to when it is 30 m.

Next, we study the effect of changing the attacker's desired destinations on the UAVs' deviation

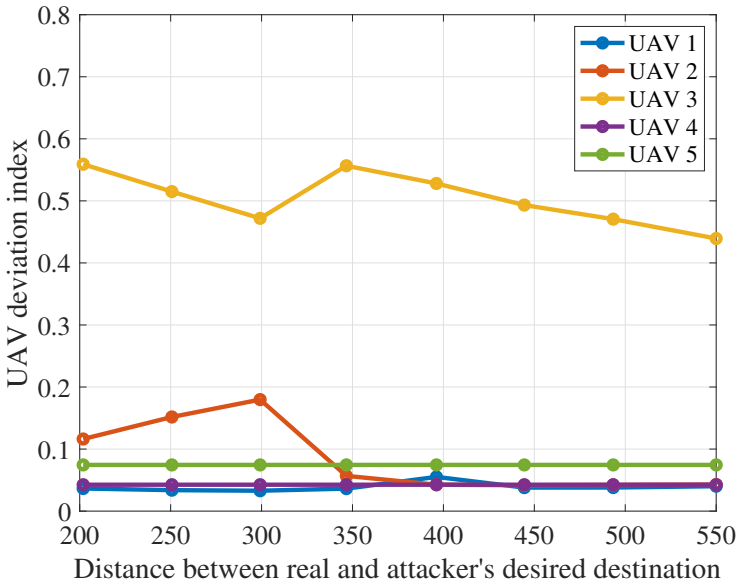
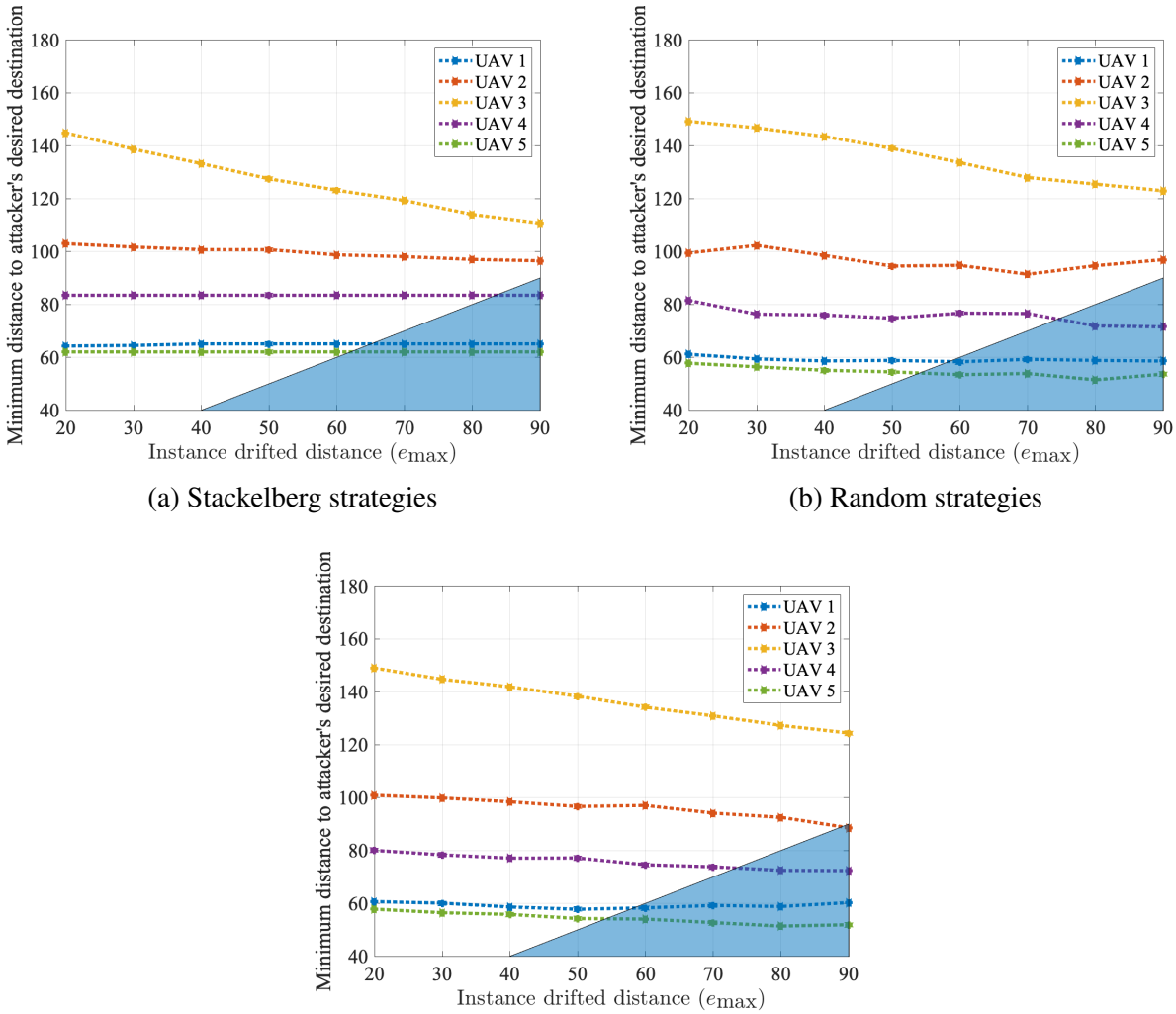


Figure 2.8: UAVs deviation index change due to shifting the attacker's desired destinations.

indices. In this case, we use the same parameters as in Figure 2.5. Different attacker's desired destinations, A_i, $i = 1, \ldots, 5$, are tested by reducing the distance between the attacker's desired destination and the real destinations randomly with an average change of 50 meters per UAV. In this case, the real destinations are fixed, and the attacker's desired destinations are shifted along the x direction only allowing for deviations to take place along the travel routes. Figure 2.8 shows the effect of changing the attacker's desired destinations on the UAVs' deviation indices. We can see that, as the attacker's desired destinations are closer to the real destinations, the deviation index increases. For instance, when the average distance between the attacker's and the real destinations is 550 m, the average deviation index for all UAVs is 0.13 compared to 0.17 when the average distance drops to 200 m. This is because when the distances are smaller, the attacker will have more opportunities (longer paths) to attack the UAVs causing them to deviate more from their planned routes. Note that, as the distances between the destinations are allows to change randomly, the changes in UAVs' distances to the attacker's desired destinations will not be constant. Hence, the UAVs will contribute differently to the attacker's utilities with each change. This will cause the attacker's actions (attacked UAVs) to be different over the travel routes, with each change. For example, we can see in Figure 2.8 that when the distance is 300 m, UAV 2 is attacked more than when the distance is 350 m. On the other hand, UAV 3 is less attacked when the distance changes from 350 m to 300 m.

Finally, we note that, in all the previous cases, while the three studied parameters affect the deviation index of the UAVs, the update distance has the most effect on the UAVs' deviation indices. This is because delaying the update will allow the UAVs to travel on the attacked routes for longer distances before correcting their locations leading to larger deviations. Meanwhile, in the other scenarios, the attacker caused a smaller average deviation on the UAVs because the UAVs update



(a) Stackelberg strategies

(b) Random strategies

(c) Deterministic strategies

Figure 2.9: The effect of the instance drifted distance, $e_{\max}$, on the possibility of UAV capture.

their locations more frequently. These findings corroborate the importance of the proposed defense mechanism and provide important insights for the drone operator to choose suitable update distances according to the available resources.

Next, we study the attacker's possibility to capture any of the UAVs under GPS spoofing attacks.

2.5.2 Capturing Possibilities under GPS Spoofing Attacks

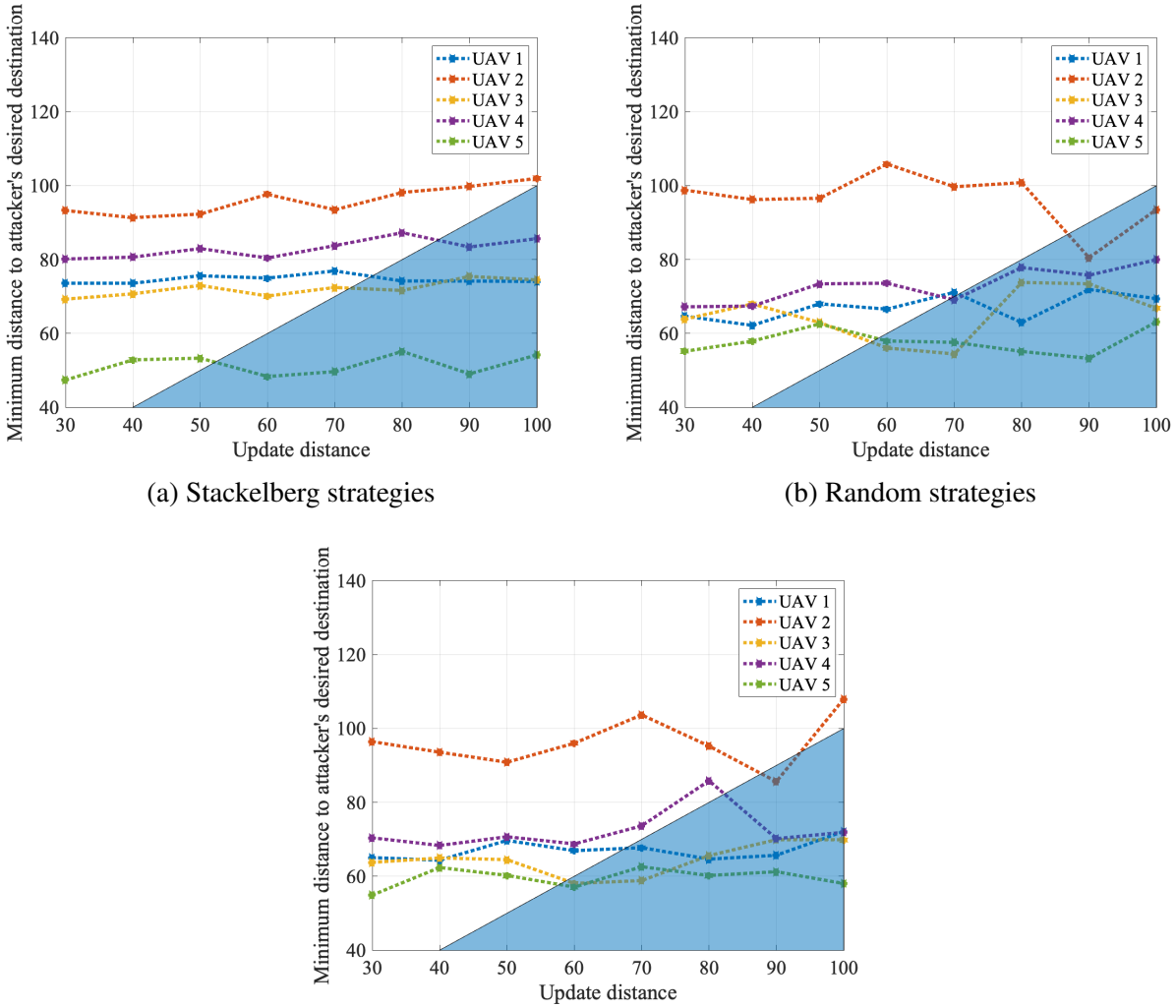
To study the capture possibility of the UAVs, we assume the attacker needs to change a UAV's route, by imposing a different location, in order to capture it. We also assume that the UAV will be captured if it reaches a distance $e_{\max}$ from the attacker's desired destination. In the following,

we will compare our proposed Stackelberg solution with two other non-game-theoretic baselines referred to as random and deterministic approaches. In the random approach, the defender chooses any UAV randomly to protect at each time step. In the deterministic approach, the defender considers all the UAVs, in order, by choosing one at each time step. In all the three cases, the attacker chooses its strategies in response to the defender's chosen strategies.

Figure 2.9 shows the minimum distance that each UAV can reach from its attacker's desired destination, for each value of $e_{\max}$. The shaded areas in Figure 2.9 represent the distances under which the UAV is considered to be captured which correspond to having a distance less than $e_{\max}$ to the attacker's desired destination. The configuration parameters in this case are similar to Figure 2.6. We can see in Figure 2.9a that using the Stackelberg strategies, the defender is able to protect all the UAVs until $e_{\max} = 60$ m. When $e_{\max} = 70$ m and $e_{\max} = 80$ m, both UAVs 1 and 5 can be captured by the attacker and when $e_{\max} = 90$ m, UAV 4 can also be captured. In Figure 2.9b, when the drone operator uses random strategies, we can see that the attacker is able to capture UAVs 1 and 5 starting from $e_{\max} = 60$ m. Moreover, UAV 4 can be captured starting from $e_{\max} = 80$ m. Under deterministic strategies, Figure 2.9c, the attacker is also able to start capturing UAVs 1 and 5 starting from $e_{\max} = 60$ m. It will be also able to capture UAV 4 starting from $e_{\max} = 80$ m and to capture UAV 2 at $e_{\max} = 90$ m.

It is clear from Figure 2.9 that following the Stackelberg strategies will help the defender to protect more UAVs, particularly for higher values of $e_{\max}$. This is due to the fact that, under Stackelberg strategies, the drone operator considers its utilities based on the attacker's response strategies which allows it to mitigate the effect of the attacker's expected actions. Figure 2.9 also shows that UAV 3 has the most changes to its minimum distance from the attacker's desired destination, when $e_{\max}$ increases. This corroborates the result of Figure 2.6 whereby UAV 3 was the most affected by the spoofer's imposed locations, in terms of deviation from its planned route. However, UAV 3 remains far enough from being captured as the defender's actions allow it to return to the correct traveling direction.

Next, we study the effect of changing the UAVs' update distance on the possibility of UAV capture. The configuration parameters in this case are similar to Figure 2.7. Figure 2.10 shows the effect of the capture possibility with the shaded areas representing the distances under which the UAVs are considered to be captured. We can see in Figure 2.10a that using the Stackelberg strategies, the drone operator is able to protect all the UAVs until an update distance of 50m. The attacker is able to capture its first UAV, UAV 5, when the update distance is 60 m or 70 m. When the update distance is 80 m, three UAVs can be captured by the attacker, and four UAVs can be captured for update distances greater than 80 m. Under random strategies, Figure 2.10b shows that the attacker is able to capture more UAVs when the update distance is 60 m as it will capture UAVs 3 and 5. Similarly, for all the consequent update distances, more UAVs can be captured compared to the Stackelberg strategies. Under deterministic strategies, Figure 2.10c shows that the attacker will be able to start capturing UAVs 3 and 5 at an update distance of 60 m. When the update distance is 70 m, the attacker will be able to capture three UAVs compared to one in the Stackelberg strategies. For the consequent update distances, the attacker is able to capture at least the same number of UAVs as the Stackelberg strategies.



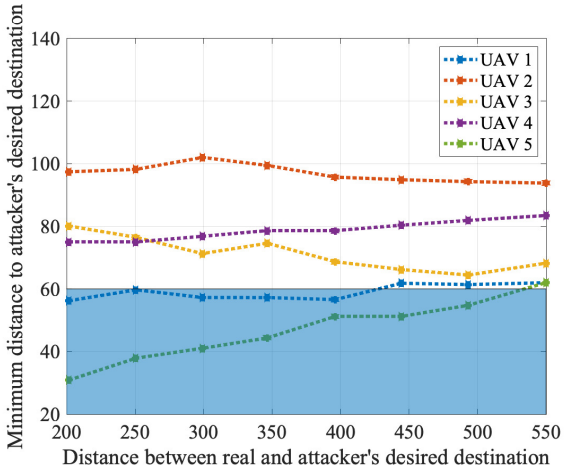
(a) Stackelberg strategies

(b) Random strategies

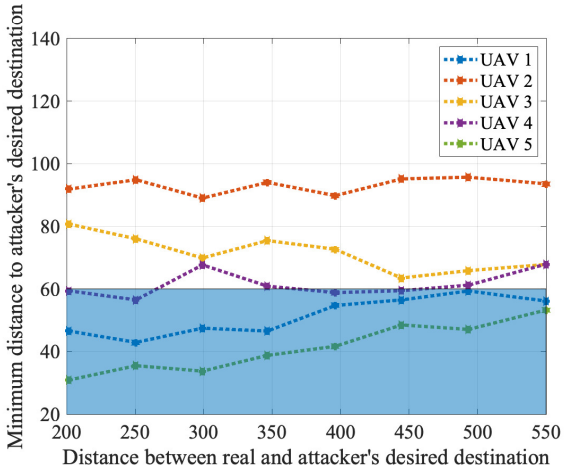
(c) Deterministic strategies

Figure 2.10: The effect of the update distance on the possibility of UAV capture.

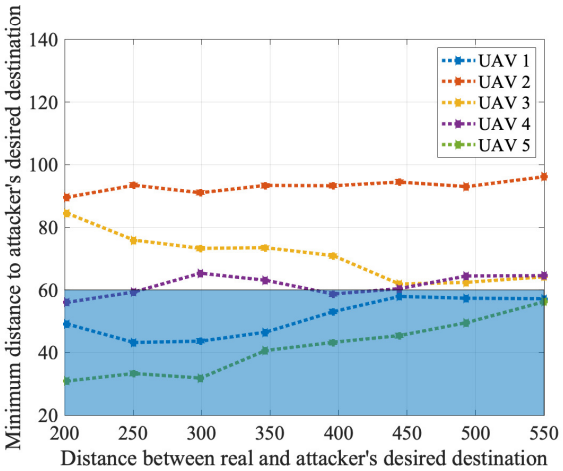
Next, we study the effect of changing the attacker's desired destinations on possibility of UAV capture. The configuration parameters in this case are similar to Figure 2.8, with $e_{\max} = 60$ m. Figure 2.11 shows this effect on the capture possibility. The shaded areas in Figure 2.11 represent the capture distances of the UAVs, i.e., distances less than 60 m. We can see in Figure 2.11a that using the Stackelberg strategies, the drone operator is able to protect all the UAVs when the average distance is 550 m. When the average distance decreases to 500 m, the attacker will be able to capture UAV 5. For any distances less than 400 m, the attacker will be able to capture UAVs 1 and 5. Under random strategies, Figure 2.11b, we can see that the attacker is able to capture UAVs 1 and 5 for all the considered distances. The attacker will also be able to capture UAV 4 under multiple distance settings. Finally, we can see in Figure 2.11c, that the deterministic strategies show very similar response to the random strategies, in terms of the possibility of UAV capture.



(a) Stackelberg strategies



(b) Random strategies



(c) Deterministic strategies

Figure 2.11: The effect of the average distance between the real and the attacker's desired destinations on the possibility of UAV capture.

Note that, from the three previous scenarios, we can see that the update distance has the most effect on the possibility of UAV capture, similar to its effect of the deviation index. This highlights the importance of choosing this critical parameter when applying the proposed defense mechanism.

2.6 Summary

In this chapter, we have proposed a novel framework to mitigate the effects of capture attacks via GPS spoofing that target UAVs. Systems dynamics have been used to model the UAVs' optimal routes towards their destinations. To study the effect of a GPS spoofer on these optimal routes,

we have mathematically derived the spoofer's optimal imposed locations on any UAV. These locations, when imposed on a UAV, cause the UAVs to deviate from their planned routes and follow new routes towards the spoofer's desired destinations. We have then proposed a countermeasure defense mechanism to allow UAVs to determine their real locations, after being attacked. This countermeasure is built on the premise of cooperative localization, in which a UAV uses the locations of nearby UAVs to determine its real location. We have, then, defined a Stackelberg game problem to allow the UAVs to better utilize the proposed defense mechanism. In particular, the game is formulated between a GPS spoofer and a drone operator that manages a number of UAVs. The drone operator is considered the leader that determines its strategies first and the spoofer then responds by choosing its strategies. We have mathematically derived the Stackelberg equilibrium strategies, for the formulated game, through a computationally efficient approach. Results have shown that the proposed defense mechanism along with Stackelberg equilibrium strategies outperform other strategy selection techniques in terms of reducing the possibility of UAV capture. We have also tested the effect of different parameters on the UAVs' deviation indices and on the possibility of UAV capture and the results have shown that the UAV update distance has the most effect on these metrics.

Chapter 3

Single Controller Stochastic Games for Optimized Moving Target Defense

3.1 Background, Related Works, and Contributions

As discussed in section 1.1.4, MTD [52] is an effective way to thwart attacks on reconfigurable environments. MTDs are built on the premise of continuously randomizing the network's configuration (e.g., cryptographic keys, network parameters, IP addresses) so as to increase the uncertainty and cost of attack on the adversary. The effective deployment of MTDs requires meeting several challenges that range from optimizing the randomization to analyzing the costs and benefits of MTDs [53–61].

A number of research works have recently attempted to address some of these challenges [52–55]. First, in [52], , the authors focus on the five dominant domains in which MTD techniques could be applied against cyber attacks in critical systems. In this work, defined these domains to be networks, platforms, runtime environments, software, and data. They studied the weakness and advantages of using MTD in these domains. In [53], the authors proposed a three-layer model to evaluate the effectiveness of MTDs in software. These layers capture low-level contexts in separate programs, model damage propagation between different programs, and provide a user interface to expresses evaluation results. The work in [54] considers an MTD to be a subclass of system agility. In this work, system agility is defined as any reasoned modification to a system or environment in response to a functional, performance, or security need. In [55], the authors propose a foundation for defining the theory of MTD. They defined key problems and hypothesis related to MTD such as the way to select the next valid configuration of the system, configuration space, and the timing problem.

The use of MTD in resource-constrained distributed devices, e.g., wireless sensor networks was studied in [56]. The authors proposed two different reconfigurations at different architectural layers. The first is applied at what they defined as a security layer by using a number of cryptographic

techniques and each node in the network can choose its encryption method for each packet by adding a special identifier in the packet header. The second reconfiguration is to be applied at the physical layer by changing the node's firmware. In [57], the authors use MTD to defend against selective jamming attacks. This work studies the problem of isolating a subset of the network by jamming the signals sent from this sub-network. The work in [57] also provides practical MTD solutions such as address flipping and random address assignment. The use of software defined networking in applying MTD was discussed in [58]. The authors defined a technique to MTD by assigning virtual IPs to hosts in the network beside their reals IPs. Software-defined networking was used to manage the IP translation. However, these works are mostly qualitative or experiment based and, as such, they do not address specific MTD problem formulations.

More recently, game-theoretic methods have recently attracted attention as a suitable tool for implementing MTDs [59–61]. In [60], the authors develop a zero-sum stochastic game model to a feedback-driven multi-stage MTD. A feedback learning framework was used to implement MTD based on real-time data and observations made by the system. The purpose of the learning algorithm for the defender is to monitor its current state and update its randomized strategy based on its observation. In this model, the attacker launches a multi-stage attack and the defender responses at each layer. In [61], the authors analyze a system in which the defender has a number of different platforms to run a critical application and the attacker has a set of attacks that are applicable against some of these platforms. The authors proposed two types of attackers, static and adaptive, and gave attack model to both of them. The authors in [54], also suggested that MTD games should be modeled as tunable hierarchical games. The output of a game at one level should determine the level of risk associated with a game at a different level.

A recent collection of publications for applying game theory in MTD [59] does not provide clear approaches to concretely reach equilibrium strategies. Moreover, the works in [60] and [61] abstract many of the details of the network considered, and, thus, they cannot directly apply to practical systems. Moreover, the work in [9] assumes the presence of a highly intelligent layered attacker which may not be true in practice.

The main contribution of this chapter is to develop a novel game-theoretic model for MTD that can be applied to securing a wireless network. In particular, we consider a wireless system in which a base station (BS) seeks to implement an MTD-based cryptographic approach in which it randomizes over various cryptographic keys and techniques so as to evade an eavesdropper that is trying to decrypt the messages. We formulate the problem as a single-controller non-zero-sum stochastic game in which the BS uses a number of cryptography techniques along with a number of keys for each technique. The BS can implement MTD by randomizing over various actions that include choosing an encryption method defined by specific encryption technique and key combination. We also consider a defense cost for applying MTD that depends on the number of consecutive changes in the system. Since our model deals with resource-constrained systems, the encryption techniques should not be highly resource consuming. Therefore, we develop an approach that attempts to avoid the use of encryption techniques with long encryption keys in order to decrease the power consumption. While short-key encryption techniques are more vulnerable against attacks, MTD will allow the BS to switch between encryption techniques and so it is unlikely that the at-

tacker will be able to reveal the key before it is changed. For this game, it is shown that a Nash equilibrium always exists. To find this equilibrium, we propose an algorithm based on bimatrix game equilibrium defined for all possible pure stationary strategies of the original game. Simulation results show that the proposed approach will yield a higher defender's utility when compared with other schemes that randomly pick the strategies.

The rest of the chapter is organized as follows. Section 3.2 provides the system model, assumptions, and defines the defender's and attacker's utilities. In Section 3.3, the stochastic game is formulated and the steps of calculating equilibrium points are shown, and also a way to define cost function in MTD. Simulation results are discussed in Section 3.4. Finally, conclusions are drawn in Section 3.5.

3.2 System Model and Problem Formulation

Consider a wireless sensor network that consists of a BS and a number of wireless nodes. The network is deployed for sensing and collecting data about some phenomena in a given geographic area. Sensors will collect data and use multi-hop transmissions to forward this data to a central receiver or BS. The multiple access follows a slotted Aloha protocol. Time is divided into slots and the time slot size equals the time required to process and send one packet. Sensor nodes are synchronized with respect to time slots. We assume that nodes are continuously working and so every time slot there will be data that must be sent to the BS.

All packets sent over the network are assumed to be decrypted using a given encryption technique and a previously shared secret key. All the nodes in the system are pre-programmed with a number of encryption techniques along with a number of encryption keys per technique, as what is typically done in sensor networks [56]. The BS chooses a specific encryption technique and key by sending a specific control signal over the network including the combination it wants to use. We note that the encryption technique and key sizes should be carefully selected in order not to consume a significant amount of energy when encrypting or decrypting packets. Increasing the key size will increase the amount of consumed energy particularly during the decryption [111]. Since the BS is mostly receiving data, it spends more time decrypting packets rather than encrypting them and, thus, it will be highly affected by key size selection.

In our model, an eavesdropper is located in the communication field of the BS and it can listen to packets sent or received by the BS. As packets are encrypted, the attacker will seek to decrypt the packets it receives in order to get information. The attacker knows the encryption techniques used in the network and so it can try every possible key on the received packets until getting useful information. This technique is known as brute-force attack.

The idea of using multiple encryption techniques was introduced in [56]. However, in this work, each node individually selects one of these technique to encrypt transmitted packets. The receiving node can know the used technique by a specific field in the packet header. Large encryption keys

were used which require a significant amount of power to be decrypted. Nonetheless, these large keys are highly unlikely to be revealed using a brute-force attack in a reasonable time. Here, we propose to use small encryption keys to save energy and, in conjunction with that, we enable the BS to change the encryption method in a way that reduces the chance that the encryption key is revealed by the attacker. This is the main idea behind MTD. In MTD techniques, the defender aims to change the attack surface [63] which represents the points that could be attacked. In this model, the encryption key represents the attack surface, and by changing the encryption method, the BS will make it harder for the eavesdropper to reveal the key and get the information from the system.

Naturally, the goals of the eavesdropper and the BS are not aligned. On the one hand, the BS wants to protect the data sent over the network by changing encryption method. On the other hand, the attacker wants to reveal the used key in order to get information. To understand the interactions between the defender and the attacker, one can use game theory to study their behavior in this MTD scenario. The problem is modeled as a game in which the attacker and the defender are the players. As the encryption method should be changed over time and depending on the attacker's actions, we must use a dynamic game.

Thus, we formulate a stochastic game Ξ described by the tuple $\langle \mathscr{N}, \mathscr{S}, \mathscr{A}, \mathscr{P}, \mathscr{U}, \beta \rangle$ where $\mathscr{N}$ is the set of the two players: the defender p_1, the BS, and the attacker p_2, the eavesdropper. $\mathscr{S}$ is the set of game states and $\mathscr{A}$ is the set of actions defined for each player at every state. $\mathscr{P}$ is the set of transition probabilities between states. $\mathscr{U}$ is the set of utilities each player will get for a given combination of actions and state. Finally, $0 < \beta < 1$ is a discount factor.

The defender can choose to use one of the N available encryption techniques or to use the current encryption technique with one of the M available encryption keys predefined for this technique. Each game state is well defined by the current encryption technique and key combination. Therefore, there will be $K = N \cdot M$ states, i.e., $\mathscr{S} = \{s_1, s_2, \ldots, s_K\}$. In each state $s \in \mathscr{S}$, each player has a set of actions $\mathscr{A}_i$. Let $\mathscr{A}_1 = \{a_1^1, a_2^1, \ldots, a_K^1\}$ be the defender's actions which represent the choice of a specific technique and key combination among the available K combinations. Let $\mathscr{A}_2 = \{a_1^2, \ldots, a_N^2\}$ be the action set of the attacker which represents the set of techniques that the attacker is trying to decrypt.

In each state $s \in \mathscr{S}$ and for each action pair in $\mathscr{A}_1 \times \mathscr{A}_2$, there is an outcome (payoff) for each player. This outcome depends on the current state and actions taken by both players in this state. This outcome is defined by player-specific utility functions in $\mathscr{U}$. For given actions $a^1 \in \mathscr{A}_1$ and $a^2 \in \mathscr{A}_2$, the defender's utility at state s_i is given by:

$$U_1(a^1, a^2, s_i) = R_1(a^2) + T_1(a^1, a^2, s_i) - P_1(s_i), \tag{3.1}$$

where R_1 is the reward gained from protecting a packet. This reward depends on the attacker's action as the defender will obtain a higher reward if the eavesdropper is attacking another encryption technique. P_1 is the power used to decrypt a packet and it depends on the technique (state). T_1 is the transition reward that the defender will gain from applying MTD and choosing a key-technique combination. This reward depends on the current system state, the defender's action taken at this state (which determines the next state), and attacker's action.

Similarly, the attacker's utility at state s_i for given actions $a^1 \in \mathscr{A}_1$ and $a^2 \in \mathscr{A}_2$ will be given by:

$$U_2(a^1, a^2, s_i) = R_2(a^1, a^2, s_i) - P_2(s_i), \tag{3.2}$$

where R_2 is the attacker's reward from examining the encryption keys for a given technique. Here, if the attacker can examine more keys, it will get closer to revealing the actual key. This reward depends on the attacker's action, current encryption technique (state), and defender's action. P_2 is the power used to decrypt a packet that depends also on the current technique.

Based on these rewards, the game is non-zero sum. Thus means, every player will try to maximize its reward and the sum of rewards is not zero. This stochastic game also exhibits an interesting property pertaining to the fact that the transition probabilities in $\mathscr{P}$ depend only on the actions of the defender. Moreover, when the defender selects an action at one state, the game moves to another state defined by the encryption technique and key combinations with a probability $p = 1$. This type of stochastic games is known as *single-controller stochastic games* [112].

This type of games is most suitable for MTD problems in which the defender aims at randomizing system parameters, as the goal of MTD is to change system parameters in order to harden the attacker's mission. The defender should take actions to change these parameters within a reasonable time. Single-controller stochastic games satisfy this property by allowing the defender to control the actions thus changing the game state which maps to changing system parameters in MTD.

3.3 Proposed MTD Game Solution

3.3.1 Equilibrium Strategy Determination

The studied game is a finite stochastic games since the number of states and the number of actions per state are finite. Stochastic games are dynamic in the sense that the game moves between states each time step. In stochastic games we are interested in the accumulated (total) utilities of the players over time. Discounted utilities over time are typically used by summing the current utility and all the expected future utilities multiplied by a discount factor. In such cases, players are interested more in current payoffs than future ones. Each player seeks to take actions that maximize its utility given the other player's actions. When no player can improve its utility by solely changing its actions, the game is said to be at equilibrium.

For discounted stochastic games, the existence of Nash equilibrium points in stationary strategies was proven [113]. Stationary strategies are those strategies in which the actions taken at each state depend on this state only. If at each state, the player selects a specific action with probability $p = 1$ then this called pure stationary strategy. If the player chooses between actions with some probabilities then it is called a *mixed stationary strategy*.

In [114], the authors propose a scheme that can find a Nash equilibrium point for discounted non-zero sum single-controller stochastic games. The key idea is to form a bimatrix game (one matrix

for each player). The rows and columns of each matrix represent pure stationary strategies for each player. The elements of these matrices represent the accumulated discounted utilities over all states (recursion) for every strategies pair. Then, any mixed strategy Nash equilibrium of this bimatrix game can be used to get a Nash equilibrium of the stochastic game.

Since the defender is the controller which selects actions to move the game to a specific state, time steps of the stochastic game are controlled by the defender. Assuming that the attacker has enough power, it can complete the brute-force attack in time t_i for $i = 1, 2, \ldots, N$ for each one of the encryption techniques. Then, the defender should choose the time step t to take the next action as follows:

$$t < \min(t_i), \;\; i = 1, 2, \ldots, N. \tag{3.3}$$

By doing this, the defender can make sure that it takes a timely action before the attacker succeeds in revealing one of the keys.

The accumulated utility of player i at state s will be:

$$\Phi_i(\boldsymbol{f}, \boldsymbol{g}, s) = \sum_{t=1}^{\infty} \beta^{t-1} \cdot U_i(f(s_t), g(s_t), s_t), \tag{3.4}$$

where $\boldsymbol{f}$ and $\boldsymbol{g}$ are the strategies adopted by the defender and attacker, respectively. The strategy specifies a vector of actions to be chosen at each of the states, e.g., $\boldsymbol{f} = [f(s_1), \ldots, f(s_K)]$ for all the K states. Actions $f(s_t)$ and $g(s_t)$ are the actions chosen at s_t, which is the state of the game at time t, according to strategies $\boldsymbol{f}, \boldsymbol{g}$. State $s_t \in \mathscr{S}$ is determined by the defender's action at time $t-1$. The game is assumed to start at a specific state $s = s_1$. Note that the utility in (3.4) is always bounded at infinity due to the fact that $0 < \beta < 1$.

When designing the bimatrix, the defender needs to calculate the accumulated utility when choosing each pure strategy against all of the attacker's pure strategies. The defender, as a controller, can know the next state resulting from its actions, and, thus, it sums the utilities in all states using the discount factor β. Let $\boldsymbol{X}$ be the defender's accumulated utility matrix for all defender's pure strategies' permutations and all attacker's pure strategies' permutations. We let $\boldsymbol{F}_{i.} = [\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_{K^K}]$ be a matrix of all defender's pure strategies' permutation where each row represents actions in this strategy and similarly $\boldsymbol{G}_{i.} = [\boldsymbol{g}_1, \boldsymbol{g}_2, \ldots, \boldsymbol{g}_{N^K}]$ the matrix of all attacker's pure strategies' permutation. Then each element $X_{i,j}$ of $\boldsymbol{X}$ will be given by:

$$X_{i,j} = \sum_{\mathscr{S}} \Phi_1(\boldsymbol{F}_{i.}, \boldsymbol{G}_{j.}, s), \forall i, j, \tag{3.5}$$

where $i = 1, \cdots, K^K$ and $j = 1, \cdots, N^K$. The attacker can only calculate its payoffs at time $t = 1$, as the attacker cannot know in advance the actions taken at each state and hence the reward it will get in future. Similarly, let $\boldsymbol{Y}$ be the attacker's accumulated utility matrix, then each element $Y_{i,j}$ of $\boldsymbol{Y}$ will:

$$Y_{i,j} = \sum_{\mathscr{S}} \Phi_2(\boldsymbol{F}_{i.}, \boldsymbol{G}_{j.}, s), \forall i, j, \tag{3.6}$$

where i and j are the same as the defender's case, and $\Phi_2(\boldsymbol{F}_{i\cdot}, \boldsymbol{G}_{j\cdot}, s)$ is only evaluated at time $t = 1$.

The solution of the bimatrix could be obtained by algorithms such as Lemke-Howson [115], which is proven to always terminate at a solution and hence finds a mixed Nash equilibrium of the bimatrix game. This solution is then used as in [114] to find the equilibrium of the stochastic game. Let $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ be any mixed strategy Nash equilibrium point for the bimatrix game $(\boldsymbol{X}, \boldsymbol{Y})$. Each $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ is a vector of probabilities with which each player can choose each strategy in all the strategies permutations.

As each strategy represents the set of actions per all states, the equilibrium point to the stochastic game, i.e, the probability of choosing each strategy, can be calculated as:

$$E^*_{i,j} = \sum_{l=1, i=F_{l,j}}^{K^K} x^*_l,\ i = 1, \ldots, K, j = 1, \ldots, K,$$
$$H^*_{i,j} = \sum_{l=1, i=G_{l,j}}^{K^K} y^*_l,\ i = 1, \ldots, N, j = 1, \ldots, K, \tag{3.7}$$

where $x^*_l \in \boldsymbol{x}^*$ and $y^*_l \in \boldsymbol{y}^*$ are the elements of $\boldsymbol{x}^*, \boldsymbol{y}^*$ that represent strategies' probabilities. Each element $E^*_{i,j}$ of $\boldsymbol{E}^*$ and $H^*_{i,j}$ of $\boldsymbol{H}^*$ is the probability of taking action i in state j for the defender and the attacker, respectively. The summations in (3.7) give the probabilities of one action i which satisfies the condition. This is repeated for all values of i to get a column which is all actions' probabilities in one state. Different values of j give the rest of the states. $\boldsymbol{E}^*$ is a $K \cdot K$ matrix that gives the probability of each of the defender's K actions in each of the K states. Similarly, $\boldsymbol{H}^*$ is an $N \cdot K$ matrix that gives the probability of each of the attacker's N actions in each of the K states. These matrices are the *equilibrium strategies* for both players.

These probabilities specify the behavior of the game. The defender in each state will choose an action (selecting an encryption method) with some probability and so the game will move to another state (encryption method). Then, again in the new state, the defender chooses a new action and so on. Using this process, the defender will keep moving between encryption methods which effectively implements a highly randomized MTD.

Finally, the value (expected utility) of each player at equilibrium can be computed by applying the equilibrium strategies and finding the accumulated payoffs of both players. These expected utilities are calculated by following all the possible transitions due to defender's actions in each state. Let $v^*_i(s)$ be player's i value at state s:

$$v^*_i(s) = \Phi_i(\boldsymbol{E}^*, \boldsymbol{H}^*, s)\ \ s \in \mathscr{S}, \tag{3.8}$$

As the players get these values at equilibrium, both players will not have an incentive to deviate from these equilibrium strategies. The player who deviates will get a lower value when the other

player uses its equilibrium strategy. This can be expressed as:

$$v_1^*(s) \geq \Phi_1(\hat{\boldsymbol{E}}, \boldsymbol{H}^*, s),\ s \in \mathscr{S},$$
$$v_2^*(s) \geq \Phi_2(\boldsymbol{E}^*, \hat{\boldsymbol{H}}, s),\ s \in \mathscr{S}, \tag{3.9}$$

for any $\hat{\boldsymbol{E}}$ and $\hat{\boldsymbol{H}}$ other than the equilibrium strategies.

3.3.2 Numerical Example

Assume that the defender will use two encryption techniques with one key each, then the defender has two actions and the game has two states. The attacker has also two actions of attacking each technique.

A defender's strategy will be $\boldsymbol{f} = [a\ b]$ where a is the action at state s_1 and b the action at state s_2.

The defender's and attacker's strategies' permutation will be given as:

$$\boldsymbol{F} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 2 & 1 \\ 2 & 2 \end{bmatrix} \boldsymbol{G} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 2 & 1 \\ 2 & 2 \end{bmatrix}$$

The elements of the bimatrix will be the utilities for each of each of these strategies combinations and all the possible transitions due to these strategies. Therefore, both $\boldsymbol{X}$ and $\boldsymbol{Y}$ will be a $4 \cdot 4$ matrix.

Now suppose the mixed Nash equilibrium for the bimatrix game is given by:

$$\boldsymbol{x}^* = \begin{bmatrix} 0.2 \\ 0.1 \\ 0.3 \\ 0.4 \end{bmatrix} \boldsymbol{y}^* = \begin{bmatrix} 0.0 \\ 0.1 \\ 0.4 \\ 0.5 \end{bmatrix}$$

Then the equilibrium strategies for both players are:

$$\boldsymbol{E}^* = \begin{bmatrix} 0.3 & 0.5 \\ 0.7 & 0.5 \end{bmatrix} \boldsymbol{H}^* = \begin{bmatrix} 0.1 & 0.4 \\ 0.9 & 0.6 \end{bmatrix}$$

where rows represent actions and columns represent states.

3.3.3 Moving Target Defense Cost

In previous sections, the defender's utility included a reward from applying MTD which corresponds to the gain from randomizing system parameters. However, applying MTD may incur associated costs for the defender. Examples include the cost of reconfiguring the system and changing parameters. In our decryption model, the BS might not be able to change the encryption method unless it ensures that all nodes are informed by the change, which requires some propagation time. Changing the method before this time can lead to a conflict in the used method between various nodes around the BS (e.g., nearby and far away nodes).

We model this cost as a function of the number of consecutive encryption method changes in the past time steps. Let the number of consecutive method changes during the past time steps be n and the cost value be q. The cost function will be $C(q, n)$ and it is an increasing function in the number of consecutive changes n. The defender's utility can then be written as:

$$U_1(a^1, a^2, s_i) = R_1(a^2) + T_1(a^1, a^2, s_i) - P_1(s_i) - C(q, n). \tag{3.10}$$

Clearly, n will be zero at the first time step. The effect of this cost can appear in the accumulated utility in (3.4) which will affect the matrix $\boldsymbol{X}$ in (3.5) and the defender's equilibrium values in (3.8).

We propose two different functions to express the cost. The first cost function can be expressed as $C(q, n) = q \cdot n$. We need to make sure that the game will remain finite after adding this cost function so that the same solution can be applied. As the cost affects the utility, we can state the following lemma:

Lemma 2. *The accumulated defender's utility will remain bounded after adding a cost function in the form $C(q, n) = q \cdot n$ and, thus, the game will still admit an equilibrium point.*

Proof. We prove this lemma by rewriting the defender's accumulated utility:

$$\Phi_1(f, g, s) = \sum_{t=1}^{\infty} \beta^{t-1}\big(R_1(a^2) + T_1(a^1, a^2, s_i) - P_1(s_i) - q \cdot n\big).$$

by noticing that the maximum for n is $t - 1$ and taking the limit as t reaches ∞ we get

$$\lim_{t\to\infty} \beta^{t-1}\big(R_1(a^2) + T_1(a^1, a^2, s_i) - P_1(s_i) - q \cdot (t-1)\big) = 0.$$

□

A second form for the cost function is $C(q, n) = q \cdot \ln(n + 1)$. We choose such a logarithmic function to reduce the effect of cost propagation. Note, Logarithmic function has a smaller rate of growth compared to the linear function in the first case. We need to ensure that the game will remain finite by adding this cost function, so we state the following lemma:

Lemma 3. *The accumulated defender's utility will remain bounded after adding the cost function $C(q, n) = q \cdot \ln(n + 1)$ and, thus, the game will still admit an equilibrium point.*

Table 3.1: Attacker's and defender's equilibrium strategies

	Attacker		Defender			
	a_1	a_2	a_1	a_2	a_3	a_4
s_1	0.7436	0.2564	0.0000	0.6622	0.1681	0.1697
s_2	0.7436	0.2564	0.4441	0.0195	0.1697	0.3667
s_3	0.3482	0.6518	0.4441	0.3667	0.0195	0.1697
s_4	0.3482	0.6518	0.4441	0.3667	0.1697	0.0195

Proof. We prove this lemma in a manner analogous to Lemma 2 where the limit will be zero also. □

In general, any function could be used to represent the propagation cost when its limit is bounded at infinity.

3.4 Simulation Results and Analysis

For our simulations, we choose a system that uses 2 encryption techniques with 2 different keys per technique. Thus, the number of system states are 4 and the defender has 4 actions in each state. For the bimatrix, the attacker has $2^4 = 16$ different strategy permutations and the defender has $4^4 = 256$ different strategy permutations. The power values are set to 1 and 3 to pertain to the ratio between the power consumption in the two different encryption techniques. These values are the same for both players. We set R_1 and R_2 to be 10 and 5 depending on the opponent's actions. We choose these values to be higher than the power values in order for the utilities to be positive. The transition reward is set to 5 and 10 for switching to another state defined by another key or another technique, respectively.

First, we run simulations when there is no transition cost, $q = 0$. The equilibrium strategies for both the attacker and defender are shown in Table 3.1. Note that actions a_1, a_2 represent the selection of two keys for the same encryption technique and actions a_3, a_4 represent two keys for another technique. Table 1 shows the probabilities over all actions for each player. These probabilities show how players should select actions in every state. For the defender, if it starts in state s_3 then it should move to state s_1 with the highest probability and move to state s_2 with a very similar probability. This is because the defender will change the technique and so gets a higher transition reward. We can see that the probability of moving to the same state is always very low and can reach 0 as in state s_1. The probability of moving to a state that has a similar encryption key is always less than that of moving to a state with different technique as the transition reward will be lower. For the attacker, the probability of attacking the same technique that is used in the current state is always higher than attacking any other technique.

In Figure 3.1, we show the effect of the discount factor on the defender's utility at equilibrium in every state. First, we can see that all utility values at all states increase as the discount factor in-

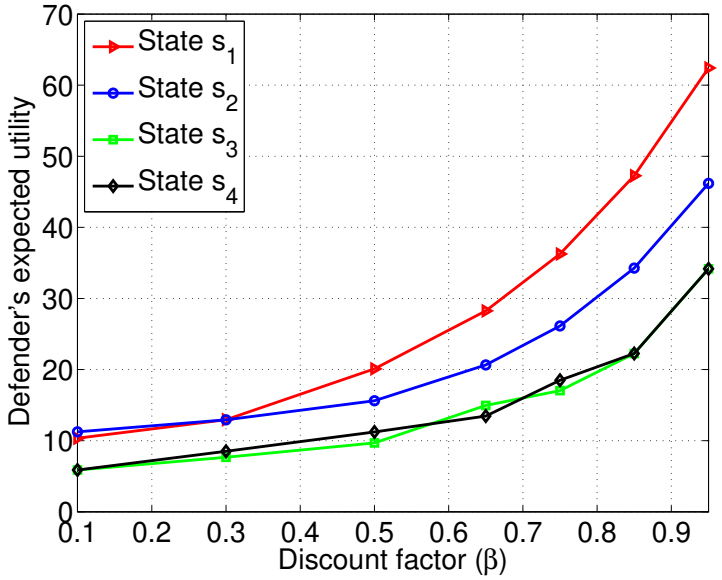


Figure 3.1: The defender's expected utility in each state against discount factor β.

creases. This is due to the fact that increasing the discount factor will make the defender care more about future rewards thus choosing the actions that will increase these future rewards. Figure 3.1 also shows that the defender's values at states 1 and 2 are higher than at states 3 and 4. This because states 1 and 2 adopt the first encryption technique which uses less power than the encryption technique used in states 3 and 4. The difference mainly arises in the first state before switching to other states and applying the discount factor. Clearly, changing the discount factor has a big effect on changing the equilibrium strategy, and, thus, the game will move between states with different probabilities resulting in a different accumulated reward.

In Figure 3.2, we study the effect of applying the proposed MTD technique against the case when the defender decides to use equal probabilities over its actions in each state, i.e., all entries equal 0.25 as there are four actions per state. Figure 3.2 shows the percentage of increase in the defender's expected utility. We can see that the minimum increase is non-zero which means that the defender will not gain from deviating from equilibrium strategies. Moreover, at high discount factor values, i.e, $\beta > 0.75$, the percentage increase is higher than that at lower β values in all states. The percentage increase ranges from 5% to about 40% at $\beta = 0.75$ depending on the state, and it can reach values between 20% and above 40% at $\beta > 0.95$. This is due to the fact that, at higher β values, future state transitions have higher impact on calculating equilibrium strategies and the defender considers more state changes in the future. This makes equilibrium strategies differ more from equal probabilities. For other β values, the percentage increase depends on how different the equilibrium strategy from the equal allocation scheme.

In Figure 3.3 we study the effect of changing the power on the defender's expected utility at equilibrium. We study three cases, first when the power required for technique 1 is less than the power required for technique 2, similar to the previous experiments. Then, we study the cases in which

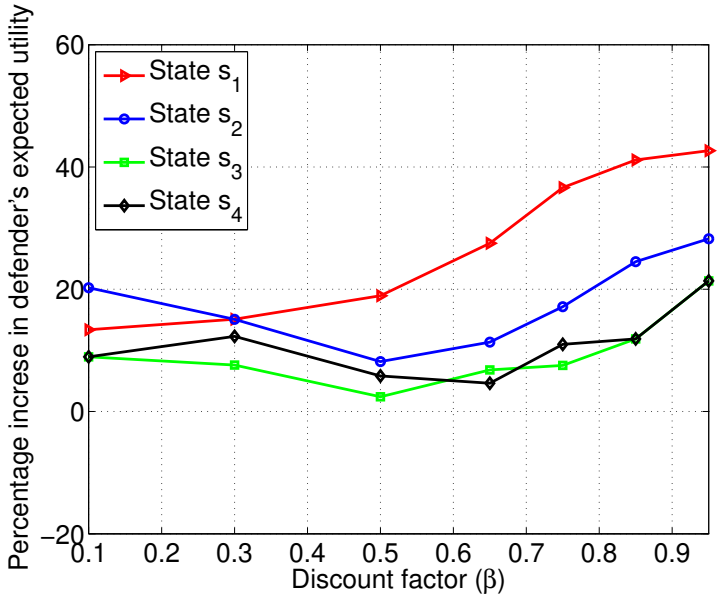


Figure 3.2: Percentage increase in the defender's expected utility when using the equilibrium strategy and when using equal probabilities over actions. This is shown in each state as function of the discount factor β.

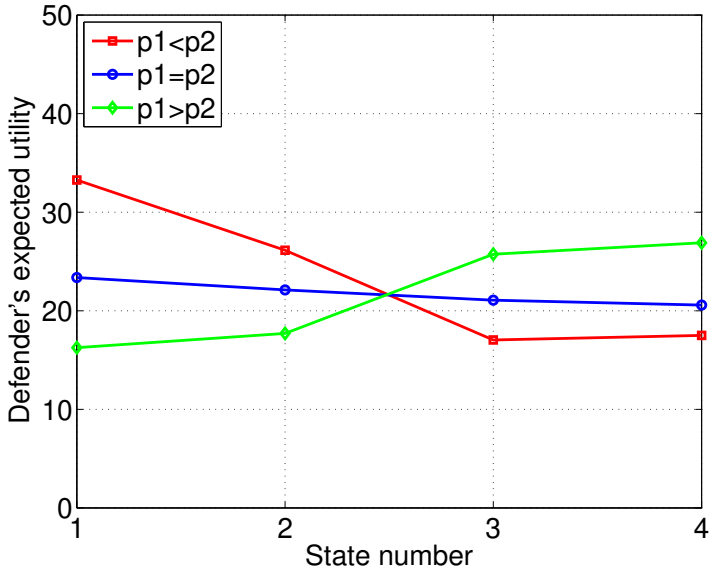


Figure 3.3: The defender's expected utility in each state for different techniques power combinations.

they are equal and in which technique 1 requires more power than technique 2. Here, we set $\beta = 0.75$. From Figure 3.3, we can see that, when the first technique's power is less than the second one, the defender gets higher reward at states s_1 and s_2 than at states s_3 and s_4. This stems from the fact that, at states s_1 and s_2, the defender begins the game using technique 1 (lower power)

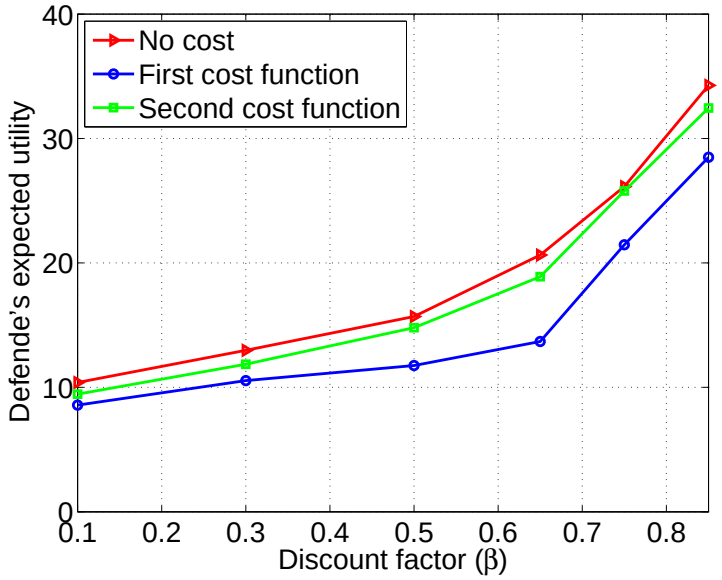


Figure 3.4: The defender's expected utility in state s_2 against discount factor β for different cost functions.

thus getting a higher reward. A similar result can be seen when the defender gets a higher reward at states s_3 and s_4 when the technique used in these states needs less power. When the two techniques use the same power, we can see that the defender's expected utility is almost the same for all states. Figure 3.3 clearly shows the effect of first state parameters on the expected utility.

In Figure 3.4, we study the effect of adding cost to the defender's expected utility. We calculate the expected utility at different discount factor values at state s_2. Clearly, the expected utility is higher when no cost is applied. When applying cost function $C(q, n) = q \cdot \ln(n + 1)$, the utility is barely reduced. When applying the cost function $C(q, n) = q \cdot n$, we notice a significant decrease in the defender's expected utility. In our problem, the cost function $C(q, n) = q \cdot n$ will be more suitable as the other cost function does not show a significant change.

3.5 Summary

In this chapter, we have studied the use of MTD in a wireless network security problem. We have formulated the problem using a non-zero sum stochastic game theory model in which the defender controls state transition. The next state is determined only by defender's actions which is suitable for MTD cases where the defender want to change system parameter's before the attacker can reveal them. This property of the game ensures that the game will always have an equilibrium point. We have provided the mathematical model for deriving an equilibrium in such games. We then provided a novel way to define cost in MTD systems that depends on the number of consecutive changes in system parameters. We have shown two different functions to define cost and have proved that the game will still have equilibrium. Simulation results have shown that this

model helps the defender to get higher expected utility in all system state than the case of assigning equal probabilities over different actions.

Chapter 4

On the Cybersecurity of m-Health IoT Systems with LED Bitslice Implementation

4.1 Background, Related Works, and Contributions

In section 1.2.2, the importance of IoT within smart cities and their new challenges to the cyber security of CI, was discussed. The IoT, in general, will be a major enablers for a variety of smart services that range from large-scale sensing to smart transportation [116] and healthcare [117]. M-Health systems that are wireless-enabled healthcare systems will be one of the primary services supported by the IoT system that will provide them with pervasive Internet access [118]. As discussed in [117], m-Health IoT systems include a number of smart devices and sensors that monitor a patient's medical conditions such as blood pressure, pulse or body temperature. The measured data is then sent to remote physicians via an access point or a gateway [119]. This gateway is responsible for collecting and sending the data [120] as well as providing wireless connectivity, via multiple networking interfaces, to the m-Health IoT devices.

This pervasive wireless connectivity for small, m-Health IoT devices, will bring forward new security challenges and vulnerabilities. Malicious attacks can now leverage the connectivity of these devices to launch remote attacks and potentially access the patients' critical data that is being transmitted by the m-Health devices. Taking these attacks into consideration, security and privacy constitute key concerns in all IoT systems. The work in [121] highlights the main security issues in the IoT while outlining the main existing solutions that have been developed to maintain the confidentiality, authenticity, and integrity of data in IoT. The authors discuss security features that need to be applied in a security architecture of four levels distributed between the devices and the cloud. Device authentication, data encryption, and key agreement are highlighted as the most critical security requirements that need to be addressed at the devices side. Note that some wireless security approaches, e.g., physical layer security [122] cannot be used with the IoT due to its heterogeneous nature.

Recently, such security requirements received significant attention in the literature due to the specific nature of the IoT. The huge number of heterogeneous limited-resources devices in the IoT complicate the security mechanisms. The limited resources make it hard for the IoT devices to run complex security algorithms. Hence, lightweight encryption techniques are seen as the cornerstone of IoT security. In [123], the authors present a lightweight encryption method to authenticate RFID tags at the readers. The work in [124] evaluates two major types of attribute-based encryption on different IoT devices. This work shows that the performance of attribute-based encryption cannot be readily deployed in small IoT devices, due to resource constraints. A more recent work in [125] proposes a lightweight attribute-based encryption scheme based on elliptic curve cryptography. The scheme is shown to have low communication overhead provided that the number of attributes remains small.

While data encryption is not a sufficient security mechanism for the IoT [126] as it does not protect against insider attacks, that is not the case in m-Health IoT. In the IoT, both data encryption and device authentication [127] are taken into consideration. However, as the devices in an m-Health system are usually operated around the patient and known to the gateway, security mechanisms are oriented more towards data encryption for enhancing data privacy. In [119], the authors provide a prototype for applying asymmetric, public key, encryption in an m-Health IoT system. Due to the high computational power of public key encryption, it is applied at the level of the gateway. The more recent work in [128] demonstrates the benefits of applying cloud computing in an m-Health system by using hybrid encryption schemes. In hybrid schemes, symmetric secret key encryption, which is known to have low computational power, is used between devices and the gateway while a public key, which consumes significant power but provides more security, is used between the gateway and the cloud. However, in all these systems, using secret key encryption can be problematic if the key was revealed by an attacker through any of the known attacks.

One promising technique to improve a system's security is the so-called moving target defense (MTD) [52]. MTD is the concept of continuously randomizing a system's configuration in order to increase the uncertainty and cost of an attack. In the IoT, the system's configuration can essentially include encryption keys, network parameters or IP addresses. While applying MTD improves a system's security, it can also incur some costs that reduce the overall performance. The authors in [129] applied MTD by frequently changing the IP address of IoT devices to increase the security. Security improvement and network latency were studied for implementing MTD over low-powered personal area networks. The work in [38] applies MTD using a stochastic game between an attacker and a base station acting as a defender. Multiple encryption techniques with multiple shared secret keys are implemented at the nodes. The security benefit as well as the MTD costs are studied in this scenario. However, existing MTD works such as [129] and [38] are not designed for m-Health IoT systems and do not provide specific implementations of the encryption system.

The main contribution of this chapter is an MTD security framework tailored to the unique nature of m-Health IoT systems. The framework uses a hybrid encryption scheme in which secret keys are used to encrypt the data sent from the devices to the gateway, and a public key to encrypt data from the gateway over the Internet. The proposed MTD scheme is applied by frequently changing the secret keys used in the communication between the devices and the gateway. The gateway

takes the decision to update all the keys in the network hence allowing each device to calculate its new key and start using it. The new encryption key is generated by encrypting the old key using another pre-shared key. Hence, only two secret keys need to be pre-shared between each device and the gateway. A case study is provided to study the effect of applying MTD on an enhanced (in terms of performance) real system. In this system, a lightweight encryption technique, LED [130], is used for encrypting the data. As gateways typically apply performance improvement techniques to speed up the process of decrypting the collected data, we propose a new bitslice implementation for LED that can be used at the gateway. To the best of our knowledge, this is the first 64-bit bitslice implementation for LED algorithm. We also provide a modified 32-bit version suitable for 32 bit registers. The system is tested on a virtual 64-bit ARM Cortex-A processor and the results show that the bitslice implementation consumes half of the processor's instructions compared to the original LED implementation. Results also show that using MTD and bitslice does not yield any significant degradation in the system's performance when some packets are missed compared to the original implementation.

The rest of the chapter is organized as follows. Section 4.2 presents the proposed security mechanism and the MTD scheme. In Section 4.3, the bitslice implementation is presented in detail, and the metrics used to measure the performance improvement are discussed. Performance evaluation using a real-world implementation are presented in Section 4.4. Finally, conclusions are drawn in Section 4.5.

4.2 Encryption model in M-Health IoT system using Moving Target Defense

Consider an m-Health network consisting of a number of smart devices and sensors, referred to as nodes, that monitor a patient's medical condition and send the measured readings to a gateway. The gateway will send the collected data over the Internet to a remote hospital or a clinic. Unless there is an emergency that needs to be reported, we assume all the devices are synchronized to send frequent updates about what they sense or measure. The frequency of sending the updates depends on the medical situation and the criticality of the patient's health status.

All the data sent from the nodes is encrypted at each node before it is sent to the gateway. The gateway decrypts the received data and re-encrypts it using a more powerful encryption algorithm to be sent over the Internet. Due to the resource limitations of the IoT nodes, a lightweight encryption technique should be used to encrypt the data at every node. Typically, symmetric algorithms, which use a pre-shared secret key, are less power demanding than asymmetric algorithms, which use two different keys known as public and private keys. Therefore, symmetric lightweight algorithms are more suitable for IoT nodes. A secret key must be shared between every node and the gateway prior to connecting to the Internet. At a given node i, a plaintext P is encrypted using node i's secret key K_i to get the ciphertext $C = E_{K_i}(P)$. The gateway, which is a computationally capable device, will then use an asymmetric algorithm to decrypt the data and send it over the

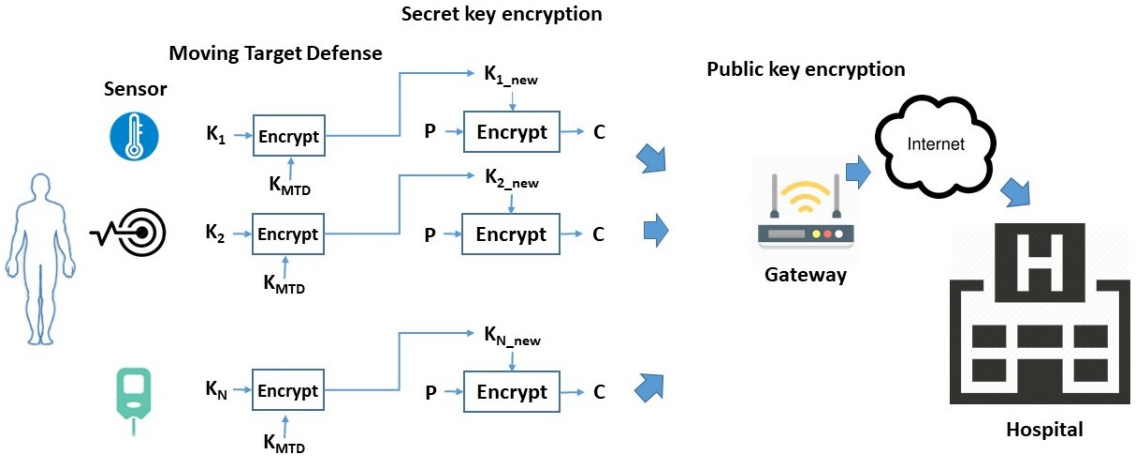


Figure 4.1: MTD security mechanism for m-Health IoT system.

Internet. This makes the m-Health encryption system, a hybrid system combining both symmetric and asymmetric encryption algorithms.

Symmetric encryption algorithms can be vulnerable to some attacks like brute force attacks, known plaintext attacks, chosen plaintext attacks, and differential cryptanalysis attacks. The goal of all such attacks is to reveal the secret key used in the encryption allowing the adversary to access and read the private data or even send fake data impersonating another node by using its key. To mitigate the effect of a successful attack and make the system more resilient, we use MTD by frequently changing the secret key used in the communication between every node and the gateway. The gateway decides to update the keys and informs the devices which should start using the new keys immediately. The time needed to apply MTD, i.e., initiating new keys is decided by the gateway depending on the frequency of sending new packets from the devices. This potential time delay yields a trade-off between increasing the attacker's chance to perform a successful attack and incurring more cost by frequently changing the keys as discussed in Section 4.3.

New secret keys are calculated by encrypting the old keys, within each node, using another pre-shared secret key referred to as the MTD key. Given a key K_i used by a node i, the new key will be given by $K_{i_{\text{new}}} = E_{K_{\text{MTD}}}(K_i)$, where K_{MTD} is the pre-shared MTD key. Consequently, both the gateway and the device can get the new key without having to share any additional keys. Figure 4.1 shows the proposed model for m-Health security mechanism. Note that each node can use a different MTD key.

The use of MTD in this mechanism will increase the uncertainty on any attacker, thus improving the security of the system. This is due to the fact that there is no fixed key, no fixed time to change the key, and the new keys are generated locally to eliminate the possibility of being intercepted. Even if the attacker was able to reveal one or more keys, it will not be able to reveal the new keys as they are generated using the MTD key which is stored locally at each node. Therefore, the attacker will lose any privilege once the keys are updated and will have to start a new attack.

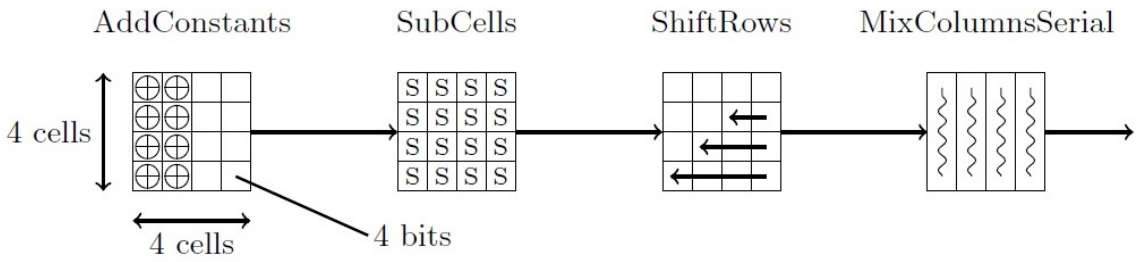


Figure 4.2: The four operations in a single LED round.

Finally, the security mechanism proposed here does not require any device-level hardware modification. It only requires a small software modification to add the pre-shared keys and to allow the nodes to respond to the gateway signals of changing the key. Next, we present the practical implementation of the proposed mechanism. We define a performance improvement technique to be used by the gateway in decrypting a number of packets at once and then study the effect of applying MTD on the performance.

4.3 Case Study: LED Bitsliced Implementation

4.3.1 LED Block Cipher

Lightweight encryption techniques are designed for resource-constrained devices. Some techniques target the hardware such as area on the chip, power, or energy consumption while others provide light software such as low memory and small code size. In this case study, we choose to implement LED block cipher [130]. LED is hardware-oriented which provides the smallest silicon footprint in its class of block ciphers with a reasonable performance. LED was chosen for this case study as hardware consumption is more critical because the software performance can be improved by some techniques as shown later.

In terms of design, LED's design is similar to the design of advanced encryption standard (AES) schemes. The main difference between LED and AES is that LED uses no key schedule and the same key is applied every round. The user-provided key can range from 64-bit to 128-bit. Increasing the key length will increase the security and the power consumption as well. In this work, since we adopt MTD and we depend on the key change as a defense, no need to consider longer keys which consume more computational power and hence a 64-bit key will be suitable. LED applies rounds like AES. In each round, four operations are applied to the state, which are: AddConstants, SubCells, ShiftRows, and MixColumnsSerial. The state refers to the current input to each round, which is initially the plaintext. Figure 4.2 from [130] shows the four operations in each round.

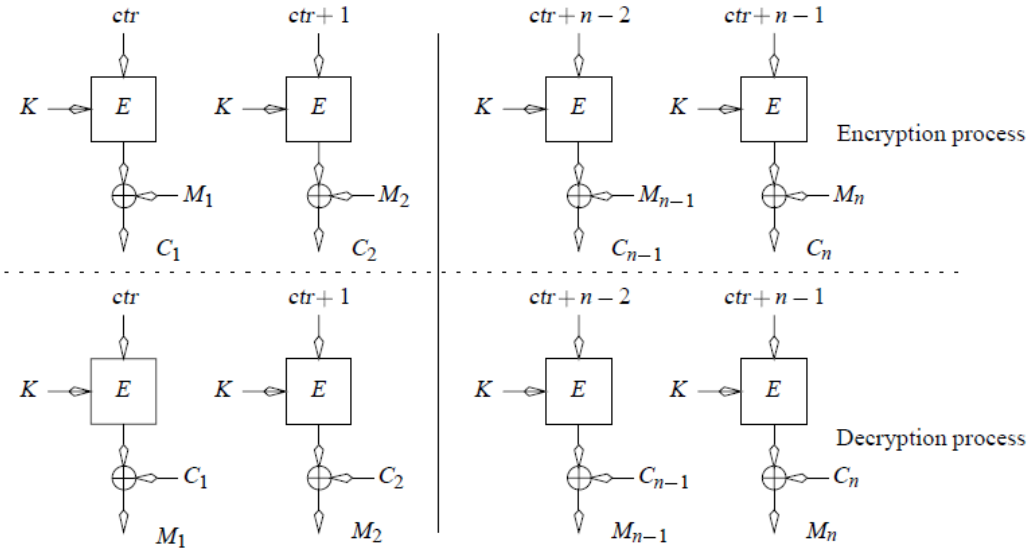


Figure 4.3: CTR mode encryption and decryption.

In AddConstants, some predefined constants are combined with each state's bits. SubCells is used to replace the bytes of each cell in the state using an S-Box. ShiftRows is used to rotate the cells to the left a number of times depending on their row. Finally, MixColumnsSerial is used to multiply the cells by another predefined matrix and the multiplication is done over a defined Galois field$GF(2^4)$ with the irreducible polynomial $X^4 + X + 1$. LED applies this round four times to the same state before adding (XOR) the key again. In the case of LED-64, this process is repeated eight times, i.e., in total 32 rounds are applied to the state. In the case of LED-128, the key is split into two 64-bit portions each applied after four rounds, and then the whole process is repeated six times, i.e., 48 rounds are applied to the state.

4.3.2 LED Decryption

As any block cipher, LED decryption could be accomplished using the counter (CTR) mode. Fig. 4.3, from [131], shows the encryption and decryption processes in CTR mode. In this mode, an incremented counter value is ecrypted each time using the provided key. Then the output is Xored with the plaintext to get the ciphertext. Decryption is the reverse process where the cipher text is Xored with the encrypted key to get the plaintext.

However, as the CTR mode is not applicable in all scenarios, we decided to implement the actual decryption process. Basically, LED consists of four operations as mentioned bfore and these operations need to be used in order. In decryption process, we need to apply the following operations with the order: InvMixColumnSerial, InvShiftRows, InvSubBytes, and InvAddConstants which is the reverse order of the encryption process.

Table 4.1: LED SBox and inverted SBox

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S(x)$	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2
$S^{-1}(x)$	5	E	F	8	C	1	2	D	B	4	6	3	0	7	9	A

In the MixColumnSerial, each state is multiplied by a fixed matrix:

$$\boldsymbol{M} = \begin{bmatrix} 4 & 1 & 2 & 2 \\ 8 & 6 & 5 & 6 \\ B & E & A & 9 \\ 2 & 2 & F & B \end{bmatrix}$$

The multiplication occurs over the field $GF(2^4)$. In InvMatrixColumnSerial, we need to multiply by the inverse of the previous matrix, which is:

$$\boldsymbol{M^{-1}} = \begin{bmatrix} C & C & D & 4 \\ 3 & 8 & 4 & 5 \\ 7 & 6 & 2 & E \\ D & 9 & 9 & D \end{bmatrix}$$

The rest of the operation is the same as the original one. The field multiplication does not change.

In the original ShiftRows, each cell is shifted some positions to the left based on the row number. InvShiftRows, is the same but shifting the cells to the right with the same criteria of shifting, i.e., row 0 is shifted 0 times, row 1 is shifted 1 time and so on.

The only difference between subcells and Invsubcells is to use the inverted Sbox for substitution instead of the original Sbox. Table 4.1 shows the original Sbox denoted as $S(x)$ and the inverted Sbox denoted as $S^{-1}(x)$.

4.3.3 LED Bitsliced Implementation

Bitslicing is the process of slicing the data into its bit level and performing the required operations on these bits. Bitslicing is designed for a specific processor size, e.g., a 32-bit or a 64-bit processor, which essentially maps to the size of the data types that the processor can handle. In this case study, we design a bitslicing scheme suitable for a 64-bit processor which can typically be found in gateways and modern mobile devices. Although bitslicing is not an optimization technique, it can offer a great flexibility to improve the performance if it is used appropriately.

LED encrypts a 64-bit plaintext. These 64-bits are organized in a state as a 4×4 matrix of nibbles

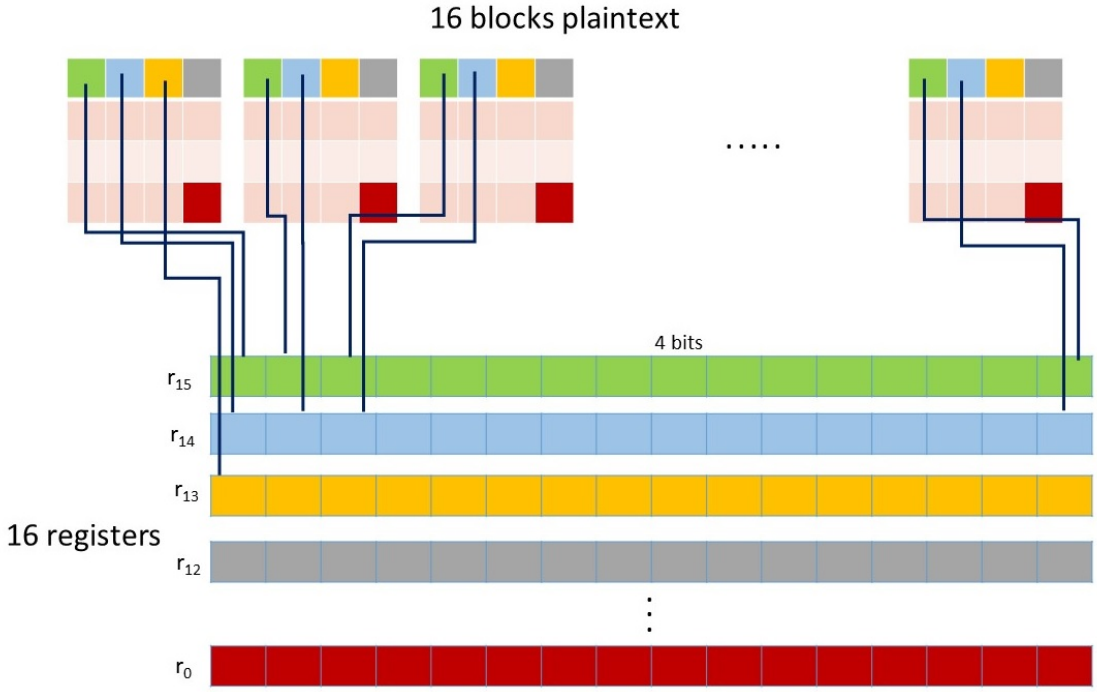


Figure 4.4: Bitsliced representation of 16 plaintext blocks into 16 64-bits processor's registers. Colors represent data that is stored in the same register.

and each nibble consists of 4-bits. In the design of bitslicing implementation, we take every nibble, 4-bits, to be the minimum chunk that will be processed. Every nibble will be stored in a different processor's register, hence 16 blocks of plaintext should be processed at the same time to make use of the 64-bits registers. In an m-Health IoT network, as the gateway receives data from multiple devices, it is very likely to have 16 or more plaintexts at a given time. Processing data in such different arrangements, requires modifying all the operations of the original LED. In addition to the four operations of LED and the key adding step that must be modified, an initialization operation need to be executed to transform data blocks into the desired arrangement in processor's registers. Next, each modified operation is discussed in detail.

• *BitTranspose*: The 16 blocks of plaintext as well as their corresponding 16 64-bit keys are transposed first to a form suitable for bitslicing. Sixteen 64-bits registers are needed. Figure 4.4 shows our bitslice implementation arrangement. Every first nibble in each plaintext is stored in the first register, i.e., (r_{15}) at consecutive locations. The next nibbles are stored in the second register r_{14} and so on to fill all the remaining registers.

• *AddRoundKey*: The encryption key is added first before applying other operations. In AddRoundKey, every plaintext nibble is XORed with the corresponding nibble in the key. As the plaintext and the key are transposed using the same mechanism, every two nibbles need to be XORed will be in the same locations of the transposed plaintext and the transposed key. Therefore, a direct

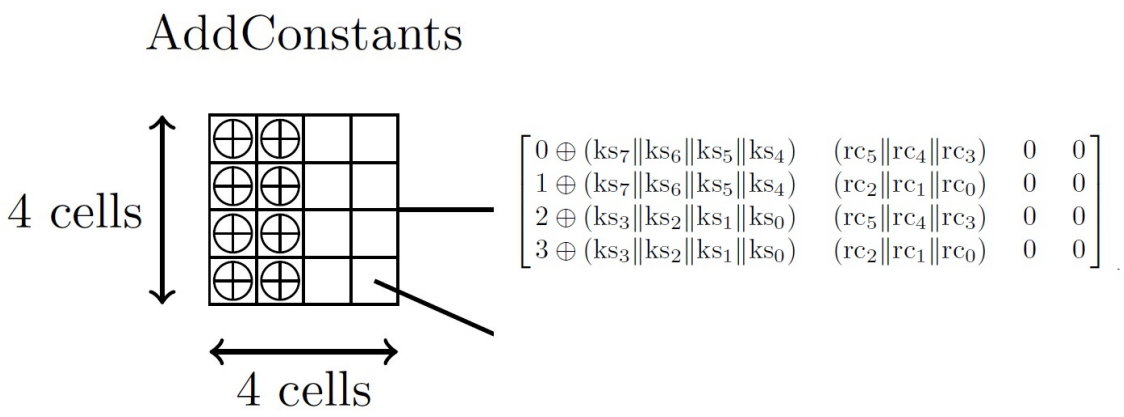


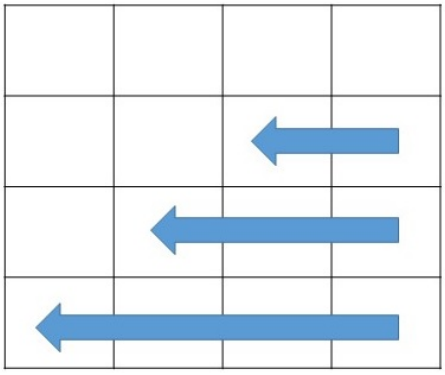
Figure 4.5: AddConstants operation of LED.

XOR operation can be applied to every pair of transposed registers which gives a total of 16 processor instructions to apply round keys. The original algorithm deals with separate nibbles and needs to XOR every nibble separately, which requires 16 instructions for every plaintext and, thus, 256 instructions for the 16 plaintext blocks. This process will be repeated 32 times before each round, thus the transposed representation reduces significantly the number of instructions required to apply round keys.

• *AddConstants*: In AddConstants, half of the nibbles are modified as shown in Figure 4.5. Therefore, only eight out of the sixteen registers need to be updated. Instead of using the original constants provided by LED, new constants suitable for the bitslice representation need to be calculated from the original constants. These new constants will be stored and used directly each round. Each nibble of the first column, in each state, is XORed with one of four different values. These values are constant and, hence, can be computed in advance. As each register holds the same nibble in different plaintext blocks, each of these four values is concatenated sixteen times to fit all of the nibbles. For example, register r_{15} will be XORed with the new value 4 concatenated 16 times. The same is applied to registers r_3, r_7, and r_{11} which hold the nibbles of the first columns in all plaintext blocks. Nibbles in the second column, in each state, are XORed with specific three bits of the round constants. The values of these three bits, concatenated sixteen times, are stored in advance and, hence, can be used directly in the bitslicing implementation. This modification can save only a few processor instructions but is necessary for the bitslicing implementation.

• *SubCells*: In SubCells, each nibble is replaced by a corresponding value from the S-Box. As we still deal with a whole nibble, no modification need to be applied to the S-Box. Each nibble was separated from the register and then substituted from the S-Box which requires twice the processor instructions used in the original implementation.

• *ShiftRows*: In ShiftRows, each row of the state is shifted to the left by a multiple of four bits as shown in Figure 4.6. The figure also shows which register is used to store each nibble in the state. We made use of the fact that each register in this implementation holds nibbles from the same



r_{15}	r_{14}	r_{13}	r_{12}
r_{11}	r_{10}	r_9	r_8
r_7	r_6	r_5	r_4
r_3	r_2	r_1	r_0

Figure 4.6: Registers considered for swapping in ShiftRows operation of LED.

location in each state. As such, all the nibbles in the register need to be shifted by the same amount. Therefore, instead of doing actual shifting, we need to just swap the registers. For example, register r_{10} is placed in r_{11} then r_9 is placed in r_{10} then r_8 is placed in r_9 and, finally, r_{11} is placed in r_8 and so on for similar registers. We need one temporary register for the swapping process, and five assignment instructions. This can save a lot of instructions from the original operations where nibbles were considered separately.

• *MixColumnSerial*: In this operation, a constant matrix is multiplied by the state matrix. Each row in this matrix is multiplied by a column in the state matrix to update one nibble in the state matrix. As multiplication is done nibble by nibble, we had to define a new MixColumn operation that extracts nibbles from the registers and use them in the multiplication process. Even though a number of extra instructions are needed for separation, the updated nibbles can be calculated for the all the 16 registers at one iteration. This parallel calculation allows, in total, saving a significant amount of instructions when compared to the original case in which each plaintext is processed separately. For example, nibbles from registers r_{15}, r_{11}, r_7, and r_4 are processed together in the multiplication process which is different from the original implementation in which multiple iterations are needed. The rest of the columns are processed in the same way.

Finally, another version of this bitslicing was designed to suit 32-bit processors. A 32-bit version is obtained by decrypting 8 data blocks instead of 16. The 8 blocks are stored in 16 32-bit registers in the same way discussed in the 64-bit version. Similarly, every nibble from the plaintext is stored in a different register. The rest of the operations will follow as the 64-bit version but dealing with a smaller size of input. This implementation could be used either for 32-bit processors or 64-bit processors that support 32-bit registers. This implementation is beneficial for the gateway when applying MTD as shown in Section 4.4.

4.3.4 Performance Metrics

The performance of bitslicing is maximized when the total number of data blocks is available at the same time, this is sixteen plaintext blocks in our implementation. Bitslicing uses a constant number of processor's instructions whether 16 blocks are available or not. Here, we calculate the average number of instructions needed to decrypt a plaintext block, a as follows:

$$a = \frac{N}{b},$$

where N is the total processor instructions and b is the actual number of blocks that were encrypted and is bounded by the maximum number of blocks allowed by the design which is 16 in our implementation. Clearly, if we have fewer than sixteen blocks, the average number of instructions per block will increase and degrade the performance.

Another metric that we consider is *the cost of applying MTD* in m-Health IoT systems that consist of heterogeneous devices differing in their computational capabilities. Here, when the gateway asks the devices to update their keys, they can have different response times. Hence, they may encrypt new packets using the old encryption key while the gateway is expecting data encrypted with new keys. This incurs a processing cost at the gateway, which is the wasted processor's instructions to decrypt data with wrong keys and the extra instructions required to decrypt again using the old keys. We measure the wasted instructions as the difference between the number of instructions used to decrypt the maximum number of packets that was expected and the number of instructions used for the correctly decrypted packets. Th number of instructions needed to re-decrypt packets using old keys will differ according to the number of missed packets. The gateway is given the option to re-decrypt the missed packets using either the original LED implementation, the 32-bit slicing version, or the 64-bit bitslicing version. This choice depends on the implementation that will use the least number of instructions to re-decrypt the missed packets. The choice of the re-decryption technique and the mathematical formulation for the cost are discussed, in detail, in Section 4.4.

4.4 Evaluating LED Bitsliced Implementation

For our evaluation, we use an ARM Cortex-A 64-bit processor as the target processor to evaluate our implementation. ARM 64-bit processors such as Cortex-A53 and Cortex-A57 can be found in many mobile devices. Evaluating the code on a real processor is challenging as other operations can affect the measurements. Therefore, we use the ARM development studio (DS-5) [132] which gives the ability to create a virtual processor emulator for a specific ARM processor then run the code on it. We use Cortex-A53 as our implementation processor, and we adjust the compiler optimization flags to the maximum performance in all the next experiments. In Cortex-A53, we can use both 64-bit registers or 32-bit registers which allows to use both our 32-bit and the 64-bit bitsliced implementations.

The designed bitslice implementation presented in the previous section is suitable for the encryption process, however, what is typically done on the gateway is the decryption phase. Therefore, we had to invert the encryption algorithm to get the decryption scheme. The inverted operations are applied in a reverse order to the original operations, i.e., InvMixColumnsSerial, InvShiftRows, InvSubCells, and InvAddConstants which are the reverse operations applied in order. The inverted operations are designed as follows. In InvMixColumnsSerial, the state is multiplied by the inverse of the constant matrix that is used in MixColumnsSerial. In InvShiftRows, the rows of the states are shifted to the right with the same criteria of shifting as in ShiftRows. In InvSubCells, the inverted S-Box is used for substitution. Finally, in InvAddConstants, the same round constants as AddConstants are used but provided in the reverse order of rounds. The bitslice is then applied in the same way as the encryption process.

First, the bitslice implementation is evaluated to measure the reduction in the number of processor instructions when applying bitslicing. We used the reference LED implementation provided by the work in [133]. Figure 4.7 shows the average number of instructions required to decrypt one block of plaintext data when different number of plaintext blocks are available. We assume that every block is received from a different device. We compare the original LED implementation, our 32-bit bitsliced version, and our main 64-bit bitsliced implementation. Note that the 32-bit version processes only 8 blocks at a time and, thus, we apply it twice for more than 8 blocks. From Figure 4.7, we can see that the original implementation has an approximately constant average. In fact, there is only a small difference when the number of packets is small due to the processor initialization instructions having a higher effect on the total number of instructions. However, this difference is not significant. The 32-bit bitsliced version has the lowest average when there are 3 to 8 packets to be decrypted. Our bitsliced implementation requires half of the processor instructions required by the original implementation when decrypting 8 packets. The increase after decrypting 8 packets happens because the processor will start over to apply the bitslicing again and, thus, needs to execute more instructions. Therefore, applying the 32-bit version twice consumes a little bit more instruction than the 64-bit implementation. At 16 decrypted packets, our 64-bit implementation consumes half of the processor instructions compared to the original implementation.

Finally, the results in Figure 4.7 allow the gateway to determine the algorithm that will be used to decrypt the number of available packets. If there are less than 3 packets, the original algorithm is preferred. The 32-bit implementation should be used when there are 3 to 8 packets. The 64-bit implementation is superior for more than 8 packets of data.

Next, we discuss the cost of applying MTD when bitslicing is used at the gateway. Note that bitslicing itself is known to increase the code size on the device, i.e., the gateway. However, as the gateway is assumed to be a computationally capable device, the increased code size will not be problematic so it will not be considered as a cost here. The focus will be on the number of wasted (or additional) processor instructions. Clearly, if all the devices will send their next packets with the updated key, no cost will be incurred. However, when some packets are received encrypted with the old key, the gateway will decrypt them using the new keys which will result in wrong packet formats. The gateway will conclude that the key is not updated yet in these devices and will

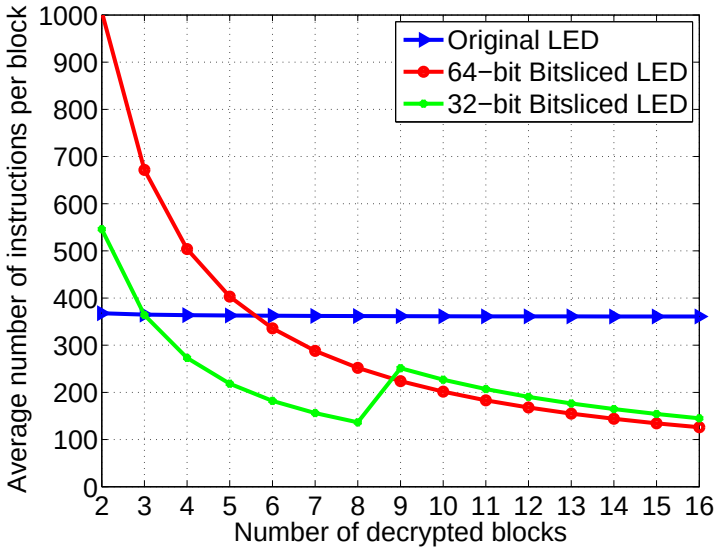


Figure 4.7: Average number of instructions required to decrypt one block in the original LED implementation, 32-bit bitsliced applied twice, and the 64-bit bitsliced implementation. The number of instructions is normalized by 1000 for an easier representation.

re-decrypt these packets using the old keys. Thus, we can formulate the cost as follows:

$$C = \left\{ \begin{array}{ll} (b_{\max} - b) \cdot (\frac{L_{32}}{8} + R), & \text{for } 3 < b_{\max} \leq 8, \\ (b_{\max} - b) \cdot (\frac{L_{64}}{16} + R), & \text{for } 8 < b_{\max} \leq 16, \end{array} \right\}$$

where $b_{\max}$ is the maximum number of packets expected by the gateway, b is the number of successfully decrypted packets, and L_{32} and L_{64} are the total process's instructions for the 32-bit and 64-bit bitsliced versions, respectively. The decryption cost R is determined by the number of re-decrypted packets $b_{\max} - b$. If the number is 3 or less, the gateway will use the original LED to decrypt each packet individually, if the number exceeds 3 either version of bitslicing will be used and R will equal L_{32} or L_{64}.

Figure 4.8 shows the cost in terms of gateway processor instructions if some devices send a single packet with the old encryption key. Three cases are considered when one quarter, half, and three quarters of the devices will send one packet with the old key. In case only a quarter of the devices wrongly encrypt one packet, we observe that the increase rate in the cost is less after twelve packets. This is due to the fact that, after twelve packets, the quarter will exceed three packets and, hence, the 32-bit bitslice version will be used to re-decrypt the missed packets thus reducing R as well as the total cost. A similar behavior can be seen for the case of half of the devices, where the cost increases at a slower rate after eight packets when the 32-bit bitslice version is used. However, in the case of three quarters of the devices, the increase in cost is higher after eight packets as the 32-bit version was used until eight packets, i.e., $R = L_{32}$ and the 64-bit version will be used after that consuming more processor's instructions as $R = L_{64}$.

It is interesting to note that the maximum cost according to this implementation is when all the

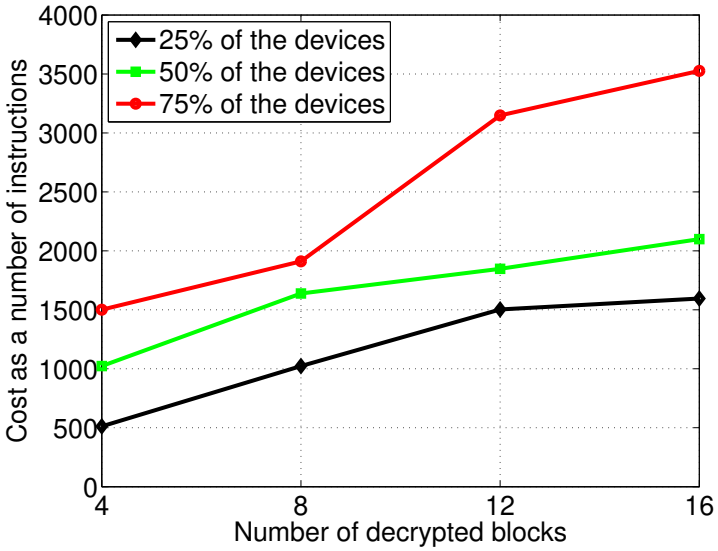


Figure 4.8: The cost of applying MTD in a system with late response devices. The figure shows three cases for the percentage of the devices that will have a delay. The number of instructions is normalized by 1000 for an easier representation.

sixteen packets need to be re-decrypted, i.e., $C_{max} = 2 \cdot L_{64}$. As L_{64} equals half of the instructions required by the original LED implementation as shown in Figure 4.7, then the maximum cost equals the same number of processor instructions of the original LED implementation, if no packet is missed. Missed packets due to using MTD, with the original LED implementation, are re-decrypted individually causing more cost. Thus, the worst-case cost of applying MTD with bitslicing is bounded by the best-case, no cost, of applying MTD with the original LED implementation.

4.5 Summary

In this chapter, we have proposed a novel security mechanism for m-Health IoT systems. The mechanism depends on using secret keys between the devices and the gateway and then applying MTD by frequently changing the encryption keys used in the network. The new key is calculated by encrypting the old key using another pre-shared secret key known as the MTD key, hence only one key needs to be shared between the gateway and each device. We have applied this mechanism to a system which involves a performance improvement technique for the encryption algorithm using bitslicing. We have formulated a 32-bit and 64-bit bitslicing implementations for LED, a light weight encryption technique. We have also defined performance metrics for the system including the cost for applying MTD. We have used a virtual processor to evaluate both bitslicing implementations and the cost of applying MTD. Implementation results have shown that the bitslicing implementation significantly outperforms the original implementation of the encryption algorithm. We have also discussed the optimal packet number for using both bitslicing

versions. Results have also shown that the worst-case cost for applying MTD is bounded by the number of instructions in the original LED implementation.

Chapter 5

Resilient Critical Infrastructure: Bayesian Network Analysis and Contract-Based Optimization

5.1 Background, Related Works, and Contributions

As we discussed in Chapter 1, critical infrastructure (CI) are vital to modern day cities and communities [26]. As such, maintaining proper operation of CIs, in presence of failures or security threats, is therefore a critical challenge. We discussed the notions of reliability and resilience and pointed out the importance of studying CI resilience.

In light of the resilience challenges discussed in section 1.3, the literature is in need of a general framework to evaluate the resilience of different CIs and to help in designing general resilience improvement techniques. Some studies in the literature, e.g., [28, 83, 84, 134], proposed general resilience frameworks for CIs. However, the approaches proposed in this prior art mostly evaluate the CI resilience based on satisfying a number of pre-determined criteria as detailed in the next section. The resilience measures based on these properties fail to capture the effect of different disruptive events on the CI. In contrast, here, our goal is to introduce a general framework to evaluate and improve CI resilience based on the effect of disruptive events on the CI's components. Prior to providing our key contributions, we will first review existing related frameworks and techniques in the next section to pinpoint their limitations.

5.1.1 Related Work

Critical Infrastructure resilience has recently attracted significant attention [28, 83, 84, 134, 135]. In [28] the authors considered four properties for resilience: robustness, redundancy, resource-

fulness, and rapidity and the resilience was quantified using four interrelated dimensions: technical (physical), organizational, social, and economic. The authors in [134] proposed a resilience framework that seeks to achieve three resilience properties pertaining to the ability of a system to absorb the impacts of perturbations, adapt to undesirable situations, and quickly return to its normal operations. In [84], a three-stage framework, reflecting the infrastructure's resistant, absorptive, and restorative capacities, is introduced to analyze the resilience. The DHS work in [83] developed the notion of a resilience measurement index (RMI) which is an indicator to determine the degree to which the elements pertaining to resilience have been implemented by a CI. These elements include the preparedness of the CI to possible failures and the extent to which recovery mechanisms and mitigation measures are installed. The work in [135] introduced a quantitative assessment for infrastructure's resilience using optimal control design in which recovery processes and costs are integrated to derive the resilience. However, one key limitation of these studies, [28, 83, 84, 134, 135], is that they can be used to compare different CIs, yet, they do not capture the effect of specific events on the infrastructure. Therefore, their use is mostly limited to evaluating the resilience of CI but not to improving it.

Other studies in the literature have focused on improving CI resilience by allocating CI-specific physical resources [85, 136–138]. In [136], the resilience of a cyber-physical system is improved by allocating a number of inter-network edges to the nodes of the interdependent network connecting the system's cyber and physical layers. The effect of cascading failure among nodes is studied to help in the process of resource allocation. The authors in [85] proposed a new approach to repair system components using a graph-theoretic approach. In [137] and [138], CI resilience is studied from a general perspective without defining a quantitative metric for resilience. The authors in [137] consider the problem of allocating resources to highway bridges to improve the resilience of a transportation system. In [138], a framework is proposed to allocate resources to CIs based on their vulnerability level. Contract theory is used to formulate the problem to optimize the economic benefit from the allocated resources which are offered to CIs through contracts managed by the system operator. Note that, in [84], beyond defining resilience properties, a framework is proposed to improve CI resilience. The framework depends on allocating resources to improve the resilience by hardening CI's components, duplicating components, or ensuring rapid recovery of failed components. The framework is applied to improve the resilience of a power grid whose components are the generators and the resources are allocated to the generators.

One limitation of these previous studies [84] and [85, 136–138], is that individual CIs are abstracted within the system, e.g. as nodes within a generic graph. This provides no information on improving individual CIs resilience as the solutions introduced in these studies [84] and [85, 136–138] consider the resilience of an entire system of multiple CIs while being agnostic to each individual CI's resilience properties. Indeed, individual CIs and their specific failures are largely abstracted and not considered in enough details. Hence, in such prior art, when resources are allocated within the system, no information is provided on how to effectively allocate them at the level of each CI.

In light of the preceding discussions, we propose a general framework to study and improve the resilience of CIs. The framework addresses *the resilience at the level of both individual CIs and*

their collective effect on an entire system of multiple CIs. We introduce an analytical resilience index to quantify the resilience of individual infrastructures and to give insights about improving this resilience. Resilience is evaluated as a function of the CI's probability of failure derived from the cascading failure of its physical components. Resources are then allocated to the individual CIs according to their contribution to the entire system. Examples of resources here include redundant components or monitoring devices such as sensors or cameras. Finally, each infrastructure can use the allocated resources to improve its resilience based on the introduced allocation algorithm. The key contributions stemming from this framework are outlined next.

5.1.2 Contributions

The main contribution of this chapter is a comprehensive analytical framework for analyzing and optimizing the resilience of CIs. The proposed framework can be applied to different systems and CIs to evaluate their resilience and optimize it. In particular we have the following key contributions:

- We develop a novel three-state Markov chain model for CI performance which incorporates a proposed "warning" state to represent the case of partial CI failure. This is in contrast to the binary models that are used in the literature such as in [23]. This three-state model allows derivation of a novel quantifiable resilience index that relates the CI probability of failure to the probability of recovering from the proposed warning state. The analytical derivation of the resilience index and the associated proofs are also provided.
- As the resilience index is directly related to the probability of recovering from a warning state, we introduce a Bayesian network design to capture the effect of each CI component's failure on probability of failure of the entire CI. The probabilistic inference from this Bayesian network is used to calculate the transition probabilities from the warning state and, hence, the actual resilience index values of the given CI. It also allows calculation of the effect of fixing each component on the infrastructure's resilience index.
- We also develop a novel algorithm, using the Bayesian network, to prioritize each CI's components based on their effect on the resilience index. This algorithm can be used by individual CIs to determine the order in which they should secure their key components through external resources.
- We then investigate the process of improving the resilience of a system of multiple CIs. For this purpose, a case study pertaining to hydropower dams is introduced. Within this case study, a hydropower dam's resilience is evaluated based on its probability of successfully generating electricity. We propose a new approach for improving the dam's resilience by securing its main components using external resources.
- The problem of allocating these resources to multiple dams, based on their economic contribution to the entire system (the power grid), is modeled using *contract theory* [139] and the

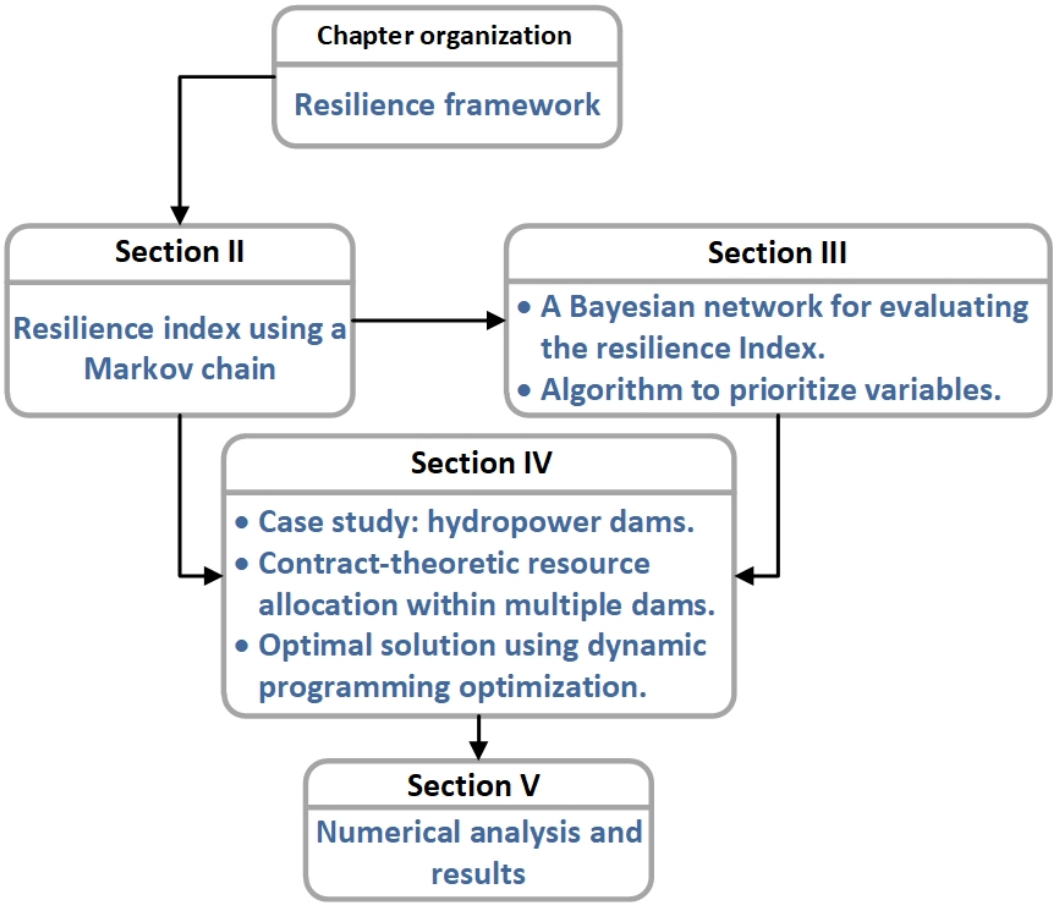


Figure 5.1: Organization of the chapter.

optimal solution to this problem is analytically derived using dynamic programming.

- Through simulations, we show that both the system operator and individual CIs can benefit from the process of resource allocation. The system operator can maximize its reward from the allocated resources using contract theory, while CIs significantly improve their resilience indices.

The rest of this chapter is organized as summarized in Figure 5.1. The Markov chain model and the analytical analysis for deriving the resilience index is presented in Section 5.2. The Bayesian network analysis and components' prioritization algorithm is discussed in Section 5.3. The case study of hydropower dams and the optimal solution to the problem of CI resource allocation is derived in Section 5.4. Numerical results are presented and analyzed in Section 5.5 and Section 5.6 concludes the chapter.

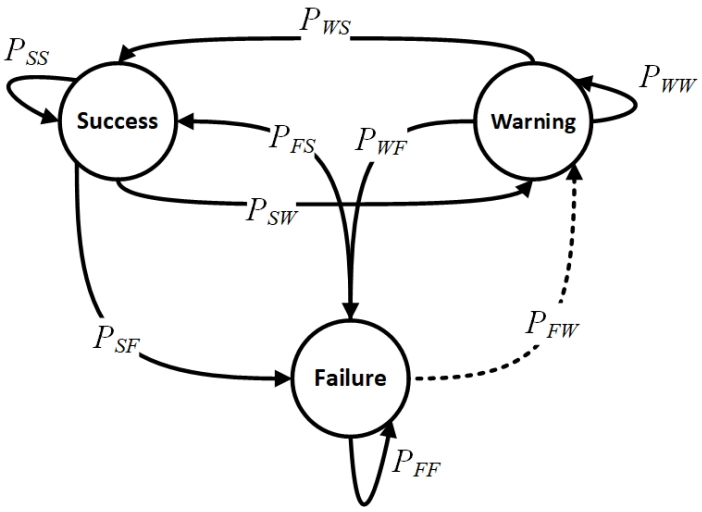


Figure 5.2: Markov chain modeling the states of a CI.

5.2 Evaluating the Resilience of Critical Infrastructure using Markov Chains

5.2.1 General Case

Consider a critical infrastructure whose performance at a given time n is a function of the state of the system as captured by the random variable Y_n. Y_n can take values from a set $\{S, W, F\}$ whose values represent three CI states: success (S), warning (W), and failure (F). The success state, S, represents normal service, i.e., the infrastructure is properly delivering its designated service. The CI will be in a warning state, W, with the occurrence of a partial failure to its components that may lead to a complete failure. The failure state, F, represents the failure of the CI to deliver its designated service. This failure can occur either suddenly due to, e.g., natural disasters, or as a result of a partial failure from a previous warning state. We introduce a Markov chain to model these states as shown in Figure 5.2.

The transition probabilities between the different states can be induced from the Markov chain. Let $P_{AB} = \text{Prob}[Y_{n+1} = B|Y_n = A]$ where $A, B \in \{S, W, F\}$. Then, the full transition probability matrix $\boldsymbol{P}$ will be given by:

$$\boldsymbol{P} = \begin{bmatrix} P_{SS} & P_{SW} & P_{SF} \\ P_{WS} & P_{WW} & P_{WF} \\ P_{FS} & P_{FW} & P_{FF} \end{bmatrix}. \tag{5.1}$$

The values within each row of $\boldsymbol{P}$ will sum to 1 as they represent the probability distribution for all possible next states whenever the system is at a specific state.

We assume that, whenever the infrastructure is at a warning state, it is either fixed and restored to a success state or it continues to fail and eventually goes to a failure state. This transition is based on the actions taken at a given time step, however, there is still a small probability that no action is taken at this time step as captured by the probability of remaining in the warning state P_{WW}. We assume that P_{WW} is fixed to a value ϵ which should be small. Based on this assumption, the transition probability can be simplified, as follows:

$$\boldsymbol{P} = \begin{bmatrix} P_{SS} & P_{SW} & P_{SF} \\ P_{WS} & \epsilon & 1-\epsilon-P_{WS} \\ P_{FS} & P_{FW} & P_{FF} \end{bmatrix}. \tag{5.2}$$

Note that, the probability matrix $\boldsymbol{P}$ specifies transition probabilities for a single time step. Transition probabilities for n time steps can be calculated as $\boldsymbol{P}^n$. The probabilities of being at a given state i.e., $P(S), P(W)$, and $P(F)$, can be calculated using $\boldsymbol{P}^n$ and the vector of the initial probability distribution of being at each state $\boldsymbol{U}_0$, as follows:

$$\boldsymbol{U}_n = \boldsymbol{U}_0 \cdot \boldsymbol{P}^n, \tag{5.3}$$

where $\boldsymbol{U}_n = [u_n^S, u_n^W, u_n^F]$ is the probability distribution vector of being at each state after n time steps.

We define the resilience γ as the effect of the probability P_{WS} of an infrastructure on the reciprocal of its probability of being in the failure state in the long run, as follows:

$$\gamma = \lim_{n\to\infty} \frac{1}{u_n^F}\bigg|_{P_{WS}}, \tag{5.4}$$

where u_n^F is the probability corresponding to the failure state in the vector $\boldsymbol{U}_n$. This probability will always exist if the Markov chain is irreducible as shown in [140], i.e., there is no absorbing state.

In practice, the chain in Figure 5.2 cannot have an absorbing state so it is irreducible. The resilience here, is measured as the effect of P_{WS} on the long run probability of failure. In other words, if the CI has a higher P_{WS} then it will have a high probability of recovering from partial failures to a success state. This, in turn, implies that the CI will have a smaller probability of being in a failure state after partial failures which conforms to the definition of resilience in [27]. The long-run probability of failure was chosen to reflect the CI's operation on the long-run and because it is a suitable minimization objective for the CIs. Note that, relating the resilience to the probability of failure was introduced in [23] however, in [23], the CI was allowed to only operate in either a satisfactory or a failure state. Our model, on the other hand, introduces the definition of the warning state and derives a quantitative resilience measure based on the Markov chain transition probabilities. This, in turn, requires new analysis that is different from [23] (and references therein).

Here, we note that some existing works (in the context of power systems) [141] define multiple "derated" states to describe the output capacity as fractions of the nominal capacity. However, our

definition of the "warning" state describes the CI system status based on its internal components. This makes it more general and, hence, can apply to multiple CIs compared to the "derated" states which are limited to power systems.

To calculate the value of γ, the value of u_n^F needs to be evaluated at high values of n which in turn will depend on $\boldsymbol{P}^n$. The powers of $\boldsymbol{P}$ for high values of n can be calculated in advance if $\boldsymbol{P}$ is a regular transition matrix [140]. The powers $\boldsymbol{P}^n$ are shown to converge to a matrix $\boldsymbol{V}$ in which all rows are the same and each row is a strictly positive probability distribution vector [140] if $\boldsymbol{P}$ is a regular transition matrix. Converging to a constant matrix means any further multiplications of $\boldsymbol{V}$ with $\boldsymbol{P}$ will not change $\boldsymbol{V}$, i.e.,

$$\boldsymbol{V} \cdot \boldsymbol{P} = \boldsymbol{V}. \tag{5.5}$$

As all rows in $\boldsymbol{V}$ are the same, the probability vector $\boldsymbol{U}_n$, in (5.3), will no longer depend on the initial probability distribution $\boldsymbol{U}_0$. Multiplying the values of $\boldsymbol{U}_0$, which sum up to 1, with the constant columns of $\boldsymbol{V}$ will yield the same constant values of $\boldsymbol{V}$. Hence, (5.3) can then be written as:

$$\boldsymbol{U}_n = \boldsymbol{U}_0 \cdot \boldsymbol{V} = \boldsymbol{v}, \tag{5.6}$$

where the vector $\boldsymbol{v}$ is the constant row of $\boldsymbol{V}$. It consists of three probabilities representing the transition probabilities to each one of the states. According to (5.5), this vector will satisfy the following property:

$$\boldsymbol{v} \cdot \boldsymbol{P} = \boldsymbol{v}. \tag{5.7}$$

As discussed earlier, the matrix $\boldsymbol{V}$ only exists if the matrix $\boldsymbol{P}$ is a regular Markov matrix. However, proving this analytically can be intractable for the general case. Instead, we provide an analytical proof in Section 5.2.2 for a special, yet practical, case.

To shed some light on the number of time steps (iterations) needed for the matrix $\boldsymbol{P}$ to converge to $\boldsymbol{V}$ in the general case, three different CIs are examined as shown in Figure 5.3 and Figure 5.4. Figure 5.3 shows the values of P_S when it reaches a constant value after some time steps while Figure 5.4 shows the values of P_W. In this example, the first CI has a high probability of being at a success state, $P_{SS} = 0.7$ and high probabilities of returning to the success state, $P_{WS} = P_{FS} = 0.7$. The values of ϵ and P_{FW} are set to 0.1. Figure 5.3 and Figure 5.4 show that convergence occurs at $n = 3$. The second CI has a high probability of being at a success state $P_{SS} = 0.8$ but lower transition probabilities $P_{WS} = 0.2$ and $P_{FS} = 0.3$. The values of ϵ and P_{FW} are set to 0.1 and 0.3, respectively. The convergence in this case occurs at $n = 7$ for P_S and $n = 5$ for P_W. Finally, the third CI has a low probability $P_{SS} = 0.4$ and high transition probabilities $P_{WS} = 0.9$ and $P_{FS} = 0.7$. The values of ϵ and P_{FW} are set to 0.1 and 0.2, respectively. The convergence occurs approximately at $n = 6$ for P_S and $n = 3$ for P_W. Clearly, only few time steps are needed for each CI's transition matrix to converge which corroborates the practicality of our proposed approach. Note that the actual transition probabilities of the matrix $\boldsymbol{P}$ should be estimated for each CI. A thorough assessment can be carried out to determine the infrastructure's main parameters such as structural conditions, possible defects, and aging of the components. These parameters can

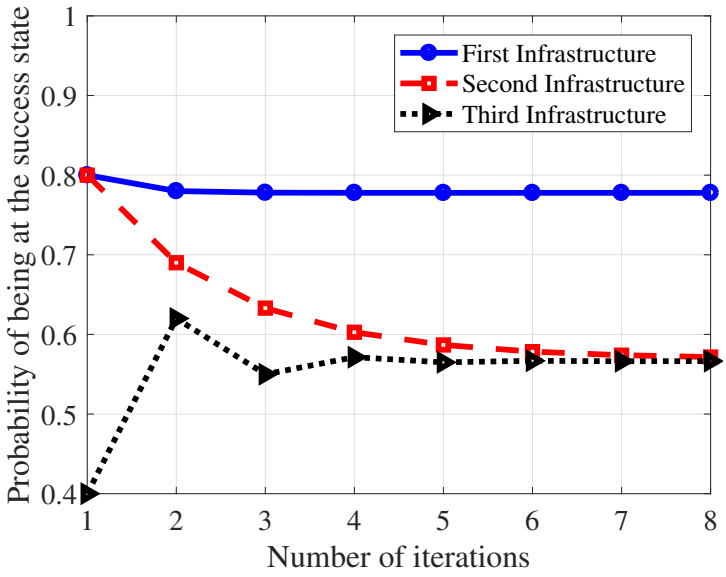


Figure 5.3: P_S convergence with number of iterations.

then be used to estimate the transition probabilities, e.g., P_{SS}, P_{SW}, and P_{FS}, using a number of mathematical models as done in [142].

In the next section, we discuss a special case where $P_{FW} = 0$. Under this case, we can analytically prove that the matrix $\boldsymbol{P}$ is a regular Markov matrix thus deriving a closed-form expression for the resilience index in this case. Note that, the special case is introduced for the analytical tractability however, the analysis in Sections 5.3 and 5.4 will still hold for both the general and the special case.

5.2.2 Special Case: $P_{FW} = 0$

In this section we discuss a special case where $P_{FW} = 0$. This represents the case when fast remedial actions are taken in the failure state so that the CI will be able to recover to a success state, without being at any warning state. This case also includes scenarios in which there is still a small probability that the CI operates in a warning state after being in a failure state, however, this CI must not remain for too long in the warning state and, hence, the probability P_{FW} can be approximated to zero.

Based on this assumption, the transition probability can be simplified, as follows:

$$\boldsymbol{P} = \begin{bmatrix} P_{SS} & P_{SW} & P_{SF} \\ P_{WS} & \epsilon & 1-\epsilon-P_{WS} \\ P_{FS} & 0 & 1-P_{FS} \end{bmatrix}. \tag{5.8}$$

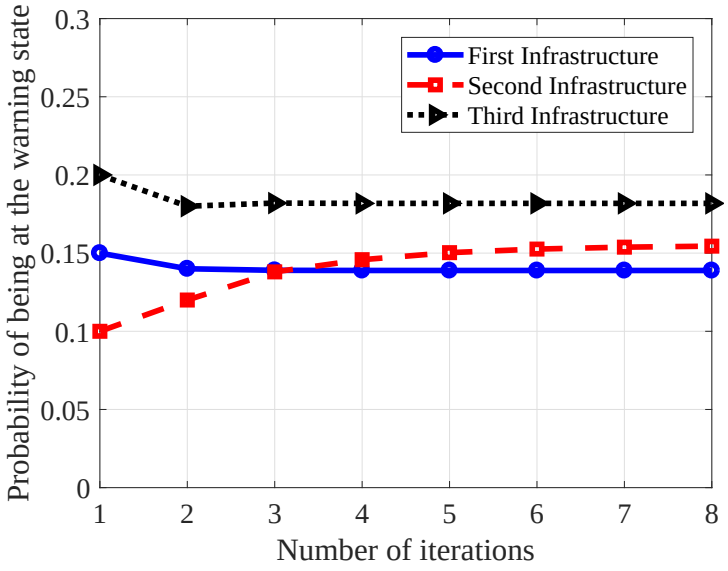


Figure 5.4: P_W convergence with number of iterations.

In the following theorem, we prove the necessary conditions that $\boldsymbol{P}$ must satisfy in order to be a regular Markov matrix.

Theorem 3. *When $P_{FW} = 0$, it can be shown that the probability transition matrix $\boldsymbol{P}$ is a regular Markov matrix.*

Proof. Let $\boldsymbol{v} = [v^S, v^W, v^F]$, then, the values of v^S, v^W, v^F can be computed using (5.7), as follows:

$$\begin{aligned} v^S &= v^S \cdot P_{SS} + v^W \cdot P_{WS} + v^F \cdot P_{FS}, \\ v^W &= v^W \cdot P_{SW} + v^W \cdot \epsilon, \\ v^F &= v^S \cdot P_{SF} + v^W \cdot P_{WF} + v^F \cdot P_{FF}, \\ v^S &+ v^W + v^F = 1. \end{aligned} \tag{5.9}$$

The solution of this set of equations gives the values:

$$\begin{aligned} v^S &= \frac{(1-\epsilon)\cdot P_{FS}}{P_{FS}\cdot(1-\epsilon+P_{SW}) + (1-\epsilon)\cdot(1-P_{SS}) - P_{WS}\cdot P_{SW}}. \\ v^W &= \frac{P_{FS}\cdot P_{SW}}{P_{FS}\cdot(1-\epsilon+P_{SW}) + (1-\epsilon)\cdot(1-P_{SS}) - P_{WS}\cdot P_{SW}}. \\ v^F &= 1 - \frac{P_{FS}\cdot(1-\epsilon+P_{SW})}{P_{FS}\cdot(1-\epsilon+P_{SW}) + (1-\epsilon)\cdot(1-P_{SS}) - P_{WS}\cdot P_{SW}}. \end{aligned} \tag{5.10}$$

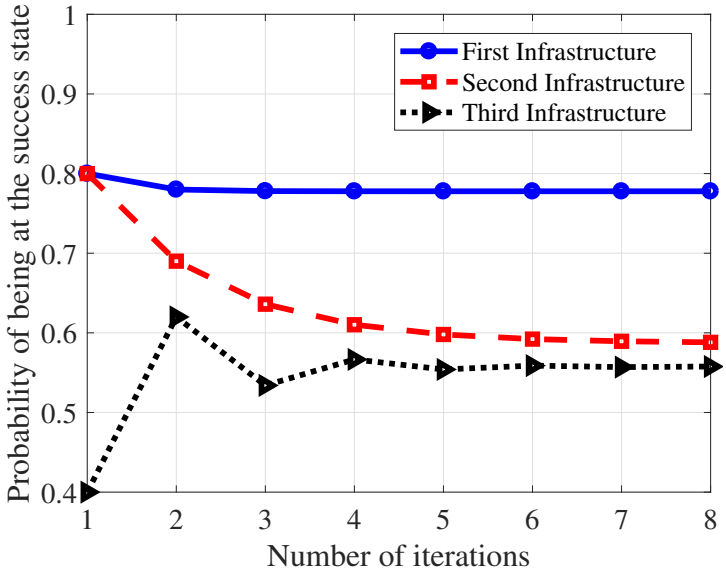


Figure 5.5: P_S convergence with number of iterations.

Substituting P_{WS} in the denominator by $1-\epsilon-P_{WF}$, the denominator can be written as $P_{FS}\cdot(1-\epsilon+P_{SW})+P_{SW}\cdot P_{WF}+(1-\epsilon)\cdot P_{SF}$, with all the terms being positive. The numerators can then determine the sign of the values of v^S, v^W, v^F. It is obvious that v^S, v^W, v^F will have positive values if P_{SW}, P_{FS} are nonzero. However, if P_{SW} equals zero, there will be no transition to the warning state when the CI starts at the success state. This implies that the warning state will be isolated which contradicts the fact that the chain is irreducible. The assumption P_{FS} is zero also cannot hold from a practical point of view since in this case the infrastructure cannot be recovered to the success state hence is not resilient.

This shows that the values v^S, v^W, v^F will always be positive hence they represent valid transition probabilities which proves that $\boldsymbol{P}$ is a regular Markov matrix. □

Theorem 3 shows that, under our proposed model, the transition probabilities in (5.6) will have positive values and, hence, the resilience measure in (5.4) will always converge to a real number. The resilience in (5.4) can then be written as:

$$\gamma = \lim_{n\to\infty} \frac{1}{u_n^F} = \frac{1}{v^F}\bigg|_{P_{WS}}. \tag{5.11}$$

We then compare the number of time steps needed for the matrix $\boldsymbol{P}$ to converge to $\boldsymbol{V}$ when $P_{FW}=0$. We use the same parameters as in Figure 5.3 and Figure 5.4, however, here, the value of P_{FW} is set to zero and its value is added to that of P_{FF}. We can see that Figure 5.5 for P_S is almost the same as Figure 5.3. Figure 5.6, on the other hand, shows different convergence steps values for

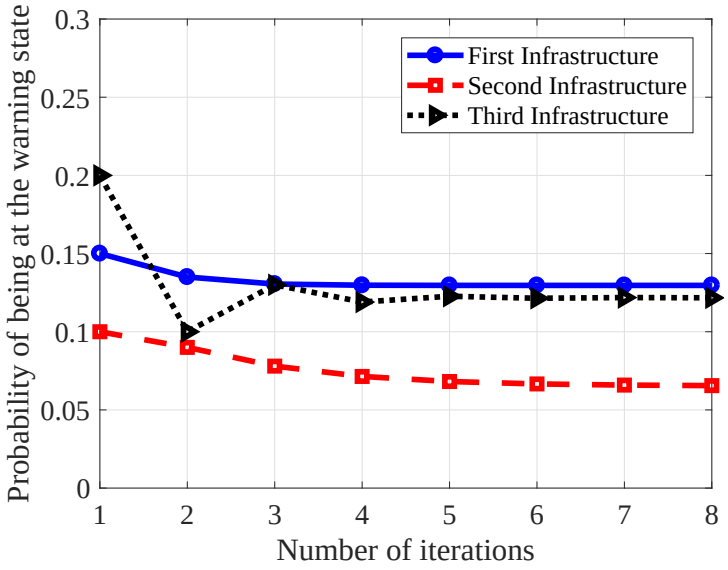


Figure 5.6: P_W convergence with number of iterations.

P_W than Figure 5.4. However, in both cases only few time steps are needed for each CI's transition matrix $\boldsymbol{P}$ to converge to $\boldsymbol{V}$.

Since our notion of resilience primarily depends on the effect of the probability P_{WS} on the CI's probability of failure, we are interested in improving P_{WS} by increasing the transition probability from warning (W) state to the success state (S) thus reducing the probability of failure. To this end, we evaluate the rate of change of the resilience with respect to the probability P_{WS}, as follows:

$$\frac{\partial v^F}{\partial P_{WS}} = \frac{-P_{FS} \cdot P_{SW} \cdot (1 - \epsilon + P_{SW})}{\big(P_{FS} \cdot (1 - \epsilon + P_{SW}) + (1 - \epsilon) \cdot (1 - P_{SS}) - P_{WS} \cdot P_{SW}\big)^2}. \tag{5.12}$$

This rate of change is strictly negative which implies that v^F will always decrease with the increase of P_{WS}. From (5.10) and (5.11), it can be clearly seen that the resilience will have a positive rate of change with respect to the probability P_{WS}.

Finally, we define the *resilience index* θ of a CI as:

$$\theta = \frac{\gamma}{\gamma_{\max}}\bigg|_{P_{WS}} = \frac{v^F_{\min}}{v^F}\bigg|_{P_{WS}}, \tag{5.13}$$

where $v^F_{\min}$ is the minimum value of v^F that can be achieved at the maximum value of $P_{WS} = 1 - \epsilon$

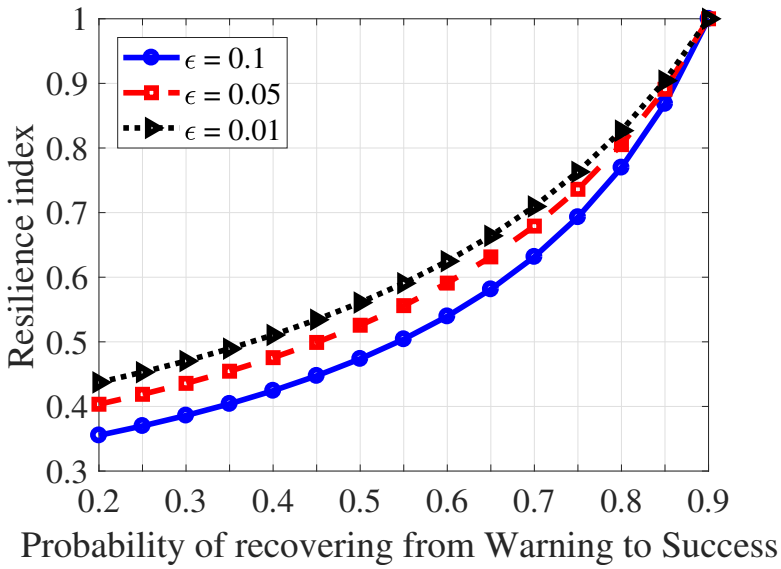


Figure 5.7: Resilience index change with the probability P_{WS}.

when substituted into (5.10) and (5.11). This value achieves a maximum resilience $\gamma_{\max}$. It is straightforward to show that θ is positive with $\theta \leq 1$. The resilience index in this way helps to evaluate how far each CI is from its maximum achievable resilience. It can also help to compare different CIs as their resilience is measured on the same scale. Figure 5.7 shows the values of the resilience index with the increasing values of P_{WS} for different ϵ values when $P_{SS} = 0.8, P_{FS} = 0.5$.

Next, we study how to compute the probability P_{WS} for an infrastructure and the effect of improving this probability on the resilience of a given CI. A Bayesian network is defined for this purpose as explained next.

5.3 Bayesian Network Model for CI Probability of Failure

To compute P_{WS}, we need to evaluate the probability of failure of a CI, given the probability of failure of each of its individual components. Since the failure of one or more components can cause other components to fail, we need to consider the relationship between the components when computing P_{WS}. To this end, a Bayesian network [143] is a suitable framework.

5.3.1 Bayesian Networks: Preliminaries

A Bayesian network is a network that describes the causality and relationship between independent random variables under incomplete information [143]. A Bayesian network is normally

represented by a directed acyclic graph (DAG) in which each node represents one random variable. Let $G(\mathscr{X}, \mathscr{E})$ be a Bayesian network, then $\mathscr{X} = \{X_1, \ldots, X_n\}$ is the set of nodes which represent different random variables and $\mathscr{E}$ is a set of directed edges. A directed edge from a node X_j to X_i means that node X_i depends on the node X_j and in this case X_j is called the parent of node X_i. A node can have multiple parents and the set of parent for a node X_i is given by $\pi(X_i)$. Every variable X_i can take a value from a finite set of values, e.g., if the variables are binary they can either be true or false. Finally each node is associated with a conditional probability table (CPT) while roots, i.e, nodes without parents, are assigned direct probabilities. A CPT for a node X_i gives the conditional probabilities between X_i and every node in $\pi(X_i)$.

Consider a Bayesian network with binary values, each variable in the network can be either true ($\mathcal{T}$) or false ($\mathcal{F}$) with a given probability. For a variable C_1, its $\mathcal{T}$ and $\mathcal{F}$ probabilities, i.e, $P(C_1) = \mathcal{T}, P(C_1) = \mathcal{F}$ are written as $P(c_1), P(\bar{c}_1)$ where the lowercase letter indicates a value of the variable. The probabilities within each variable sum to 1, i.e., $P(c_1) + P(\bar{c}_1) = 1$. If a variable, e.g. D_1, has two parents C_1 and C_2, then the CPT of D_1 will have eight entries representing the possible combinations of C_1 and C_2 with the $\mathcal{T}$ and $\mathcal{F}$ values of D_1. However, as the $\mathcal{T}$ and $\mathcal{F}$ values for any variable sum to 1, only half of the CPT entries must be stored, i.e., the $\mathcal{T}$ values of D_1.

Once the probabilities and the CPTs are assigned, probabilistic inference can be performed to calculate the probability of any variable given some evidence in the network. Calculating probabilistic inference in general Bayesian networks in known to be NP-Hard [144], however, Pearl [145] introduced a polynomial time algorithm to perform probabilistic inference in *singly connected Bayesian networks*. A singly connected Bayesian network, also known as a polytree, is a Bayesian network where it has no loops, i.e., there is only one path between any two nodes in its underlying undirected graph.

5.3.2 Evaluating CI Probability of Failure

For our resilience problem, we introduce a Bayesian network to model the possible failure events of a given CI that can prevent it from delivering its designated service. We model the various possible failure events of the components of a CI which can lead to total CI failure with a given probability. The total failure probability calculated from this Bayesian network will effectively represent the transition probability P_{WF} as this is the probability with which a partial failure causes a total failure. Here, we note that, the warning state can be reached by disasters, failures, attacks, or even normal wear-and-tear of the components. For the scenarios in which disasters cause gradual failure, the infrastructure will still be operational, however, some components are damaged and can potentially lead to the whole infrastructure failure, then the infrastructure will be in a warning state. In this case, we can use the Bayesian network to evaluate the probability P_{WF} of the CI by considering the failure probability of the affected components. On the other hand, if the disaster causes a sudden failure to the CI, the CI will directly go to a failure state. Consequently, such sudden disaster-related failure events require improving the transition probability P_{SF} and can be

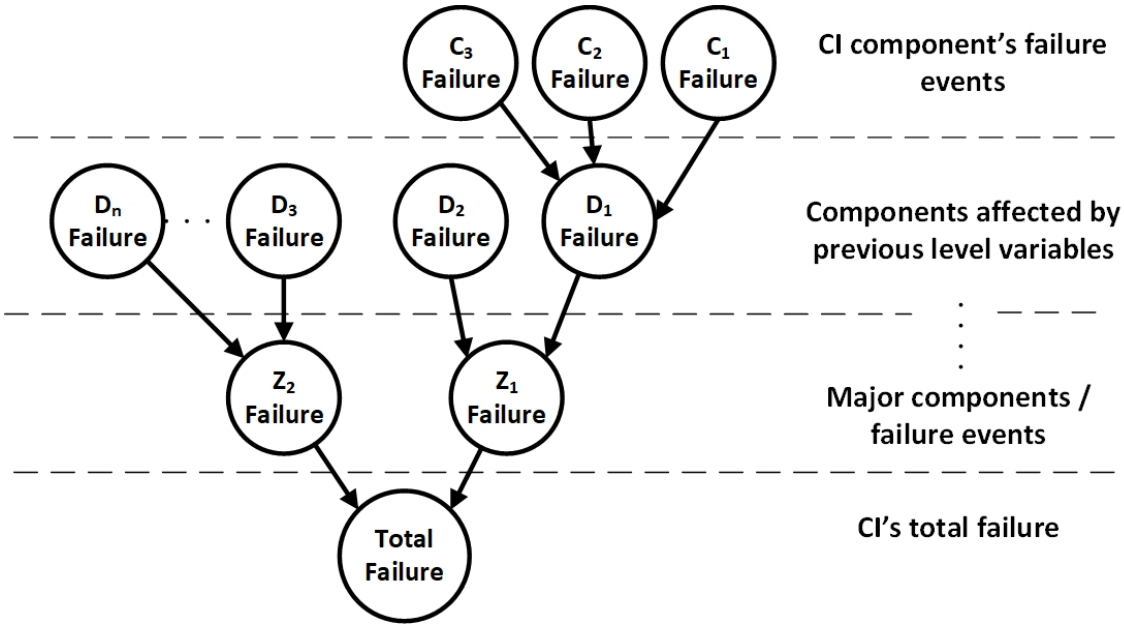


Figure 5.8: A Bayesian network representing the hierarchical failure events within a CI's components.

subject to future extensions of our model.

The Bayesian network is constructed such that failure events are modeled as variables (nodes) in a hierarchical way. Nodes are grouped into levels where failures in one level can cause failures in the next level. Figure 5.8 shows the structure of the proposed Bayesian network in which the number of nodes and levels vary according to each infrastructure. Roots in the network, $(C_1, C_2, C_3, D_2, \ldots, D_n)$, represent possible failure events to respective CI components. Failure, here, can happen due to external effects or normal wear-and-tear of the components. The subsequent levels represent the cascading failure to other major components, e.g. Z_1 and Z_2. These major failures can, in turn, cause a failure in their next level, with given probabilities, and so on until the whole infrastructure fails. The CI failure is represented by the single leaf in the Bayesian network.

As the variables represent failure events, they can be either true ($\mathcal{T}$) or false ($\mathcal{F}$) with a given probability. A $\mathcal{T}$ value implies that the failure event has occurred, e.g., after a disaster or normal wear-and-tear of the components and $\mathcal{F}$ means it has not occurred. Root variables are assigned $\mathcal{T}$ and $\mathcal{F}$ probabilities, while the variables at subsequent levels are associated with conditional probabilities for the possible combinations of their parents' values. Probabilistic inference can then be performed to compute the total probability of failure, of a CI according to the failure probabilities of its components. Note, the proposed network in Figure 5.8 is a singly connected Bayesian network and, hence, probabilistic inference can be performed in polynomial time.

Note that, this Bayesian network design is similar to fault trees [146] which are used to study

systems reliability. The main advantage of using a Bayesian network over a fault tree is that fault trees use logic gates to define the relation between some components and their next level component. In such a case, the failure probability of next-level components will only depend on the occurrence of some or all of the previous level failures. In contrast, Bayesian networks allow the assignment of conditional probabilities for each component and its next-level component. This allows the use of different weights to take into account the effect of some components on their next-level component.

Some work in the literature considered using Bayesian networks to study CIs. In particular, the work in [147] and [148] studied Bayesian networks in the context of safety analysis and fault diagnosis, respectively. The main focus in these and similar contributions, is designing the Bayesian network and estimating the different probabilities. In our work, we provide the design as well as the necessary Bayesian inference for the failure events and their effect on the total probability of failure which we are considering next.

To compute the probability of failure P_{WF} for any given CI, we start by calculating the *prior marginal probability* of failure. This probability is calculated from the initial assigned probabilities. Assume, without loss of generality, that $X_n \in \mathscr{X}$ is the variable representing the failure, then the prior marginal probability of x_n, the $\mathcal{T}$ value of X_n, can be calculated in a manner analogous to [149]:

$$P(x_n) = \sum_{\mathscr{X} \setminus X_n} P(x_1, \ldots, x_n), \tag{5.14}$$

where $P(x_1, \ldots, x_n)$ is the joint probability for all the instantiations of the independent random variables $X_1, \ldots, X_n$. The summation is calculated over all variables except X_n, thus these variables are marginalized from the joint probability. The joint probability in (5.14) is given by:

$$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i | \pi(x_i)), \tag{5.15}$$

where $P(x_i|\pi(x_i)) = P(x_i)$ when the set $\pi(X_i)$ is empty.

This prior marginal probability $P(x_n)$ represents the initial P_{WF} in our CI Markov chain model. It can be used to calculate the initial resilience index of a given CI. Then, the effect of each node on improving the resilience index can be calculated and, hence, the components of a CI can be prioritized based on their effect on P_{WF}. However, the effect of each component should not be considered separately, as securing a component will reduce P_{WF} which will also reduce the effect of other components on P_{WF}. The proposed procedure for sorting the components is given next.

We calculate the *posterior marginal probability* of failure given the evidence of each root variable separately. Let the number of root variables be $m < n$. Then, the marginal probability will be given by [149]:

$$P(x_n|\bar{x_i}) = \sum_{\mathscr{X} \setminus X_n} P(x_1, \ldots, x_n|\bar{x_i}), i \in \{1, \ldots, m\}, \tag{5.16}$$

where $\bar{x_i}$ is the evidence value for the variable X_i when $P(X_i) = \mathcal{F}$. We consider the false (failure) probability to capture the positive effect of a variable on the total probability of failure.

Algorithm 1: Variables Sorting According to Their Effect

Input: Bayeian network variables $\mathscr{X}$ where $X_n \in \mathscr{X}$ is the only leaf representing the failure

Output: A sorted set of the root variables $\mathscr{S}$

begin

- Calculate the prior marginal probability of failure $P(x_n)$
- **for** *each root node* **do**
  - Calculate the posterior marginal probability $P(x_n|\bar{x}_i)$
  - Calculate the variable's effect $P(x_n) - P(x_n|\bar{x}_i)$
- Sort variables descendant according to their effect
- Determine the variable X_i with the greatest effect
- Store the variable X_i in the set $\mathscr{S}$
- Define the partial set $\mathscr{X}_r$ as the set of roots excluding X_i
- **while** *The set $\mathscr{X}_r$ is not empty* **do**
  - **for** *each node $X_j \in \mathscr{X}_r$* **do**
    - Calculate the posterior marginal probability $P(x_n|\bar{x}_i, \bar{x}_j)$
    - Calculate the variable's effect $P(x_n|\bar{x}_i) - P(x_n|\bar{x}_i, \bar{x}_j)$
  - Determine the variable X_j with the greatest effect
  - Store the variable X_j in the set $\mathscr{S}$
  - Update the set $\mathscr{X}_r = \mathscr{X}_r \setminus X_j$
- **return** $\mathscr{S}$

This is calculated for all root variables and the values are sorted in a descending order. The variable that causes the greatest reduction in P_{WF} then represents the first component of the CI that must be overhauled. This variable is also used as a new evidence variable in the Bayesian network to determine the second most affecting variable (component) from the remaining roots.

Assume without loss of generality that X_1 is the root with the most effect on P_{WF}, then the next posterior marginal probability is calculated considering only the $\bar{x_1}$ instantiation of X_1. The probability is calculated for the remaining root variables individually as given by:

$$P(x_n|\bar{x_1}, \bar{x_i}) = \sum_{\mathscr{X}\setminus X_n, X_1} P(x_1, \ldots, x_n|\bar{x_1}, \bar{x_i}), i \in \{2, \ldots, m\}. \tag{5.17}$$

This procedure is applied to all the roots adding one root to the evidence variables each time. The procedure will end by sorting all the components of a CI in a descending order according to their effect on the probability of failure. The steps of this procedure are summarized in Algorithm 1.

Note that, according to (5.15), the joint probability considers the parents of each node. Thus, (5.17) can be derived from (5.16) by considering changes in the branch between the leaf node and

the new variable only. All the other summations in (5.16) will not change as the evidence variable does not belong to this branch. This allows a reduction in the complexity of calculating the updated probabilities after fixing the components.

Algorithm 1 can then be used by any CI to determine the order according to which it must fix its components. As CIs typically allocate resources to improve their resilience [84, 85, 136–138], Algorithm 1 can help CIs to determine the components to which resources will be allocated within each CI.

Here, we note that, in practice, CIs operate within larger systems (e.g., an entire city) that are composed of multiple, interdependent CIs that collectively provide a common service. As such, the function loss of one CI will impact other interdependent CIs and, therefore, when analyzing the resilience of a large-scale system, one must consider all the interdependent CIs. This, in turn, brings forward a new problem of allocating resources, such as monitoring devices among a system of multiple CIs which is addressed next. Within the context of resource allocation, Algorithm 1 is applied by each CI to make the best use of its allocated resources.

5.4 Resource Allocation for Optimized Resilience

As evident from the previous discussion, our next step is to study the problem of allocating resources in a system of multiple CIs, while taking into account the individual Bayesian network model of each CI. Resources can range from cyber resources to personnel or physical equipment. We classify resources into two categories: preventive and rapid intervention resources. Preventive resources are resources that help CI's components become less vulnerable to failures. This might include replacing some components with more reliable ones or installing redundant components. Rapid intervention resources, on the other hand, requires monitoring and alarming systems to be deployed and requires the existence of on-site facilities that can be used to fix or replace corrupted components in a timely-manner. The choice of either category of resources depends on the nature of the infrastructure and the cost of using each. For instance, in a power plant, preventive resources can represent installing redundant switches or replacing old stators, while rapid intervention resources can represent excessive monitoring of the generators to repair any defects once they occur to help keep the generator working.

As resources are infrastructure-specific, we introduce an application-specific case study to highlight the importance of our framework. Though the framework can be applied to any CI, studying the problem of resource allocation within the context of a specific CI, as a case study, helps better illustrate our framework, as shown next.

5.4.1 Hydropower dams: A case study

We apply the proposed framework to hydropower dams and their impact on power systems as a practical CI in order to measure the resilience improvement that can be achieved. Dams are classified as one of the critical infrastructure sectors according to the US DHS [3]. Hydropower dams provide a good platform to apply our proposed framework as they have many connected components that could be affected by numerous failure events. Recall that, according to our proposed framework, failure will be defined as the inability of the dam to produce electricity.

A Bayesian network is designed for each dam where the parents to the node representing failure are the dam's main components such as penstocks, generators, turbines and transformers. In turn, these variables are modeled as children nodes of the variables representing smaller components such as stators, rotors, intake gates, and blades. Components are connected in a hierarchical manner until the roots that represent small components failure. Figure 5.9 shows a scheme for a hydropower dam highlighting its main physical components along with part of the Bayesian network defined for this dam. We use previous failure statistics and reliability analyses [150] to assign probabilities of failure to the roots of the Bayesian network. Conditional probabilities between components are assigned based on the components' relations similar to method used in fault trees. Note that the same method of assigning probabilities can be applied to any other CI.

From a resources perspective, preventive resources are seen as a long-term solution to improve the resilience of dams in service. Preventive resources require some components to be replaced, which might not be applicable when the dam is in service. Therefore, *we focus on rapid intervention resources* which include monitoring devices, such as sensors and cameras, and maintenance equipment. We propose to use both fixed sensors and drones in the monitoring process. Drones can be used in general to inspect areas of interest in CIs [151] and help to inspect hard-to-reach points where conventional sensors/monitoring methods cannot be used. Recently, the use of drones to inspect even the inner parts of the dam was shown to be applicable in [152]. In this work, the authors modified a drone and used it to inspect the inside of the pentstocks of a number of dams. Mechanical robots on the other hand can be used to inspect a dam's key sections that cannot be reached by drones such as underwater components [153].

The majority of dams in the United States, are privately owned [154] and their owners are responsible for their safety. However, there is still a federal role for ensuring dams' safety as dams can severely affect persons and properties in case of failure. The same applies to electricity supply, the failure of a dam to generate electricity will affect huge parts of the electric grid that it supplies. These facts reveal the importance of having a system operator to manage the process of resource allocation within multiple dams. The system operator is considered as a centralized agency that provides the resources to a dam, or more, to increase their resilience and hence can avoid long interruptions to the electric service. Having a system operator that can manage the resources, especially drones, is useful as the operation of drones is regulated by the federal aviation administration (FAA) [155] and is not granted to all private organizations. In the following, we will use a general notion of resources, without being restricted to drones, as the framework can be applied to any

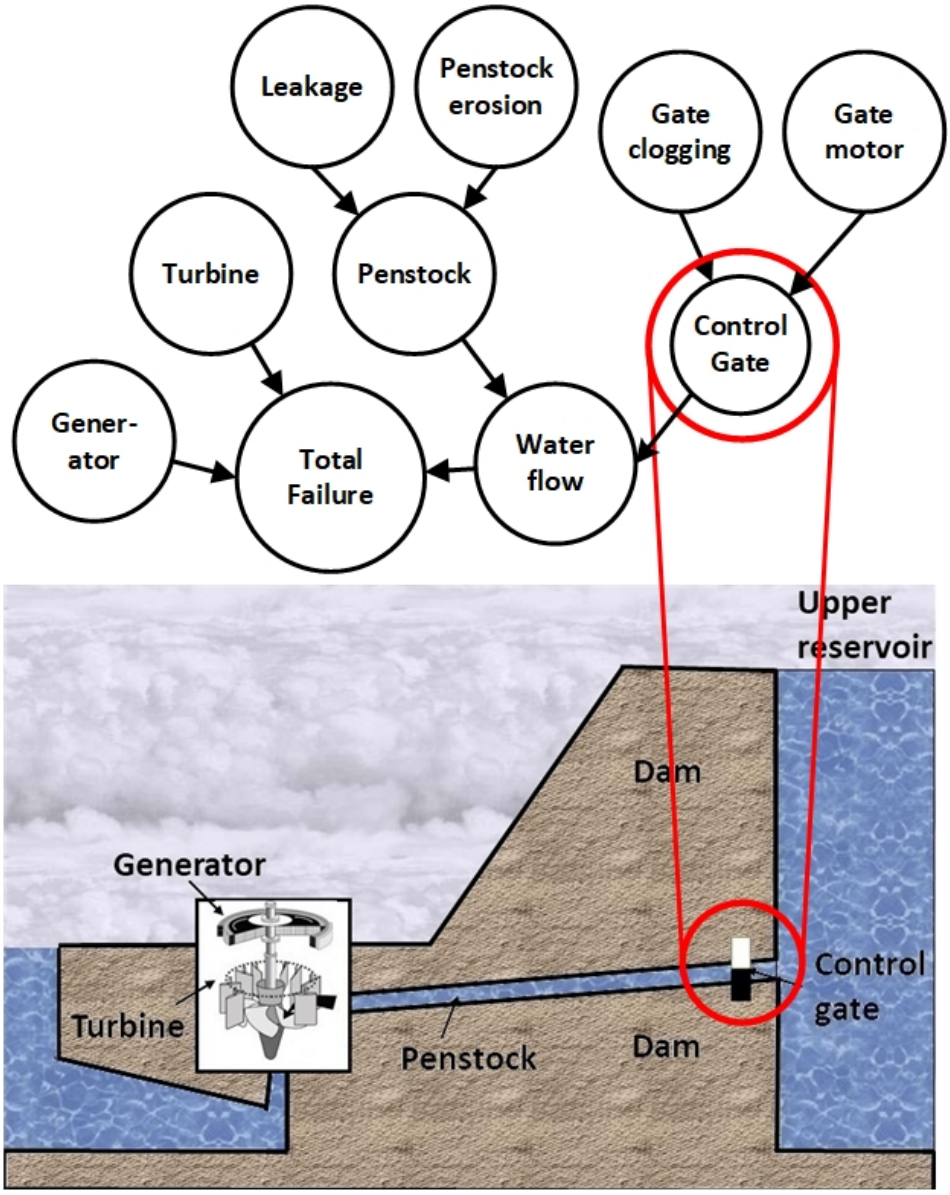


Figure 5.9: A Bayesian network model for the hydropower dam in which each node defines the failure probability for one of the physical components all the way to the total CI failure.

type of resources.

Next, we formulate the problem of allocating resources within a system of multiple dams (CIs). We propose to use *contract theory*, a powerful framework from microeconomics that provides useful tools for designing contractual agreements between a principal and a number of agents [139]. Contract theory is chosen as it allows the system operator to maximize its reward in light of the individual rewards of the owners of the dams. In the formulated problem, the system operator is modeled as the principal and the owner of dams as agents, as discussed next in more details.

5.4.2 Resource Allocation using Contract Theory

We consider a system in which an electric grid operator, referred to as the principal, is interested in providing a number of resources to the owners of dams to be used in the process of surveillance and rapid intervention. Let $\mathcal{N}$ be the set of N targeted dams. Dams are assumed to have different owners. These dams, being part of the grid, sell their generated electricity to the power grid managed by the principal. Each dam can utilize the resources to improve its resilience and hence reduce the probability of failure to generate electricity.

The principal has a limited number of discrete (integer valued) resources R to be allocated to the dams and, hence, it decides on how to optimally use these resources. The goal of the principal is to invest in improving the probability of power generation and, hence, decreasing the probability of losses due to a dam's failure. We model the principal's payoff as the difference between the total rewards it gets from the dams' owners and the total expected losses due to each dam's failure. Losses are modeled as a function of the total probability of failure before and after the deployment of resources. The total utility $Z_p(\boldsymbol{R})$ that the principal achieves as a function of the vector of resource allocation $\boldsymbol{R}$ is given by:

$$Z_p(\boldsymbol{R}) = c \cdot \sum_{i=1}^{N} R_i - \sum_{i=1}^{N} (\alpha_{f_i} - \alpha_R) \cdot B_i(R_i) \cdot n_i \cdot P_i, \tag{5.18}$$

where $\boldsymbol{R}$ is the resource allocation vector across all dams with each element R_i specifying the number of resources allocated to dam i, α_{f_i} is the expected real-time energy price if dam i fails to generate electricity, α_R is the average real-time energy price in normal operation, P_i is the contracted power production for dam i, n_i the expected number of hours the dam will be out-of-service due to failure, and c is the monetary reward the principal gets for a unit of resources which can also be seen as a cost. Note that, the resources in (5.18) refers to monitoring resources or drones as discussed earlier. We introduce the function $B_i(R_i)$ to measure the improvement in the resilience of a dam i due to the amount of allocated resources. Specifically, the dam's owner evaluates the difference in the dam's resilience index before and after using the resources R_i, and $B_i(R_i)$ is given as:

$$B_i(R_i) = \gamma_{i_{R_i}}^{-1} - \gamma_i^{-1} = v_{R_i}^F - v^F, \tag{5.19}$$

where $\gamma_{i_{R_i}}$ and $v_{R_i}^F$ are the values calculated from (5.10) and (5.11) respectively for the updated values of P_{WS}. These updated values are calculated from the Bayesian network as the result of fixing a number of variables equal to R_i according to the order specified by Algorithm 1. The effect of the first unit of resources on P_{WS}, for each dam, is calculated from (5.16) while the effect of the remaining resources is calculated from (5.17) for each additional unit of resources. Without loss of generality, we assume that each component of a dam can be secured by a single unit of resources. Note that, if a unit of resources can be used to monitor or fix multiple components, the utility function can still be used with a slight modification. In this case, the accumulated effect of fixing all the components should be considered in (5.19) for this specific unit of resources. This also requires the dam to fix these components simultaneously and update its Bayesian network before using Algorithm 1 to consider the remaining components in the further steps.

Each dam's owner will evaluate the amount of resources it receives based on the resilience enhancement that will result from the allocated resources. This resilience improvement is reflected by a higher probability of generating power and, hence, a higher probability to sell the generated power with the real-time prices. The utility $Z_{d_i}(R_i)$ of dam i is defined as follows:

$$Z_{d_i}(R_i) = \alpha_{d_i} \cdot B_i(R_i) \cdot n_i \cdot P_i - c \cdot R_i, \tag{5.20}$$

where α_{d_i} is the average day-ahead energy price for dam i. The remaining parameters are similar to those in (5.18).

The principal wants to offer contracts to the dam's owners to maximize its utility in (5.18). A *contract* [139] can be seen as an agreement between the principal and the dam's owner using which the principal provides and operates resources to monitor, inspect, and fix points of interest in the dam and gets monetary rewards in return. Every contract is defined as a pair $(R_i, c \cdot R_i)$ representing the amount of resources and the monetary reward (cost) the dam's owner should pay for these resources.

In our model, we assume the principal has complete information about the targeted dams. This information should be provided by each dam's owner as the resource evaluation, from the Bayesian network, is dam-specific and cannot be estimated by the principal without the dam owners. Moreover, the principal, being the system operator, already knows all of the other parameters. Hence, the focus of the principal is to design contracts in a way to ensures each dam's owner participation in order to maximize its total benefit. Contracts offered by the principal should then satisfy the key property of *individual rationality*, under which each dam's owner is interested in participating only if the benefit it gets is greater than or equal to the amount it pays , i.e.,

$$\alpha_{d_i} \cdot B_i(R_i) \cdot n_i \cdot P_i - c \cdot R_i \geq 0. \tag{5.21}$$

The principal can then design the optimal contracts by maximizing its utility and satisfying the constraints as follows:

$$\max_{R_i} \quad c \cdot \sum_{i=1}^{N} R_i - \sum_{i=1}^{N} (\alpha_{f_i} - \alpha_R) \cdot B_i(R_i) \cdot n_i \cdot P_i, \tag{5.22}$$

$$\text{s.t.} \quad \alpha_{d_i} \cdot B_i(R_i) \cdot n_i \cdot P_i - c \cdot R_i \geq 0, \; i \in \mathcal{N},$$

$$\sum_{i=1}^{N} R_i = R.$$

Note that, for cases in which the principal fully owns and operates the dams, the principal can still use (5.18) to calculate its benefits from allocating the resources. However, in this case, (5.21) will no longer be needed as it will no longer be possible for a dam owner not to accept a contract. The problem in (5.22) can be easier to solve in this case as any resource allocation that benefits the principal, will be valid.

5.4.3 Optimal Contract

Solving the problem in (5.22) is challenging as the function $B_i(R_i)$ is not continuous. $B_i(R_i)$ has discrete values for a finite set of R_i values. To address this challenge, we first start by inspecting the properties of the function $B_i(R_i)$ in order to solve the problem in (5.22).

Proposition 2. *The values of the function $B_i(R_i)$ represent a monotonically increasing concave sequence.*

Proof. We prove this proposition by showing how the values of $B_i(R_i)$ are calculated. Let the values a_{i-1}, a_i, a_{i+1} be any three consecutive values for the improvement in P_{WF} achieved by fixing any three consecutive variables as calculated from Algorithm 1. These values satisfy the following two properties:

$$\begin{aligned} a_{i-1} &\geq a_i \geq a_{i+1}, \\ a_{i-1} - a_i &\geq a_i - a_{i+1}, \end{aligned} \tag{5.23}$$

according to the selection criteria defined in Algorithm 1 in which the biggest improvement is captured first.

Let b_{i-1}, b_i, b_{i+1} be the values calculated from (5.19) for the updated P_{WS} values for a_{i-1}, a_i, a_{i+1}, respectively. Each b_i is the difference between the updated $v^F_{R_i}$ and the current v^F. According to (5.12), the values of $v^F_{R_i}$ are inversely proportional to the P_{WS} values, hence, the values b_{i-1}, b_i, b_{i+1} follow the same relation and it can be seen that $b_i = f(\frac{1}{a_i})$. Therefore, we can conclude that the sequence is monotonically increasing:

$$b_{i-1} \leq b_i \leq b_{i+1}, \tag{5.24}$$

and the difference relation becomes:

$$b_{i+1} - b_i \leq b_i - b_{i-1}. \tag{5.25}$$

Rewriting the last inequality we get:

$$b_{i+1} + b_{i-1} \leq 2 \cdot b_i, \tag{5.26}$$

which proves that the sequence is concave [156]. □

Using Proposition 2 with the first constraint in problem (5.22), we can see that the constraint is a difference between a monotonically increasing concave sequence $\alpha_{d_i} \cdot B_i(R_i) \cdot n_i \cdot P_i$ and a strictly increasing linear sequence $c \cdot R_i$. The result will be a concave sequence that can have both positive and negative values depending on the difference between the two sequences. This result is used by the principal to determine the range of values, i.e. $[R_{i_{\min}}, R_{i_{\max}}]$, that meets the first constraint and, hence, will be acceptable for the dam's owners as it satisfied individual rationality.

Next, we study the properties of the objective function in (5.22) based on the results of the previous proposition.

Lemma 4. *The objective function in (5.22) is a convex sequence with respect to each dam that is monotonically increasing.*

Proof. We prove this lemma by showing the relation between the terms of the objective function. For a given dam i, the objective function is the difference between the reward $c \cdot R_i$ and a constant number $(\alpha_{f_i} - \alpha_R) \cdot n_i \cdot P_i$ multiplied by the concave function $B_i(R_i)$. The reward term represents a linear strictly increasing function in the number of resources R_i, while the second term is monotonically increasing concave function. Clearly, the difference between both terms will be a monotonically increasing convex sequence. □

According to Lemma 4, while being convex, the objective function might have negative values until a certain amount of resources is used, that is when the second term is higher than the rewards term. The principal can use this value to update the minimum number of resources, i.e. $R_{i_{\text{min}}}$, that should be allocated to each dam to represent a feasible solution to the principal. The update is done based on the larger of the two minimum values calculated in proposition 2 and Lemma 4.

After determining the range of possible values for each allocation, we propose to use dynamic programming optimization [157] techniques to calculate the solution to the problem in (5.22). In our dynamic programming representation, stages will represent the current allocation of resources for each dam and the state of each stage will be the current value of the objective function. The update from a stage to another, i.e., from a specific allocation to another, aims at increasing the value of the objective function. If no increase can be achieved at one stage, then the current allocation is the optimal. This is proved analytically in the next theorem.

Theorem 4. *The optimal resource allocation can be found using dynamic programming by updating the number of resources assigned to each dam at each stage while maximizing the benefits of each allocation.*

Proof. The values of the objective function are calculated at each value of the resources R_i in the feasible range for each dam i, $[R_{i_{\text{min}}}, R_{i_{\text{max}}}]$. These values are stored for all dams as a matrix of size $N \cdot R$. Each value $z_{i,k}$ in the matrix is the objective function value evaluated for dam i when it is assigned a number of resources k. The values of each row represent a monotonically increasing sequence as shown in Lemma 4.

The first stage in the problem starts by allocating the maximum feasible resources k to the dam i with the highest objective function value, i.e., $R_i = k$ for the dam with $R_{i_{\text{max}}} = k$ and $z_{i,k}$ is the largest among all other dams. As the values for this dam i represent a convex sequence and are monotonically increasing, the principal will not gain more by assigning a lower number of resources to this specific dam. Since this value of $z_{i,k}$ is the maximum among all dams, this allocation represents the maximum value that the principal can get for this number of resources k. The rest of the resources, i.e., $R - k$ are assigned to dam j having the largest $z_{j,R-k}$ among all dams.

This allocation represents the optimal solution at the first stage as the principal gets the highest utility for the current resource configuration. The principal then tries to get a higher utility by changing the current allocation scheme. The principal might be able to do so by decreasing the number of resources k assigned to dam i and assigning the difference to another dam j if the following condition holds:

$$z_{i,k} - z_{i,k-1} < z_{j,R-k+1} - z_{j,R-k}\ , j = 1, \ldots, N, j \neq i, \tag{5.27}$$

where k decreases by one unit of resources each time. The principal compares the utility gains that it can achieve by assigning more resources to another dam. These resources are taken from the dam with most resources. The current allocation ensures that the principal gets the largest utility from the allocated resources, so it is the optimal solution at this stage. Here, if the principal cannot increase its utility by changing a unit of resources, then it will try to change more than one at a time until the condition is satisfied or all the values of resources are checked.

The procedure continues at each stage by assigning less resources to the dam with the most resources if a higher utility can be achieved by allocating these resources to another dam. This ensures that the principal achieves its maximum utility at each stage. This procedure by starting at the final allocation and moving backward ensuring the maximum utility is achieved at each stage satisfies the Bellman equation [158] which is the necessary optimality condition in dynamic programming, Hence, the procedure achieves the optimal resource allocation. □

The complexity of calculating the optimal resource allocation for the previous dynamic programming problem is $O\big(N \cdot (R_{\text{max}} - R_{\text{min}})\big)$ where:

$$\begin{aligned} R_{\text{max}} &= \max(R_{i,\text{max}}),\ i = 1, \ldots, R, \\ R_{\text{min}} &= \min(R_{i,\text{min}}),\ i = 1, \ldots, R, \end{aligned} \tag{5.28}$$

as for each number of resources in the range $[R_{\text{max}} - R_{\text{min}}]$, the program at most compares the utility function N times for each of the N dams.

Finally, we summarize our framework steps in the flow chart shown in Figure 5.10. Note that the problem discussed in this section, though discussed within the context of a case study, can be to used to allocate resources in other systems of multiple CIs. Selecting a specific CI, hydropower dams here, helped to design meaningful Bayesian networks and to define CIs utilities in the contract-based allocation. However, other systems can utilize the same framework after modeling suitable Bayesian networks and defining new utilities that can capture the different parameters within the other systems.

Next, we show some numerical results built on the selected case study, i.e., hydropower dams.

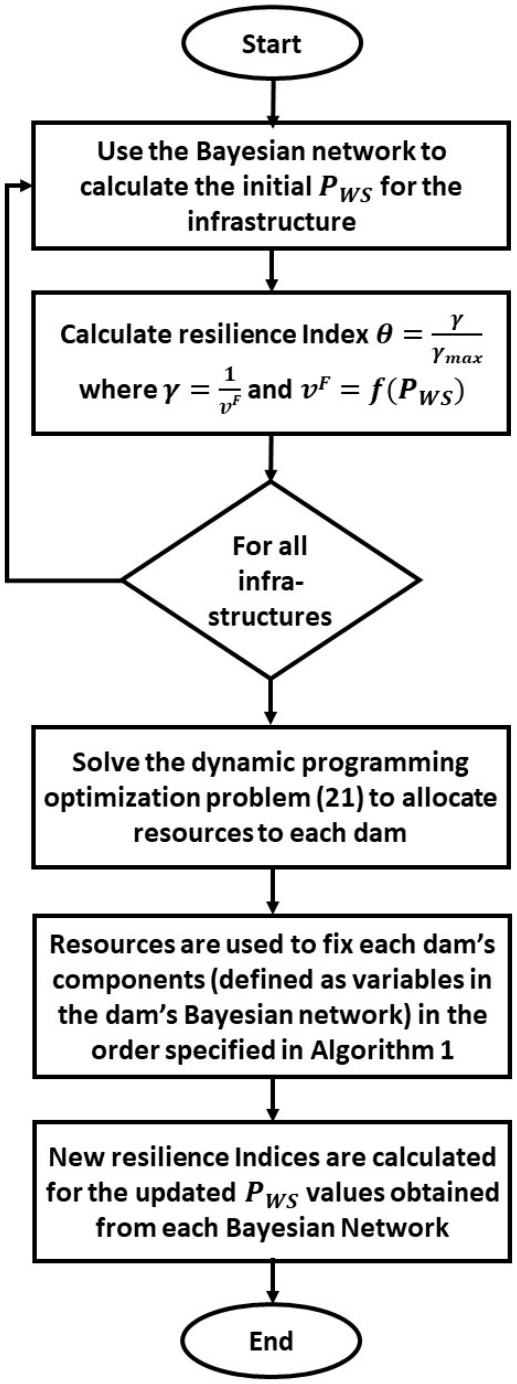


Figure 5.10: Flowchart for the proposed mechanism.

5.5 Numerical Analysis and Results

Although our framework can be applied to any number of dams, for our simulations, we consider a case of two dams, in order to better highlight each dam's effect on the process of resource al-

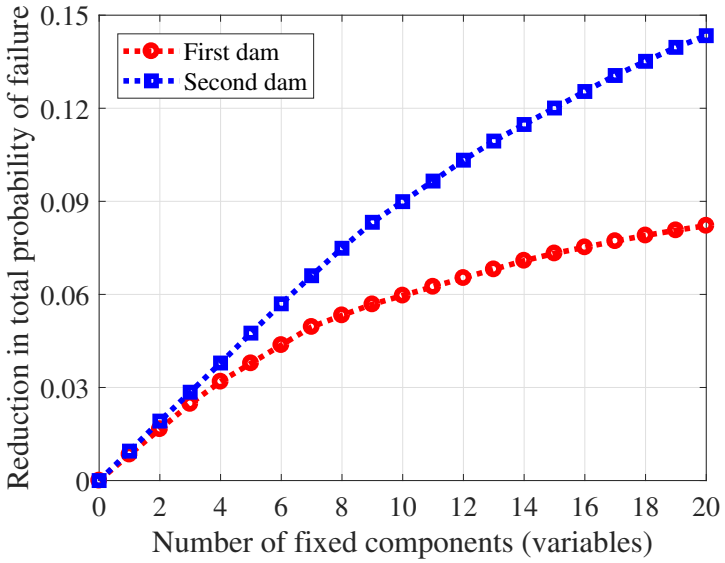


Figure 5.11: Dam's probability of failure improvement with the number of overhauled components (their related variables are adjusted in the Bayesian network).

location. Two Bayesian networks are designed, using SamIam software [159], for the two dams using similar components but with different probabilities. In the following experiments, the probability P_{FW} is fixed to zero and the transition probabilities $P_{SS} = 0.8, P_{SW} = 0.15, P_{FS} = 0.5$, and $\epsilon = 0.1$ are assumed to be the same for both dams to neutralize their effect on the results. However, the first dam is assumed to have a lower initial $P_{WF} = 0.37$ while the second dam will have $P_{WF} = 0.7$, as calculated from their corresponding Bayesian networks. Other parameters are set as follows: $\alpha_{d_1} = \$26, \alpha_{d_2} = \$20, P_1 = 120$ MW/h, $P_2 = 150$ MW/h, $n_1 = 30$ hours, $n_2 = 20$ hours, $\alpha_R = \$33, \alpha_{f_1} = \$40$, and $\alpha_{f_2} = \$46$. α_{f_2} is assumed to be higher than α_{f_1} as P_2 is assumed to be higher than P_1, so the failure of the second dam will have a larger effect on increasing the prices of power. In what follows, MATLAB is used to implement our dynamic programming solution and SamIam is used for the Bayesian network related analysis.

In Figure 5.11, we show how the total probability of failure v^F can be reduced by overhauling each dam's components. The components are overhauled using the allocated resources in the order specified by Algorithm 1 then the variables representing these components are adjusted in the Bayesian network. Note that, when v^F decreases, both the resilience γ and the resilience index θ increase according to (5.11) and (5.13). We can see that the second dam achieves a higher reduction in the probability of failure v^F. This because the second dam has a higher initial P_{WF}, so the difference between the initial and final P_{WF} is higher resulting in a higher difference in v^F. This difference represents the function $B_i(R_i)$ as in (5.19). Figure 5.11 also corroborates Proposition 2 by clearly showing that the improvement in each dam follows a monotonically increasing concave sequence.

Figure 5.12 shows the benefit that each dam receives from the allocated resources, evaluated as the first term of (5.20) before subtracting the cost, which is a function of $B_i(R_i)$. The figure shows

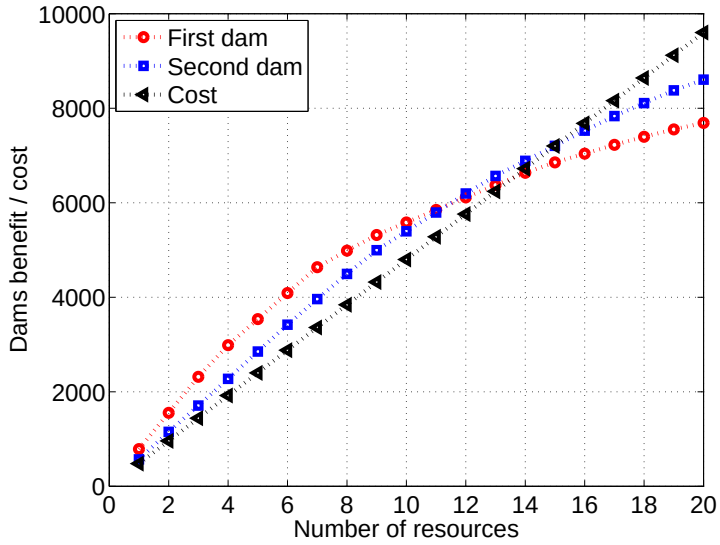


Figure 5.12: Dam's benefit and resources cost. The intersection represents the range of resources each dam is willing to accept.

the cost of the resources separately. The intersection between the cost and each dam's benefit will represent the range of values $[R_{i_{\min}}, R_{i_{\max}}]$ that each dam i is willing to accept. Any additional resource beyond $R_{i_{\max}}$ will yield a negative dam's utility as the cost will be higher than the dam's benefit. The range can be seen from Figure 5.12 to be $[0, 13]$ and $[0, 15]$ for the first and second dams, respectively. The first dam has a smaller range as its utility is lower than the second dam, starting at 12 units of resources. This utility is lower as the first dam has a smaller power production P_1 and a higher initial resilience index that causes the changes in $B_1(R_1)$ to be small.

In Figure 5.13, we show the utilities of the dams as given by (5.20). We can see that both dams have nonnegative utilities only in the ranges discussed before, i.e., $[0, 13]$ for the first dam and $[0, 15]$ for the second dam. Figure 5.13 also shows that both utilities are concave and each has a maximum value at a certain amount of resources. The first dam has its maximum utility when it uses 7 units of resources, while the second dam can achieve its maximum at 9 units of resources. These values represent the maximum distance between the benefit and the cost in Figure 5.12. Note that, according to the proposed framework, the owners of the dams are willing to accept resources in their feasible ranges regardless of their maximum utility. This is because the extra resources will help improve their resilience.

In Figure 5.14, we show the principal's utility calculated for each dam. Figure 5.14 shows two curves: one for each dam where the second curve is plotted upside down. The horizontal axes show the amount of resources allocated to each dam, while the remaining resources are allocated to the other dam. Therefore, the principal's total utility, at each allocation, is the summation of the values from the two curves corresponding to this allocation. Figure 5.14 corroborates the result of Lemma 4 where we showed that the principal's utility calculated for each dam separately represents a monotonically increasing convex sequence. From Figure 5.14, we can see that the

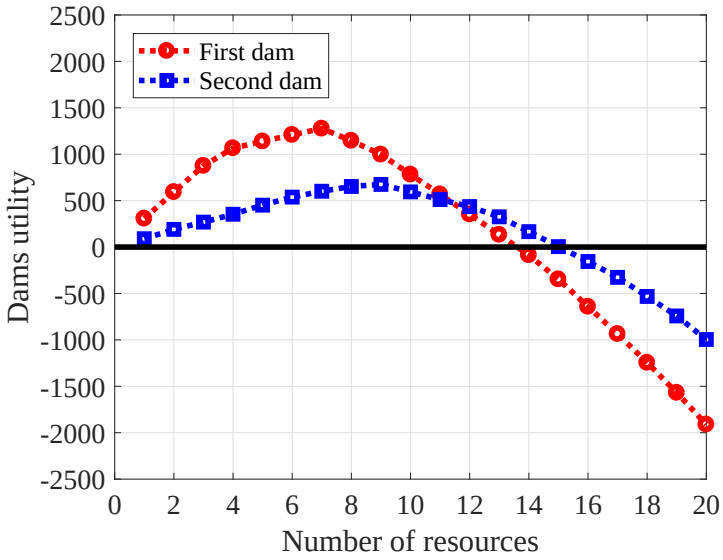


Figure 5.13: Dam's utilities relation with the number of allocated resources.

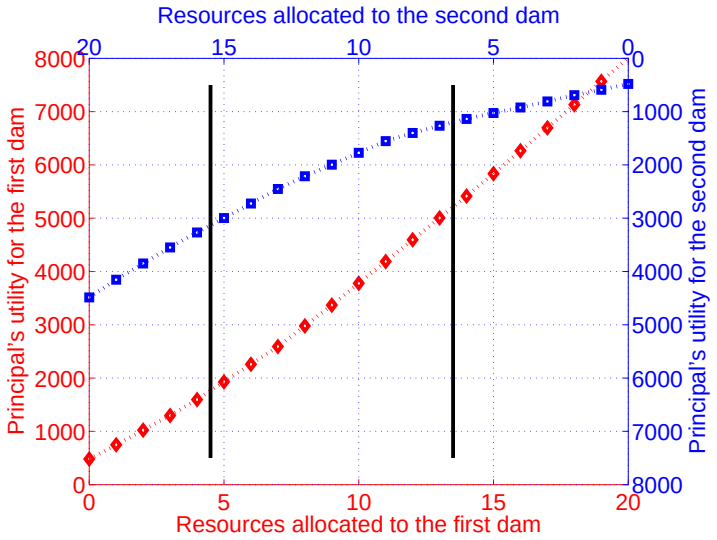


Figure 5.14: The principal's utility for each individual dam with respect to the number of the allocated resources to this dam.

principal achieves a higher utility by allocating resources to the first dam. This stems from the fact that the first dam has a lower power production P_1 and a higher initial resilience, i.e., lower $B_1(R_1)$. Figure 5.14 also has two solid vertical lines, each of which representing the maximum number of resources $R_{i_{\max}}$ for a dam. Any allocation of resources beyond these lines will no longer be feasible as it yields negative dams' utilities. The lines correspond to the maximum resources in the feasible range for each dam (13 for the first dam and 15 for the second dam). The optimal allocation in this case is to allocate 13 units of resources to the first dam and 7 units of resources to the second dam.

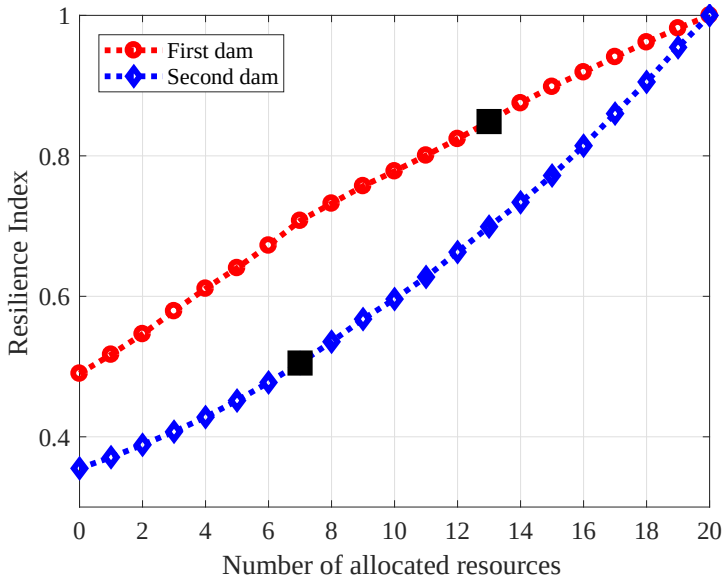


Figure 5.15: Resilience Index for each dam as a relation in the amount of allocated resources to each dam.

In Figure 5.15, we show the resilience index as a function of the amount of resource allocated to each dam. The horizontal axis shows the resources allocated to each dam separately. The two curves show the possible resilience index improvements for both dams. Figure 5.15 shows that the first dam has a higher initial resilience index, however, theoretically, both dams can reach their maximum resilience index. The values of the resilience indices for both dams, achieved at the optimal resource allocation, are marked with black squares. From Figure 5.15, we can see that the first dam's resilience index at the optimal allocation equals 0.85, while the second dam achieves a resilience index of 0.5. This is due to the fact that the first dam has a higher initial resilience index and it is allocated more resources. Figure 5.15 shows that the first dam achieves about 70% increase over its initial resilience index, while the second dam achieves about 50% increase over its initial resilience index. This makes the average increase of the resilience index in the system about 60%.

We next study the effect of varying the reward that the principal charges for a unit of resources on the optimal solution. We apply the same parameters as the previous experiments but the reward per resources is now varied from $\$100$ to $\$800$.

Figure 5.16 shows the principal's utility when applying the proposed allocation, its utility when allocating resources to only one of the dams, and when using the greedy algorithm to allocate the resources. We see that the principal can achieve its highest utility at the value of $\$700$ per a resources unit, when using the proposed allocation. On the other hand, the value of $\$100$ is shown not to be enough for the principal to achieve a positive utility. The values in the range $[\$200 - \$400]$ yield a negative utility for the second dam, therefore the optimal allocation is to allocate the maximum amount to the first dam. In the range $[\$400 - \$700]$, both dams can achieve

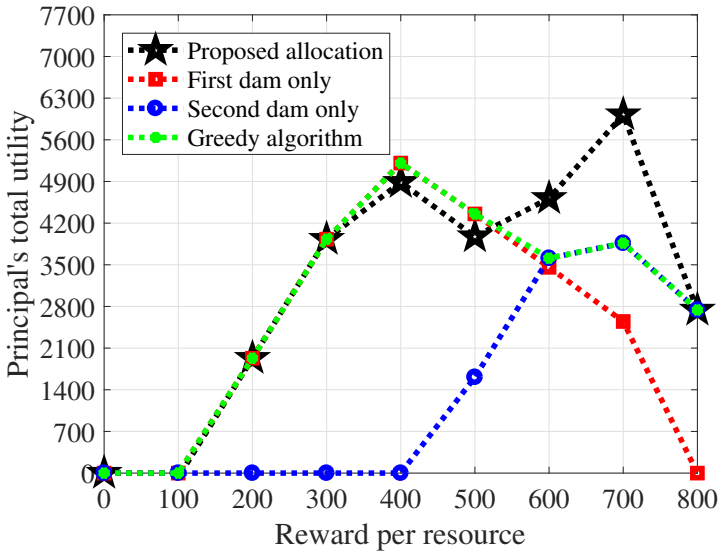


Figure 5.16: Principal's utility as a relation in the reward charged per unit of resources.

positive utilities and, hence, the solution involves both dams in the process of resources allocation. At the value of $\$800$, the first dam will achieve negative utility so resources are allocated to the second dam only, i.e., its maximum allowed value. It can also be seen that the greedy algorithm coincides with the maximum of single dam allocations (first dam only and second dam only). This is because the greedy algorithm tries to achieve the maximum benefit from each unit of resources, so it allocates each unit to either the first or the second dam whether achieves a higher utility for the principal.

From Figure 5.16, we can also see that the principal can achieve a higher utility if it allocated all the resources to the first dam for reward values of $\$400$ and $\$500$. This is because the proposed solution is primarily centered around improving the resilience index and not maximizing the principal's reward. Hence, it allocates all of the available resources, as long as the principal achieves a positive utility. From Figure 5.16, we can see that the proposed solution allocates 18 and 12 to the first dam for the reward values of $\$400$ and $\$500$, respectively. The remaining resources, i.e., 2 and 8 respectively are allocated to the second dam although they caused the principal's utility to be lower.

In Figure 5.17, for comparison purposes, we introduce a slight modification to our dynamic programming procedure to find the optimized solution from the rewards point of view. The main difference between the reward-optimized allocation and our original proposed allocation is that the principal does not have to allocate all the available resources. Instead, the principal allocates resources up to the limit that keeps its utility increasing. For instance, at a reward value of $\$400$, the reward-optimized allocation assigns 18 units of resources to the first dam and nothing to the second dam, compared to 18 and 2 in the original proposed allocation. This helps the principal to achieve a higher utility at $\$400$ and $\$500$ as shown in Figure 5.17. This reward-optimized solution coincides with the first dam's single allocation in Figure 5.16 for the same rewards range.

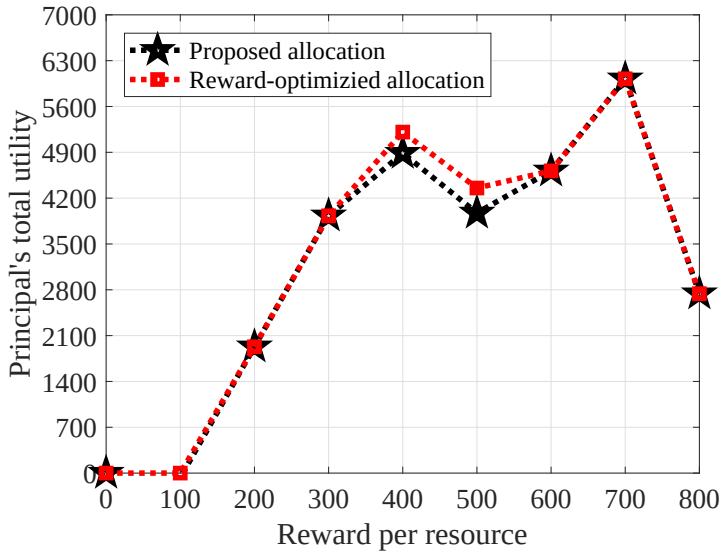


Figure 5.17: Principal's utility under the proposed allocation and the introduced reward-optimized allocation.

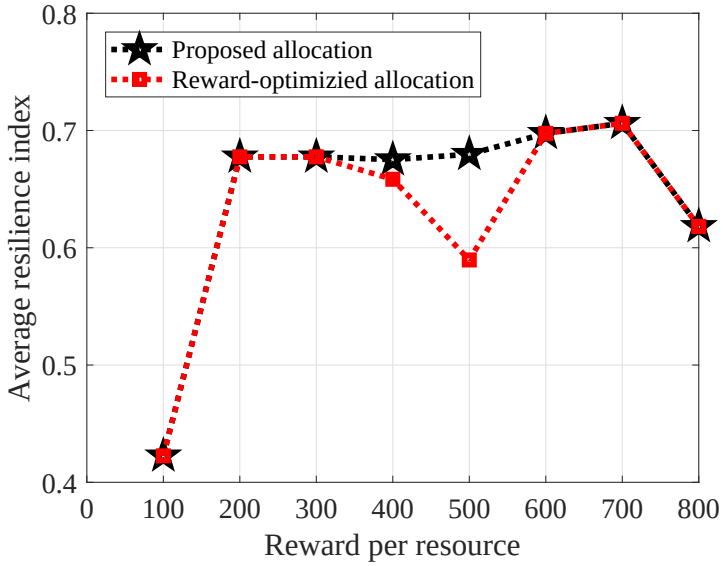


Figure 5.18: Average resilience index under the proposed allocation and the introduced reward-optimized allocation.

The extra reward achieved using the reward-optimized allocation comes at the cost of the dams' resilience. Figure 5.18 shows the average resilience index for both dams when using the two allocations. We see that at reward values of $\$400$ and $\$500$, the average resilience index of the reward-optimized allocation is 3% and 13% less than our proposed allocation, respectively. This is because less resources are used and, hence, dams can achieve less resilience improvement.

Next, we show the average resilience index multiplied by the principal's utility to show the com-

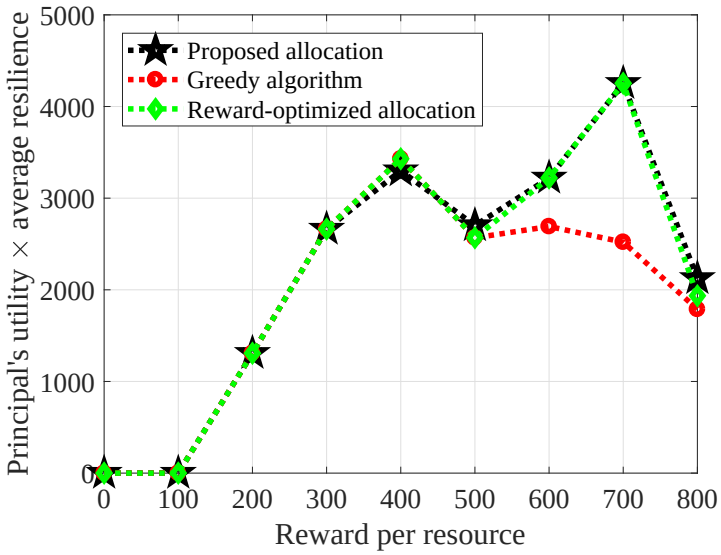


Figure 5.19: The average resilience utility for the principal with the rewards value.

bined effect of both, we call this *the average resilience utility* for the principal. Figure 5.19 shows that the gap between the proposed allocation and the reward-optimized allocation is smaller than the case of comparing just the utilities. This happens as the proposed mechanism allocates all the available resources which helps increase dams' resilience indices and hence the average. In the reward-optimized allocation, some resources are not allocated if they will cause the principal's utility to go lower, hence, dams achieve lower resilience indices and the average will be lower. From Figure 5.19, we can see that the reward-optimized allocation slightly outperforms the proposed-allocation in the average resilience utility only at the value of $\$400$. It is slightly lower at the value of $\$500$ and coincides with the proposed allocation at all the other values. It is also clear from Figure 5.19 that our proposed allocation outperforms allocating resources using the greedy algorithm in terms of the combined effect of principal's reward and dams' resilience. In particular, for the reward values of $\$600$ and $\$700$, the proposed allocation achieves 18% and 68%, respectively, higher average resilience utility compared to the greedy algorithm.

Finally, to shed more light on the effect of different transition probabilities in (5.8) on the resilience index values, we consider the case in which both dams have low probabilities for remaining in the success state and, hence, high probabilities of transitioning to a warning or a failure state. Figure 5.20 shows the average resilience index when the transition probabilities become $P_{SS} = 0.3$, $P_{SW} = 0.3$, and $P_{SF} = 0.4$. These numbers are chosen to represent dams that are not well-maintained and, hence, are more prone to failures than the dams in Figure 5.18. We can see in Figure 5.20 that the average resilience index is higher than that in Figure 5.18 because the dams have high initial probabilities of being in a warning state, thus their initial resilience indices are higher than that in 5.18. We can also see from Figure 5.20, that the proposed allocation achieves higher resilience indices compared to the reward optimized allocation especially at the values of $\$400$ and $\$600 - \$800$. This is because, at these values, the reward optimized allocation does not

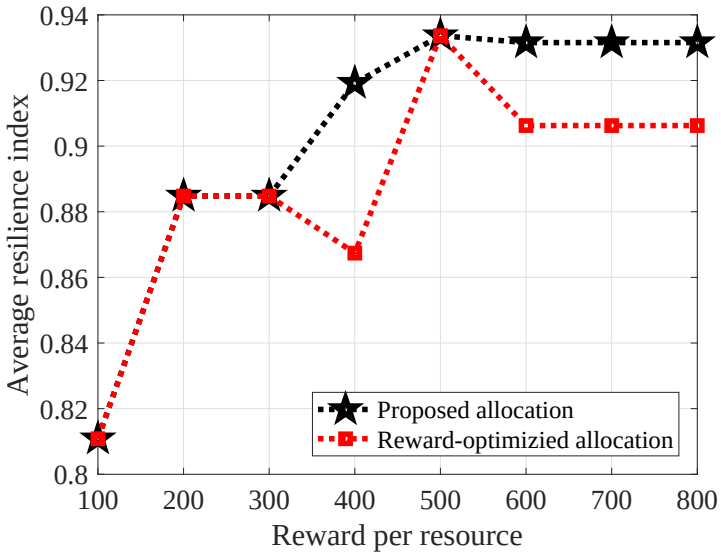


Figure 5.20: The average resilience index for the principal with the rewards value (low transition probabilities

use all of the available resources as this will reduce the principal's utility.

Here, we note that finding the solution of the reward-optimized allocation increases the complexity of the dynamic programming optimization problem to $O\big(N \cdot (R_{\max} - R_{\min})^N\big)$ as the principal needs to check all partial resource allocations. This significantly increases the solution space and the time needed to reach the optimal solution. Moreover, this allocation will achieve a lower average improvement in the resilience index as less resources are allocated. Note that, the last constraint in (5.22) needs to be relaxed to $\sum_{i=1}^{N} R_i \leq R$ to allow for partial allocations. Given the complexity of the reward-optimized allocation and the limited improvement it can achieve over our proposed allocation, the proposed allocation will be superior in allocating the resources to a system of multiple dams.

5.6 Summary

In this chapter, we have proposed a novel framework to study and optimize the resilience of CIs. A novel resilience index has been introduced that is derived from a Markov chain representing the infrastructure's performance state. The state is defined to be either success, warning, or failure. The framework focuses on the effect of the probability of transition from warning to failure on the resilience index. We have then proposed a Bayesian network to model the infrastructure's physical components and their effect on the resilience index. To prioritize the infrastructure's components in the resilience improvement process, we have introduced a Bayesian network algorithm that captures the effect of each component on the infrastructure's probability of failure. We have evaluated the proposed framework in a case study of hydropower dams. We have defined a problem of al-

locating resources to a system of multiple CIs and studied it within the context of the case study. The problem is modeled using contract theory in which a system operator wants to maximize the economic benefit from allocating the resources to CIs. Dynamic programming optimization has been used to derive the optimal solution for the problem of resource allocation. Results have shown that the proposed framework outperforms other allocation methods both in the economic reward for the system operator as well as the average resilience utility.

Chapter 6

Contract-Theoretic Resource Allocation for Critical Infrastructure Protection

6.1 Background, Related Works, and Contributions

As previously discussed in Section 1.5.5, the problem of resource allocation within multiple CIs will be discussed for the asymmetric information case. We discuss one main challenge in CI protection (CIP) is when the system operator does not have enough information of the CIs and it has only a limited amount of resources, such as personnel or even cyber resources, that can be used for CIP. Under such resource constraints and given the complex nature of CIs, it is imperative to develop practical resource management mechanisms that can optimally allocate such resources, given the criticality levels and vulnerabilities of the various CIs. Such resource deployment strategies are particularly critical for protecting CIs that are based in foreign countries or remote sites. In such scenarios, a system operator who want to protect local and foreign CIs will often own a control center (CC) that is responsible for monitoring these CIs and distributing resources among them. One major challenge for resource deployment here, is the fact that the CIs are often owned by different entities that *consider their own CIs to be the most critical*. Indeed, every CI owner will report to the CC that its own infrastructure is the most vulnerable and most critical. Determining real levels of vulnerability and criticality of each individual infrastructure is very challenging for the CC. However, intuitively, the CC should design a proper mechanism to allocate resources based on the vulnerability and criticality levels of each CI. For example, highly vulnerable CIs should get higher resources than less vulnerable ones. Similarly, highly critical CIs must be properly prioritized in the CIP process. However, as each CI will attempt to get as much resources as possible by claiming that it is the most vulnerable or critical, the CC may not be able to properly distribute its limited resources.

The problem of resource allocation for CIP has been studied in recent works such as [160] and [161]. The work in [160] studied an optimal resource allocation scenario in which resources were

allocated to CIs depending on the likelihood of these CIs to be attacked according to their valuation on possible attackers. The authors in [161] studied the problem of allocating resources where resources were supposed to protect an area and the objective was to maximize the area protected by these resources. Despite their importance, these existing works did not address the problem of asymmetric information in resource allocation which was first studied in [162]. The problem discussed in [162] cannot be generalized to a large-scale CIP system as it addresses an isolated problem and it does not take into account the impact of information availability on resource allocation.

In contrast to these works, here, we propose a contract-theoretic model to allocate resources for CIP under asymmetric information. Contract theory is a powerful framework from microeconomics that provides a useful set of tools for modeling mechanisms under information asymmetry [139]. The key idea is that the CC should offer right contracts to CIs so that they have the incentive to truthfully reveal their information. In our model, the CC is seen as the principal that offers contracts to agents which are the CIs. While contract theory has been studied in the context of wireless networks [163, 164], such works do not apply directly to CIP and their results cannot be generalized for accounting for criticality and vulnerability levels of the CIs.

The main contribution of this chapter is to propose a resource allocation mechanism for CIP that can optimally allocate resources between a number of CIs without knowing their exact types. The problem is formulated using a contract-theoretic model in which the CC calculates the amount of resources that will be given to each CI and offer contracts to CIs with these values and the rewards they should reciprocate. In particular, we proposed a novel approach to define a CI by accounting for two different CI types according to the vulnerability and criticality levels. Both types are used in the process of resource allocation and the criticality level is also used to prioritize CIs that will be protected. For the formulated problem, we analyzed the necessary and sufficient conditions for deriving the optimal contracts. We studied the optimal contract and we proved that the problem has an optimal solution for the case of two CIs. The model was also shown to motivate each CI to reveal its actual type and accept the contract designed for its type, therefore allowing resource allocation in the absence of exact information at the CC on the criticality and vulnerability levels of the CIs. Simulation results show that the proposed approach will yield a higher CC utility when compared with a baseline resource allocation algorithm.

The rest of the chapter is organized as follows. Section 6.2 provides the system model and defines the vulnerability and criticality levels of the CIs. In Section 6.3, the problem is formulated as a contract-theoretic mechanism and several properties are derived and analyzed. Simulation results are discussed in Section 6.4. Finally, conclusions are drawn in Section 6.5.

6.2 System Model and Problem Formulation

We consider a system in which one control center (CC), that can represent a government agency is interested in sending missions to secure N CIs in a set $\mathcal{N}$ that can be owned by different entities

(e.g., foreign agencies, different department of defense agencies, etc.). The missions are viewed as *resources* owned by the CC and that must be allotted to different CIs. Such resources can be personnel or cyber resources. Each CI has some vulnerable points that need to be protected. As the number of vulnerable points of a CI increases, the amount of resources needed to protect it will also increase. Infrastructures are classified into groups according to their vulnerability levels. The vulnerability level can be represented by an integer number w_i where there are M different levels in the set $\mathscr{M}$ and $M \leq N$; thus yielding different M vulnerability levels. Infrastructures are grouped by an increasing order of vulnerability levels:

$$w_1 < \ldots < w_i < \ldots < w_M. \tag{6.1}$$

A higher w implies that the CI has a higher vulnerability level. The CC *does not have exact information* on the individual w_i of every CI i. Instead, the CC can know with which probability a certain CI can belong to a certain w type . Therefore, we let p_{i,w_j} be the probability with which CI i belongs to a certain type w_j.

Each CI has also a criticality level that can be represented by a number θ_i where there are different K levels in the set $\mathscr{K}$ and $K \leq N$. Thus there exists K criticality levels to which various CIs can belong. The criticality level is determined by factors such as the service performed by this CI and its relation to other CIs. The CIs are grouped by an increasing order of criticality levels:

$$\theta_1 < \ldots < \theta_i < \ldots < \theta_K. \tag{6.2}$$

A higher θ implies that the CI has a higher criticality level and thus it is more critical for the CC to protect it. Similar to the vulnerability levels here, we assume that the CC *does not have exact information* on the individual θ_i of CI i. Instead, the CC can only know with which probability a certain CI can belong to θ type . Therefore, we let q_{i,θ_j} be the probability with which CI i belongs to a certain type θ_j. The criticality level is used mainly to help the CC decide which CIs will be protected in case not enough resources are available for all CIs.

The values of w and θ are selected in a way that makes the resource allocation depends primarily on the vulnerability level. The criticality level affects the resource allocation but without superseding the vulnerability level, i.e., the criticality level will help the CI to get more resources than a less critical infrastructure but not more than a highly vulnerable one. Therefore, the values of θ when combined with w; should satisfy the following property:

$$\theta_K \cdot w_i \leq \theta_1 \cdot w_{i+1}, \forall i = 1, \ldots, M-1. \tag{6.3}$$

In this case, θ can be seen as a sub-type under w type in the process of allocation, although they are really independent.

To address the resource allocation problem, we explore the analogy between allocating resources to CIs and forming contractual agreements between firms and employees. We propose to use *contract theory* – a powerful framework from microeconomics [139, 165], that allows to analyze the process of creating contracts between firms and employees. Here, we note that, although some recent works [163,164] have looked at contract theory for wireless communication; however, these works do not handle the challenges of CI resource allocation.

We cast the CIP problem as a *contractual situation with asymmetric hidden information* between a firm, here being the CC, and a number of employees, here being the N CIs (or their owners). The asymmetric hidden information property stems from the fact that the CC does not know the exact vulnerability and criticality levels of every CI. To overcome this information asymmetry, the CC must properly specify a *contract* defined as a pair $(T, R(T))$ where T is the amount of resources allocated to the CI, which can be viewed as the reward/payment made by the firm to the employee and R is the reward that the CC reaps when protecting this CI. We will assume in this model that the reward is an increasing linear function in resources T that takes the form $R_i(T) = r_i T$ where r_i is determined by the vulnerability type w_i such that r_i is higher with higher w's. This implies that, for a higher vulnerability level, the CI is required to pay a higher reward than a less vulnerable one, if they take the same amount of resources. Actually, this reward function design is very important in order for the contract to be binding. By using this design, the CI that claims that has a higher vulnerability level to get more resources than it needs, will be required to pay a higher reward for the needed resources. The signing of a contract between the CC and a certain CI is thus an agreement by the CC to send certain resources to protect the CI which in return will pay a reward R to the CC.

In this system, instead of offering the same contract to all of the CIs and wasting resources, the CC will attempt to offer different contract bundles that are designed in accordance with different types of w and θ for the available CIs. For the CC, when it decides to protect a certain CI of type i, its utility function can simply be defined as the difference between reward and resources allocated multiplied by the CI type, i.e.,

$$U_{\text{CC,i}}(T_i) = \theta_i w_i (R_i(T_i) - T_i). \tag{6.4}$$

Since there are M types of CIs according to type w with probability p_{i,w_j} and K types according to θ with probability q_{i,θ_j}, the total utility of the CC can be given by:

$$U_{\text{CC}}(T) = \sum_{i \in \mathcal{N}} \Big(\sum_{k \in \mathcal{K}} q_{i,\theta_k} \cdot \theta_k \Big) \Big(\sum_{j \in \mathcal{M}} p_{i,w_j} \cdot w_j \cdot (R_j(T_i) - T_i) \Big). \tag{6.5}$$

From the CI side, the utility function of a certain CI $i \in \mathcal{N}$:

$$U_i(T_i) = \theta_i w_i V(T_i) - \beta R_i(T_i), \tag{6.6}$$

where β is a positive unit cost parameter that is less than 1 and $V(T_i)$ is the evaluation function regarding the rewards (how much does this CI value the resources allocated) which is a strictly increasing function of T that takes the form $V(T_i) = vT_i$ where v is the numerical value for the evaluation function. Here, we assume that, to reward the CC, the CI has to pay some cost, such as a negotiation or implementation cost. The contract offered by the CC needs to be feasible for the CI, i.e., it needs to be persuading for the CI to accept. To this end, next, we discuss the conditions for the feasibility of a contract in the studied model.

6.2.1 Feasibility of a Contract

To ensure that both CC and CI owners have an incentive to work together for CIP, the contracts represented by the pairs $(T_i, R_i(T_i))$ must satisfy two key properties:

1. *Individual Rationality (IR):* The contract that an infrastructure selects should guarantee that the utility of this infrastructure is nonnegative, i.e.,

$$U_i = \theta_i w_i V(T_i) - \beta R_i(T_i) \geq 0,\ i \in \mathscr{N}. \tag{6.7}$$

2. *Incentive Compatibility (IC):* Each infrastructure must always prefer the contract designed for its type, over all other contracts, i.e., $\forall i,\ j \in \mathscr{N},\ i \neq j$:

$$\theta_i w_i V(T_i) - \beta R_i(T_i) \geq \theta_i w_i V(T_j) - \beta R_j(T_j). \tag{6.8}$$

The IR constraint ensures that, when a CI signs a certain contract, the received reward must compensate the effort that the CI owner has exerted for the CC. The IC constraint allows to overcome the information asymmetry as it allows to satisfy the revelation principle [139]: A certain CI of type i will always prefer the contract $(T_i, R_i(T_i))$ that the CC designed for its type over all other possible contracts. In other words, a CI i receives the maximum utility when selecting the contract designed for its own type and, thus, this CI will have an incentive to reveal its *true vulnerability and criticality levels*. A contract is therefore said to be *feasible* if both IR and IC are satisfied. We can state the following lemma from the previous conditions:

Lemma 5. *For any feasible contract (T, R); $T_i > T_j$ if and only if $w_i > w_j$.*

Proof. We prove this lemma by using the IC constraint. we have:

$$\begin{aligned}
\theta_i w_i V(T_i) - \beta R_i(T_i) &> \theta_i w_i V(T_j) - \beta R_j(T_j),\\
\theta_j w_j V(T_j) - \beta R_j(T_j) &> \theta_j w_j V(T_i) - \beta R_i(T_i).
\end{aligned}$$

By adding the two inequalities, we get:

$$\begin{aligned}
\theta_i w_i V(T_i) + \theta_j w_j V(T_j) &> \theta_i w_i V(T_j) + \theta_j w_j V(T_i),\\
\theta_i V(T_i) - \theta_j w_j V(T_i) &> \theta_i w_i V(T_j) - \theta_j w_j V(T_j),\\
V(T_i)(\theta_i w_i - \theta_j w_j) &> V(T_j)(\theta_i w_i - \theta_j w_j).
\end{aligned}$$

□

Since $w_i > w_j$ and this implies that $\theta_i w_i > \theta_j w_j$, we obtain $V(T_i) > V(T_j)$. By definition, we know that $V(T)$ is an increasing function of T, and therefore, since $V(T_i) > V(T_j)$ we have $T_i > T_j$.

Lemma 5 simply proves that the CC must provide more resources to the CI with higher number of vulnerability points, i.e., the one that belongs to a higher w type. This essentially corroborates mathematically our intuition that more resources must be dedicated to more vulnerable CI. Using this lemma, we can state the following *monotonicity* property:

$$T_i \leq T_j \text{ if } w_i < w_j, \forall i,\ j \in \mathscr{N}. \tag{6.9}$$

Another lemma that can be derived from the IR and IC constraints pertains to the utility of the CI:

Lemma 6. *For any feasible contract $(T, R(T))$, the utility of each infrastructure must satisfy:*

$$U_i(T_i) \geq U_j(T_j) \text{ if } w_i > w_j\ , \forall i, j \in \mathscr{N}. \tag{6.10}$$

Proof. This result can be shown as follows. We know that an infrastructure which asks for more resources should provide larger rewards to the CC, i.e., if $w_i > w_j$ then $T_i > T_j$ and also $R_i > R_j$. Then, if $w_i > w_j$, we have:

$$\begin{aligned} U_i(T_i) = \theta_i w_i V(T_i) - \beta R_i(T_i) &> \theta_i w_i V(T_j) - \beta R_j(T_j) \\ &> \theta_j w_j V(T_j) - R_j(T_j) = U_j(T_j). \end{aligned}$$

□

Thus, a CI with a higher vulnerability level will receive more utility than one with a lower vulnerability level. From the IC constraint and the two shown lemmas, we can easily deduce the following. If a higher type CI selects a contract designed for a lower type, the less received resources will jeopardize this CI's utility. Moreover, if a lower type CI selects a contract intended for a higher type, the gain in terms of resources acquired cannot compensate the cost that this CI must reciprocate to the CC. A CI can thus receive its maximum utility if and only if it selects the contract that can best fit its type.

Finally two more constraints must be imposed. First, the CC should take into account that the summation of all allocated resources should be equal to the maximum resources available at the control center: $\sum_{i \in \mathscr{N}} T_i = T_{\max}$.

Second, that every CI should get sufficient amount of resources to overcome its vulnerable points. That means every type w_i should be associated with a minimum amount of resources. Therefore, each CI will have a minimum required resources according to its w type. This can be expressed as: $T_i \geq T_{i,min}$.

6.3 Optimal Contracts

In this section, we first investigate how the CC can actually find its optimal contracts. In essence, given the hidden information, the only information available at the CC is p_{i,w_j} and q_{i,θ_j}. The goal

of the CC is to design contracts that allow it to maximize the use of its resources and, thus, to maximize its utility by solving the following optimization problem:

$$\max_{T} \sum_{i \in \mathcal{N}} \Big(\sum_{k \in \mathcal{K}} q_{i,\theta_k} \cdot \theta_k \Big) \Big(\sum_{j \in \mathcal{M}} p_{i,w_j} \cdot w_j \cdot (R_j(T_i) - T_i) \Big), \tag{6.11}$$

$$\begin{aligned} \text{s.t.} \quad & \theta_i w_i V(T_i) - \beta R_i(T_i) \geq 0, \ i \in \mathcal{N}, \\ & \theta_i w_i V(T_i) - \beta R_i(T_i) \geq \theta_i w_i V(T_j) - \beta R_j(T_j), i \neq j, \\ & T_i \geq T_{i,min}, \\ & \sum_{i \in \mathcal{N}} T_i = T_{\max}. \end{aligned}$$

The problem contains a large number of constraints. For instance, the IC constraints correspond to $N(N-1)$ equations. To overcome this, next, we develop a way to relax the problem and reduce the number of constraints to get a more simple problem that could be solved. The problem can be relaxed using a technique inspired from the work in [165].

6.3.1 Relaxed Problem

The incentive compatibility must to be relaxed because for every one of the CIs we need to define $N-1$ conditions. Therefore, we will now study the local IC constraints, which are, the downward local IC (DLIC) which corresponds to the relation between CIs i and $i-1$. The other local IC is the upward local IC (ULIC) which corresponds to the relation between CIs i and $i+1$. We can now prove the following:

Theorem 5. *With the IR satisfied, the local incentive constraints*

$$\theta_i w_i V(T_i) - \beta R_i(T_i) \geq \theta_i w_i V(T_{i-1}) - \beta R_{i-1}(T_{i-1}), \tag{6.12}$$

$$\theta_i w_i V(T_i) - \beta R_i(T_i) \geq \theta_i w_i V(T_{i+1}) - \beta R_{i+1}(T_{i+1}). \tag{6.13}$$

for all $i \in \mathcal{N}$ are sufficient for global incentive compatibility.

Proof. Note (6.12) is called $DLIC(i)$ and (6.13) is called $ULIC(i)$. We begin by expressing $DLIC(i)$ and $DLIC(i-1)$ as follows:

$$\begin{aligned} & \theta_i w_i V(T_i) - \theta_i w_i V(T_{i-1}) \geq \beta(R_i(T_i) - R_{i-1}(T_{i-1})), \\ & \theta_{i-1} w_{i-1} V(T_{i-1}) - \theta_{i-1} w_{i-1} V(T_{i-2}) \geq \\ & \qquad \beta(R_{i-1}(T_{i-1}) - R_{i-2}(T_{i-2})). \end{aligned}$$

Then by adding $DLIC(i)$ and $DLIC(i-1)$ we get:

$$\theta_i w_i V(T_i) - \theta_i w_i V(T_{i-1}) + \theta_{i-1} w_{i-1} V(T_{i-1})$$

$$-\theta_{i-1}w_{i-1}V(T_{i-2}) \geq \beta(R_i(T_i) - R_{i-2}(T_{i-2})).$$

and as $\theta_{i-1}w_{i-1} \leq \theta_i w_i$ we can replace it in the previous equation to yield:

$$\theta_i w_i V(T_i) - \theta_i w_i V(T_{i-2}) \geq \beta(R_i(T_i) - R_{i-2}(T_{i-1})). \tag{6.14}$$

However, (6.14) is the IC constraint for CIs i and $i-2$ which can be written as $IC(i, i-2)$. This means that $DLIC(i)$ and $DLIC(i-1)$ imply $IC(i, i-2)$. With the same principle we can show that $IC(i, i-1)$ and $DLIC(i-2)$ imply $IC(i, i-3)$, etc. Therefore, starting at $i = N$ and proceeding inductively downward until $i = 2$, $DLIC(i)$ implies that $IC(i, j)$ holds for all $i \geq j$. A similar argument in the reverse direction establishes that $ULIC(i)$ implies $IC(i, j)$ for $i \leq j$. □

We can also reduce the IR constraints. There are a total of N IR constraints must be satisfied. Assume, without loss of generality, that the CI 1 is from type w_1. By using the IC constraints and the IR constraint of the first CI, referred to by $IR(1)$, we have:

$$\begin{aligned}\theta_i w_i V(T_i) - \beta R_i(T_i) &\geq \theta_i w_i V(T_1) - \beta R_1(T_1) \\ &\geq \theta_1 w_1 V(T_1) - R_1(T_1) \geq 0.\end{aligned} \tag{6.15}$$

Thus, if the first IR constraint of w type-1 user is satisfied, all the other IR constraints will automatically hold. Therefore, we only need to keep the first IR constraints and reduce the others.

After reducing the constraints, we have a new problem which is the same as the problem in equation (6.11) but with the new relaxed constraints in equations (6.12) and (6.15) instead of the complete IR and IC constraints.

Note that design parameters such as the reward function R, θ, w, and β should be adjusted by the CC to ensure that $IR(1)$ is satisfied.

6.3.2 Solution of the Relaxed Problem

To solve the relaxed problem, we first observe that there are now only $2N$ inequality constraints and one equality constraint. We can use Lagrangian analysis along with KKT conditions to solve the problem. The Lagrangian of the problem is:

$$\begin{aligned}L(T, \lambda, \mu) = &\sum_{i \in \mathcal{N}} \Big(\sum_{k \in \mathcal{K}} q_{i,\theta_k}\theta_k \Big) \Big(\sum_{j \in \mathcal{M}} p_{i,w_j} w_j (R_j(T_i) - T_i) \Big) \\ &+ \sum_{i=2}^{N} \mu_i \Big(\theta_i w_i V(T_i) - \theta_i w_i V(T_{i-1}) - \beta R_i(T_i) \\ &+ \beta R_{i-1}(T_{i-1}) \Big) + \mu_1(\theta_1 w_1 V(T_1) - \beta R_1(T_1))\end{aligned}$$

$$+ \sum_{i=1}^{N} \mu_{N+i}(T_i - T_{i,min}) + \lambda(T_{\max} - \sum_{i=1}^{N} T_i). \tag{6.16}$$

We need to solve this Lagrangian with KKT conditions to find all T values along with μ values and λ. The solution of this problem is not straightforward as the complexity increases with the number of CIs. Therefore, we will show the solution for only two CIs, to show that the problem has a feasible solution. For two CIs, the Lagrangian will be:

$$\begin{aligned} L(T, \lambda, \mu) = & \, p_{1,w1} w1(r_1 T_1 - T_1) + p_{1,w2} w2(r_2 T_1 - T_1) \\ & + p_{2,w1} w1(r_1 T_2 - T_2) + p_{2,w2} w2(r_2 T_2 - T_2) \\ & + \mu_2(\theta_2 w_2 v T_2 - \theta_2 w_2 v T_1 - \beta r_2 T_2 + \beta r_1 T_1) \\ & + \mu_1(\theta_1 w_1 v T_1 - \beta r_1 T_1) + \mu_3(T_1 - T_{1,min}) \\ & + \mu_4(T_2 - T_{2,min}) + \lambda(T_{\max} - T_1 - T_2). \end{aligned}$$

The KKT conditions for this Lagrangian are the relaxed problem constraints along with:

$$\begin{aligned} & p_{1,w1} w1(r_1 - 1) + p_{1,w2} w2(r_2 - 1) + \mu_1(\theta_1 w_1 v - \beta r_1) \\ & + \mu_2(\beta r_1 - \theta_2 w_2 v) + \mu_3 - \lambda = 0. \\ & p_{2,w1} w1(r_1 - 1) + p_{2,w2} w2(r_2 - 1) + \mu_2(\theta_2 w_2 v - \beta r_2) \\ & + \mu_4 - \lambda = 0. \\ & \mu_1(\theta_1 w_1 v T_1 - \beta r_1 T_1) = 0. \\ & \mu_2(\theta_2 w_2(v T_2 - v T_1) - \beta(r_2 T_2 + r_1 T_1) = 0. \\ & \mu_3(T_1 - T_{1,min} = 0. \\ & \mu_4(T_2 - T_{2,min} = 0. \\ & \mu_1, \mu_2, \mu_3, \mu_4 \geq 0. \end{aligned}$$

This problem gives only one optimal solution which is $T_1 = T_{1,min}$ and $T_2 = T_{\max} - T_1$. Actually this solution is only feasible if the following condition is satisfied $T_{1,min} + T_{2,min} \leq T_{\max}$. This implies that low vulnerability type CI will take its minimum required resources and the rest goes to the higher type CI. This result is not surprising as it is aligned with contract-theoretic results that study the contractual situation between a firm and two agencies (of two different types). [139]. For the case of more than two CIs, the lower type CI will get its lower limit and the rest of resources will be allocated to higher types according to their probabilities in a way to maximize the CC's utility.

6.3.3 Practical Implementation

Beside designing contracts, the CC needs to communicate with CIs, determine which CIs to protect, and sign contracts with them. We give the actual steps taken by the CC in this regard in

Algorithm 2: Optimized Contract implementation of CC for Resource Allocation

Input: $\mathscr{M}, \mathscr{K}, p_{i,w_j}, q_{i,\theta_j}, T_{\max}, T_{i,min}$
Output: $(T, R(T))$
1. CC declares its willingness to protect some infrastructures
2. Receive request from infrastructures willing to be protected
3.**Solve the optimal contract** for current infrastructures
if *The program has a solution, i.e, the avaialble resources are sufficient for all users* **then**
    Contracts are ready, proceed to step 4
else
    Remove the least critical infrastructure (begin with higher probability)
    return to step 3
4. **The CC Offers the contracts and waits for feedback**
if *All infrastructures accepted the offered contracts* **then**
    proceed to step 5
else
    return to step 3, for any previously excluded infrastructures
5. Sign contracts with infrastructures and allocate resources

Algorithm 2. The CC begins by having the initial information such as the set of vulnerability levels $\mathscr{M}$, the probability p_{i,w_j} of that a CI i will belong to each of the M levels, the set of criticality levels $\mathscr{K}$ and the probability q_{i,θ_j} of that a CI i will belong to each of the K levels. The CC also knows the minimum amount of resources required to protect a CI in each of the vulnerability levels as well as the total amount of available resources. The CC, hence, declares that it will offer resources to protect some CIs and begins to receive requests from CIs that are willing to be protected and designs optimal contracts for them.

Algorithm 2 shows the importance of the criticality level. When the CC is not able to protect all the CIs, it will discard some CIs depending on their criticality levels as the CC is more interested in protecting higher critical CIs. This is done by removing the least critical infrastructures from the process of designing contracts. However, as the CC only knows the probabilities of criticality levels, it will remove the one that belongs to the lower criticality level with a higher probability. The CC repeats this process until there is enough resources for the rest of CIs. When CIs receive contracts, they will evaluate them and inform the CC whether they are willing to accept a contract, i.e., receive resources and return reward. If not all CIs accept a contract, the CC will reconsider any CIs that were excluded due to lack of resources. After this process is finished, the CC will sign contracts with CIs and allocate resources.

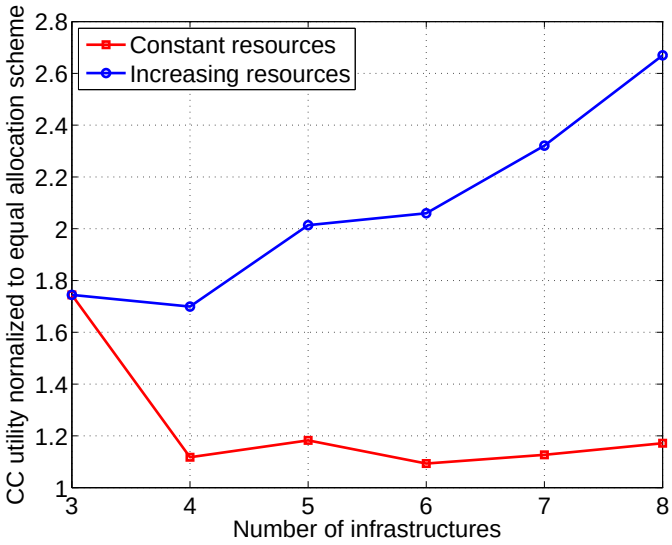


Figure 6.1: The CC utility in the case of using the proposed contract and the case of equal resource allocation when fixing $T_{\max}$ and when increasing $T_{\max}$ by 30% with every added infrastructure.

6.4 Simulation Results and Analysis

Simulations are used to evaluate the designed mechanism. For our simulations, we choose 3 vulnerability levels and 4 criticality levels. The number of available resources is set to 500. The reward function increases by 3 for each w type. The evaluation function was assumed to be two times the resources. The lower bounds associated with w types are set to 20, 60, and 100 respectively. First, we check the feasibility of our contract. We assume that all CIs ask for protection and all of them accept contracts offered by the CC. We calculate the CC's utility in case of using the proposed mechanism and in case of allocating resources equally between CIs.

In Figure 6.1, we show the variation in the CC utility as the number of infrastructures increases; the utility is normalized to the case of equal resource allocation. The figure studies two cases: fixing the amount of resources for all CIs and increasing the amount of resources each time a CI is added. When the number of resources is fixed to 500, in Figure 6.1, we can see that, with 3 CIs there is about 75% increase in the CC utility, relative to equal allocation. When more CIs are added, the percentage increase in CC utility is between 10% and 20%. This is due to the fact that, when the number of CIs is small, the CC has more resources than needed and, thus, it will give them to higher types and to get higher rewards for the same resources. In the second case, the amount of resources increases by 30% of the original amount each time we add a new CI. The CC utility in this case keeps increasing as the CC allocates the more available resources to higher types to get higher rewards.

Next, we add a new vulnerability level with associated lower bound of 140 and we increase the number of available resources to 650. We have 4 CIs, they are assumed to be within different w

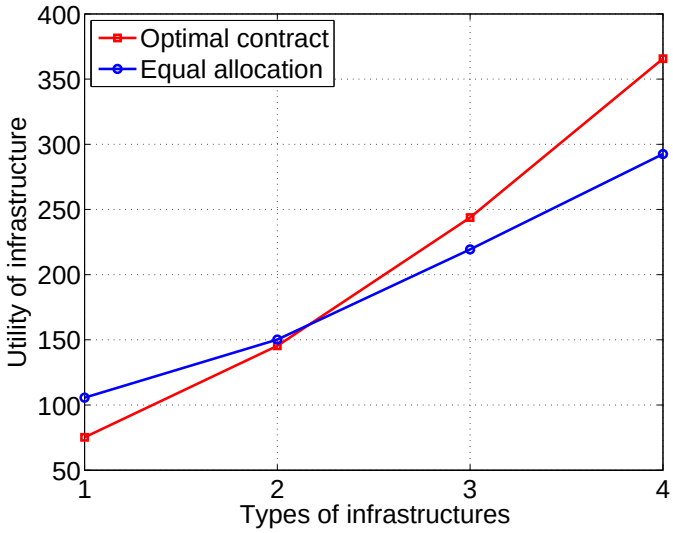


Figure 6.2: Infrastructures' utilities in the case of using the proposed contract and the case of equal resource allocation.

types in ascending order, i.e. CI 1 is within w_1 and so on. However, the CC still knows their probabilities not their actual types. Figure 6.2 shows the utility of CIs in case of using optimal contracts and in case of equal resource allocation. The figure proves the monotonicity property of the proposed contract as higher types CIs get higher utilities. We can also see in Figure 6.2 that, in case of optimal contracts, higher types CIs get higher utilities and lower types CIs get lower utilities compared to equal resource allocation. However, these lower utilities is not a problem as these CIs get much more resources than needed for protection.

In Figure 6.3, while maintaining the parameters of Figure 6.2, we show the utility of the infrastructure as the contract type varies. Here, we measure the utility of each CI if it used the contract designed for its type and contracts designed for other types. In Figure 6.3, we can clearly see that it is better for every CI to use the contract designed for its type as this maximizes its utility. Actually, CIs can get more resources from choosing higher types contracts but will be required to pay higher rewards which is reflected in decreasing their utility.

6.5 Summary

In this chapter, we have studied the problem of resource allocation for protecting CIs. We have formulated the problem using a contract-theoretic model in which a CC offers contracts to a number of CIs and each one selects its best contract. For each CI, we have defined two different types that correspond to the vulnerability and the criticality levels. In the model, the CC does not know these exact levels but only knows with which probability a CI will belong to a certain level. We

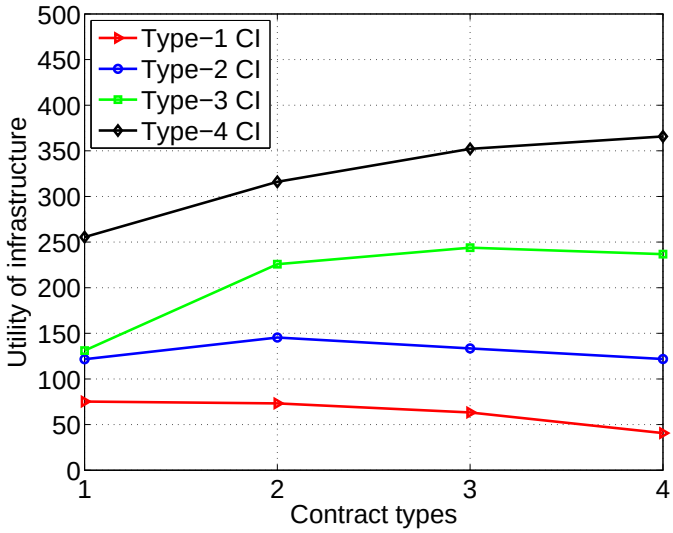


Figure 6.3: The utility of each infrastructure while accepting the contract designed for his type or other contracts.

have provided the necessary and sufficient conditions for such resource allocation contracts under asymmetric information. The problem was then relaxed and solved to show that it has an optimal solution and motivates each CI to accept the contract designed for each type. Simulation results have shown that this model helps the CC to get higher utility than the case of equal resource allocation. In addition, our results show that each CI will not gain from selecting other contracts as its utility will not increase.

Chapter 7

Towards Resilient Transportation Networks against Flooding

7.1 Background and Related Works

Transportation systems are prone to natural disasters such as earthquakes and flooding. Several approaches studied the problem of restoring the transportation networks after natural disasters, especially earthquakes, such that minimizing the restoration time and/or cost [166–168]. As earthquakes can cause physical damage to roads or bridges, restoration techniques usually focus on restoring specific roads/bridges, under a limited budget, to achieve the most traffic flow in the least restoration time. On the other hand, the problem of studying the transportation network in case of flooding did not get much attention in literature. This is because, the effect of flooding in most cases is temporary and the problem affects mostly coastal cities not any city like earthquakes. Coastal cities became more prone to flooding due to the sea level rise in the past few years caused by global warming [169]. According to [169], more than ninety coastal communities in the United States are battling unmanageable flooding that threatens people existence in these communities. For example, in Miami Beach city, the number of flooding occurrences due to rain has increased by 33% in the past decade and the number of occurrences due to tide has increased by 400% in the same period [170]. This shows how the problem became chronic in coastal cities.

Recently [171, 172] studied the effect of flooding on traffic flow. The authors in [171], proposed to direct drivers to use alternative routes based on the expected flood severity. The model is mainly empirical that directs drivers away from roads that are high likely to have low traveling speeds due to flooding. In [172], introduced the integration of flooding models into traffic simulators to measure the effect of flooding on planed trips that need to be canceled or rerouted.

In this work, we propose to develop an analytical framework to study and improve the resilience of transportation systems against flooding in coastal cities. As a first step, we will study how to improve the total system's travel time for drivers in case of flooding by partially or totally shifting

lanes between roads direction. Road shifting is typically handled by a system manager representing the authorities of a city. The system manager plans the shifting procedure by the excessive studying of the transportation network. Here, we propose to study the transportation network from a macroscopic view as it is more suitable for planning purposes. One of the most widely approaches, in studying the transportation networks, is studying the traffic under equilibrium, also known as Wardrop equilibrium [173]. Wardrop equilibrium is considered as a game-theoretic equilibrium where no traveler is welling to unilaterally change its route, and in this case, the travel time will be equal for all used alternative routes between any origin and destination in the network. We propose to calculate the travel time after flooding which increases based on the flooding severity and roads' preparedness. A bi-level problem will then be introduced to optimize the problem of lane shifting to minimize the traffic flow under flooding while maintaining a fixed budget.

7.2 System Model

We consider a transportation network modeled using a directed graph $G = (\mathscr{V}, \mathscr{E})$, where $\mathscr{E}$ represents the directed edges or roads and $\mathscr{V}$ represents the intersection points which can be sources, destinations, or intermediate points. Bi-directional roads are modeled as two different edges, an edge in each direction. We will refer to edges as links in the following.

In this network, the flow-based travel time function for a link a is given by:

$$t_a = t_{a,0} \cdot (1 + \alpha \cdot (\frac{x_a}{C_a})^{\beta}), \tag{7.1}$$

where $t_{a,0}$ is the free flow travel time for link a determined by the maximum speed allowed on link a, x_a is the amount of flow on the link, C_a is the capacity of the link determined by the road condition and its number of lanes. α and β are two parameters that are typically set to 0.15 and 4 respectively [174].

In case of flooding, water accumulates in the roads making it harder for cars to move with their regular speeds. Previous studies have shown that both the free flow travel time and the road capacity decrease in case of water accumulation [175]. The authors in [175] showed that the reduction in the free-flow speed ranges from 2% to 9% based on the rain intensity, while the reduction in the capacity remains constant and around 10% irregardless of the rain intensity. However, as rain intensity does not fully capture the flooding effect on roads, a direct relation between accumulated water depth and free-flow speed is required. Here, we adopt the relation in [1] which represents the maximum acceptable vehicle velocity that ensures safe control as follows:

$$v(w) = 0.0009w^2 - 0.5529w + 86.9448, \tag{7.2}$$

where v is the maximum safe velocity and w is the water depth. This function is derived from data collected from experimental, observational, and modeling studies as shown in Figure7.1 [1]. The function is derived using curve fitting from all the available data which is believed to provide an accurate relation between the free-flow speed and water depth.

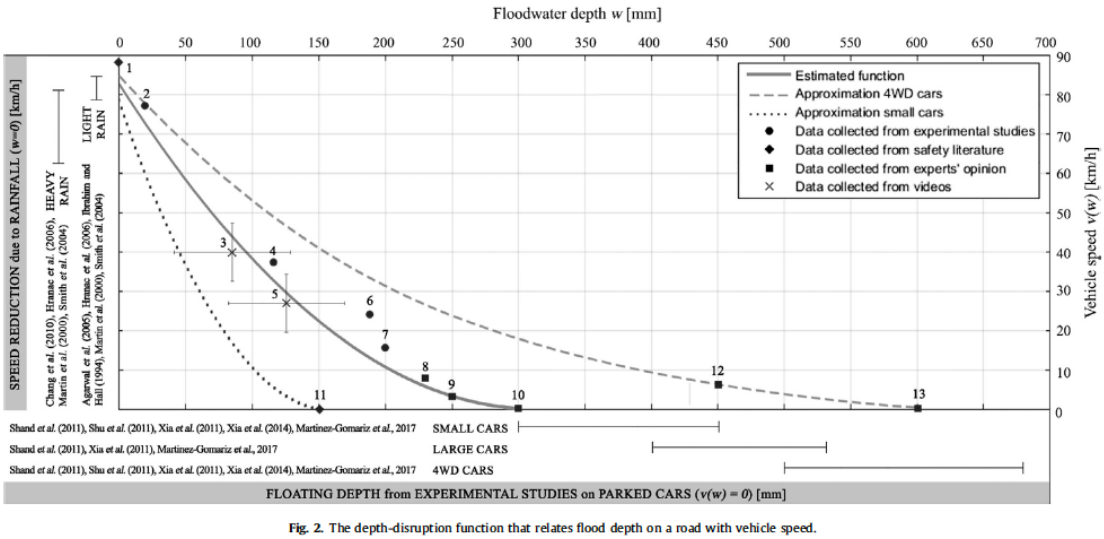


Fig. 2. The depth-disruption function that relates flood depth on a road with vehicle speed.

Figure 7.1: Relation between rain intensity and vehicles speed [1]

Moreover, as roads differ in their preparedness to flooding, e.g., road level, installed pumps, and storm drains, floods will have different effects on roads. We model this difference as the effective flood depth calculated from the actual flood depth as follows:

$$w_f = (1 - P) \cdot w, \tag{7.3}$$

where w is the actual flood depth from e.g., rain or tides and P is the road preparedness which depends on how the road (link) is prepared by drainage systems or pumps to withstand flooding. P works as a reduction factor for the flood depth, where the value $P = 0$ means the road does not have any drainage systems and, hence, the value of the effective flood depth will equal the value of the actual flood depth. On the other hand, the value $P = 1$ means the drainage system in the road is effective in removing all the water, and, hence, the effective flood depth will equal zero, i.e., normal driving conditions will apply.

Flow demands are given between certain origin-destination (O-D) pairs in the network. We consider all the possible paths between every (O-D) pair. At equilibrium, all different used paths between any (O-D) pair, should have the same travel time according to Wardrop equilibrium. Wardrop equilibrium is calculated as follows for the network:

$$\begin{aligned} min \;\; & Z(x) = \sum_a \int_0^{x_a} t_a(w) dw \\ \text{s.t.} \quad & \sum_k f_k^{rs} = q_{rs}, \quad \forall r, s \\ & f_k^{rs} \geq 0 \quad \forall k, r, s \\ \text{where} \quad & x_a = \sum_r \sum_s \sum_k f_k^{rs} \delta_{a,k}^{rs} \quad \forall a, \end{aligned} \tag{7.4}$$

where f_k^{rs} is the flow on path k that belongs to the (O-D) pair (r, s), q_{rs} is the total demand between (O-D) pair (r, s). The first constraint is called the flow reservation rule as it ensures that the amount of flow on all paths between any (O-D) pair equals the total flow demand between this pair. The second constraint, is the non-negativity as there is no negative flow. Finally, $\delta_{a,k}^{rs}$ is an indicator that equals 1 if the link a is part of the path k between the (0-D) pair (r, s), and 0 otherwise.

The solution to (7.4), gives the equilibrium flow assignment for each path int the network. The amount of flow on each link is the summation of the flow of all the paths this link is part of. The travel time of each link can then be calculated from (7.2) by substituting the optimal flow assignment. Finally, the total system's travel time is the time between each (O-D) pair multiplied by the total flow on this path.

We model the problem as a bi-level optimization. In the upper level, the system manager optimizes the road shift based on the cost in the traffic assignment problem as follows:

$$\begin{aligned} \min \ Z(\pi) &= \sum_a x_a t_a(x_a, \pi_a) \\ \text{s.t.} \quad & \sum_a c_a(\pi_a) \leq B, \end{aligned} \tag{7.5}$$

where π_a is the capacity shift in link a, $c_a(\pi_a)$ is the shift cost in link a, and B is the total budget.

This solution to the upper-level problem gives the optimal change in links direction that can achieve the minimum possible travel time. The proposed approach works as follows: links that share the same road but opposite directions are coupled together. A capacity shift can occur between coupled links which is either full or partial shift. In full shift, both links are assumed to have the same direction, which means the capacity of one link is transferred to the other link. This is proposed to occur in practical situations by declaring any road as a one-way road in times of flooding. In partial shift, a fixed number of lanes from one direction are assumed to serve as the opposite direction. This means partial capacity from one direction is transferred to the other direction. In practical, this can occur by using temporary separators between lanes and signals to indicate the change.

In the lower level, the problem in (7.4) is solved for the updated free-flow travel time as in (7.1). The solution procedure starts by solving the lower level to calculate the optimal traffic flow, which is then used in the upper level to calculate the capacity shift vector. The capacity shift vector is plugged in the lower-level again to calculate the updated flow. The process repeats until the solution of the upper and lower levels converge.

7.3 Preliminary Results

We applied the proposed framework to the network shown in Figure 7.2. This network represent a small scale transportation just for the sake of applying our mechanism. However, the mechanism applies to large-scale networks as well. The Link capacities are shown in Figure 7.2. There are two

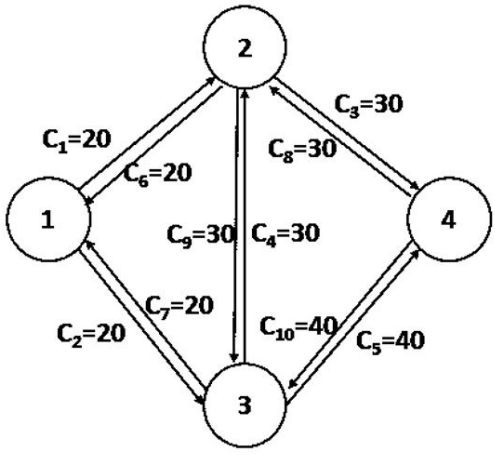


Figure 7.2: A sample transportation network to test the proposed framework.

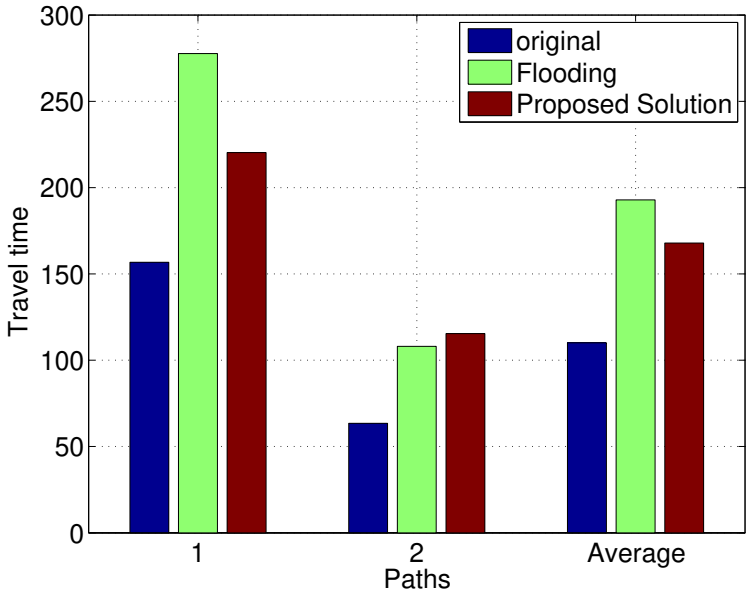


Figure 7.3: Total System's travel time.

(O-D) pairs $(1, 4)$ and $(3, 4)$ with the demand values 80 and 60 respectively. Free-flow link travel times are assumed to be $30, 20, 20, 30$, and 30 for links 1 to 5. Same values, in order are given to links from 6 to 10. The solution of the proposed framework is to shift the whole capacity from link 6 to link 1.

The original total system's travel times, the travel times after flooding, and after applying the mechanism are shown in in Figure 7.3, for both travel paths. We can notice that there is an increase in the system's travel time on path 2, meanwhile, there is a significant decrease in path 1 travel time. The average of the two paths is also shown in Figure 7.3, where we can see that there is a

significant reduction in the system's travel times due to applying the defense mechanism.

Chapter 8

Environment-Aware Deployment of Wireless Drones Base Stations with Google Earth Simulator

8.1 Background, Related Works, and Contributions

As we discussed in Chapter 1, unmanned aerial vehicles (UAVs), or drones have recently attracted significant attention as a promising approach to enhance wireless communication performance . When equipped with wireless base station hardware, drones can supplement the coverage provided by existing cellular infrastructure. The mobility of drones facilitates the creation of line-of-sight (LoS) links with users, ensuring optimal connection strength. This ability, coupled with the reliability and autonomy of drones, lends UAVs attractive qualities to service providers. In particular, UAVs are an effective approach in emergency scenarios such as disaster relief, when unplanned power outages may compound with the increased need for communication, and Internet of Things (IoT) applications [176], where the quantity and low transmit power of devices may necessitate closer-ranged wireless communications. Meanwhile, UAVs can also be used to complement existing terrestrial cellular systems by bringing additional capacity to crowded areas during temporary events. Furthermore, drones can be deployed to provide necessary wireless connectivity to rural areas in which the presence of large-scale ground wireless infrastructure is limited.

Simulation is an important challenge in network research, offering a compromise between precision and speed. Many wireless network simulators currently exist (e.g., see [177] and [178] and references therein). Among these, only some are suited specifically for the analysis of three-dimensional, aerial ad hoc networks [179]. These are typically implemented as extensions of the general network simulators [180], that operate in two dimensions. UAV-enabled networks are highly dynamic and thus require a proper integration of the movement of UAVs into the simulation environment. Moreover, analysis of these networks is made more challenging by the uncertainty of

environmental variables affecting propagation, as well as highly dynamic interference. To account for these UAV features, many models implement probabilistic expressions based on environment type, i.e. rural, urban, or dense urban [181]. Thus, the ability to identify obstacles by processing satellite images has immense value in that simulation can become more deterministic, in proportion to accuracy of image processing.

While there has be a notable number of works on UAV deployment, the do not consider the potential use of real geographical information for optimal placement of UAVs. For instance, the work in [182] optimizes the altitude of a single UAV for maximizing coverage based on a probabilistic path loss model. Using this model, the authors in [183] studied the coverage maximization problem with minimum number of drone base stations. In [184], the deployment of an aerial UAV base station for maximizing sum-rate and power gain in a wireless network is studied. Unlike the prior studies on the deployment of drones, we will exploit geographical information such as obstacles to effectively deploy and operate drone-enabled wireless systems.

We chose to implement a simulator in the Google Earth Engine platform because of its readily available datasets, its image processing tools, and its association with the well-known Google Earth program. Earth Engine is similar to Google Earth in that users can explore satellite imagery through an intuitive interface, though Earth Engine is especially suited for analysis and representation of geospatial data. The simplest way to use Earth Engine is through its built-in JavaScript IDE, which we use in this project; Python is also supported through an API. The platform is well-suited for our application because of the image processing potential, allowing us to estimate and refine network parameters, and the intuitive interface through which users can interact with outputs. More here about our work. In addition to the datasets considered in this application, Earth Engine integrates several datasets that are useful for researchers and network planners, including population density estimates and atmospheric data.

The main contribution of this chapter is a novel simulation framework for environment-aware deployment of multiple drone bases stations that provide wireless connectivity for ground users. In particular, by exploiting geographical information extracted from the Google Earth Engine, we investigate three key UAV deployment scenarios. First, we study the optimal placement of drones for maximizing the number of covered ground users. In the second scenario, we aim to provide full coverage for ground users by using a minimum number of drones. Finally, given the load requirements of users, we analyze the optimal deployment of drones for which the total flight time of drones needed to service the users is minimized. Simulation results reveal that our proposed framework yields a significant improvement in the coverage and energy efficiency of the drone-enabled wireless networks. Moreover, our results show the existence of an optimal number of drones that maximizes the wireless connectivity.

The rest of this chapter is organized as follows. In Section II, we present the system model the drone deployment scenarios. In Section III, we describe the feature (i.e, obstacle) extraction method from Google Earth. Simulation results are presented in Section IV and conclusions are drawn in Section V.

8.2 System Model and Drone Deployment Scenarios

Consider a set $\mathscr{L}$ of L single-antenna wireless users located within a given geographical area. The location of a user $i \in \mathscr{L}$ is given by (x_i, y_i). In this area, a set $\mathscr{M}$ of M quadrotor drones are used as flying base stations to provide downlink wireless service to ground users, as shown in Figure 8.1. The location of a drone $j \in \mathscr{M}$ is given by $\boldsymbol{v}_j = (x_j^D, y_j^D, h_j)$.

Each user i can be served by one drone j that provides the strongest downlink signal-to-interference-plus-noise-ratio (SINR) for the user such that $\gamma_{ij} = \arg\max_{j \in \mathscr{M}} \gamma_{ij}$ and $\gamma_{ij} \geq \gamma_{\text{th}}$ where γ_{ij} is the SINR downlink between user i and drone j and γ_{th} is threshold SINR required by the user to successfully have wireless service. Here, the SINR for user i that connects to drone j can be given by:

$$\gamma_{ij} = \frac{\eta P_j d_{ij}^{-\alpha}}{\sum\limits_{u \in \mathscr{I}_{\text{int}}} \eta P_u d_u^{-\alpha} + \sigma^2}, \tag{8.1}$$

$$d_{ij} = \sqrt{(x_i - x_j^D)^2 + (y_i - y_j^D)^2 + h_j^2}, \tag{8.2}$$

where α is the path loss exponent, σ^2 is the noise power, η is the path loss constant. d_{ij} is the distance of drone-BS j with a user i. Also, $\mathscr{I}_{\text{int}}$ is the set of interfering drone-BSs.

We assume that users have fixed locations and that drones can move to certain locations to service the users. Our goal is to optimally deploy the drones, i.e., calculate optimal locations to provide the wireless service in each of the following scenarios.

8.2.1 Maximizing the Number of Covered Users

In the first scenario, our goal is to maximize the number of covered users under limited resources (available drones). This scenario captures emergency scenarios, e.g., flooding or power outage, or highly unusual wireless service demand, e.g., a fair or a sports event in a stadium. In such cases, the goal of using drones is to provide wireless service to the largest possible number of users. Covering every user in these cases might not be possible due to high data demand that will require more drones than what is available. Determining the number of drones that can be used in these scenarios depends on the number of available drones and the expected coverage in this geographical area. In emergency cases for example, when more than one geographical area is affected and needs urgent coverage, drones are to be deployed in these areas according to the percentage of ground users coverage that can be achieved by drones.

In this scenario, the number of users is fixed to L and the number of drones is fixed to M. The goal is to find the optimal locations of the drones $\boldsymbol{v}_j$, $\forall j \in \mathscr{M}$ to maximize the number of covered users. Let $\mathbb{1}_{ij}$ be an indicator of whether or not user i is connected to drone j such that:

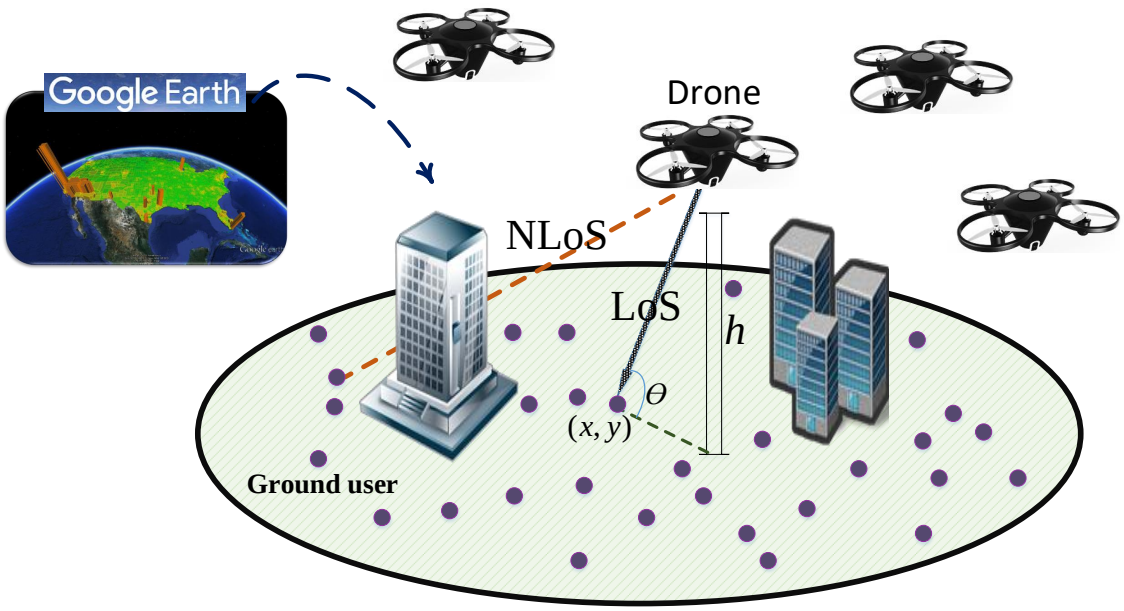


Figure 8.1: System model for drones' deployment.

$$\mathbb{1}_{ij} = \begin{cases} 1 & \text{if } j = \arg\max_{j \in \mathscr{M}} \gamma_{ij} \text{ and } \gamma_{ij} \geq \gamma_{\text{th}}, \\ 0 & \text{if otherwise.} \end{cases} \tag{8.3}$$

The problem can then be formulated as:

$$\max_{\mathscr{L}} \sum_{i \in \mathscr{L}} \sum_{j \in \mathscr{M}} \mathbb{1}_{ij} \tag{8.4}$$

$$\text{s. t.} \quad \sum_{j \in \mathscr{M}} \mathbb{1}_{ij} = 1, \forall i \in \mathscr{L}. \tag{8.5}$$

The constraint in (8.5) guarantees that every user is connected to only one drone.

8.2.2 Full Coverage with a Minimum Number of Drones

In this next scenario, every user needs to be covered using the minimum number of drones. Here, unlike the previous scenario, we do not assume limited resources. This scenario comes out usually in public safety and pre-disaster awareness where every user needs to be informed by a disaster mitigation plan. For example, in pre-disaster evacuation, we need to make sure that every user is aware of the upcoming danger in a timely-manner. This can help increase the community resilience against these type of disasters. Covering every user (i.e., full coverage) can be achieved by

deploying drones in the targeted geographical area. However, as deploying these drones is usually costly, we need to ensure full coverage while minimizing the number of drones, and, hence the cost.

The goal is to calculate the minimum number of drones required to achieve full user coverage to the L available users. This is achieved by calculating the optimal locations of the drones $\boldsymbol{v}_j,\ \forall j \in \mathscr{M}$ to achieve full coverage of the users. We use the same indicator $\mathbb{1}_{ij}$ as defined in the previous scenario. The problem can then be formulated as:

$$\min_{\mathscr{M}} \sum_{j\in\mathscr{M}} \sum_{i\in\mathscr{L}} \mathbb{1}_{ij} \tag{8.6}$$

$$\text{s. t.} \quad \sum_{j\in\mathscr{M}} \mathbb{1}_{ij} = 1, \forall i \in \mathscr{L}, \tag{8.7}$$

$$\sum_{i\in\mathscr{L}} \mathbb{1}_{ij} = L. \tag{8.8}$$

The first constraint ensures that every user is connected to only one drone and the second constraint ensures that all the users are connected to drones.

8.2.3 Minimizing Flight time of Drones in Serving Users

In this third scenario, each user needs to download some data using the wireless service and we are interested in minimizing the hover time (service time) of the drones to satisfy this data load for every user. This scenario captures the case in which the consumed energy is of importance as the drones can only serve for limited time [185]. For example, the case in which the drones are to be deployed in a geographical area that is far from their source and drones will consume a significant percentage of their energy in their traveling to the destination. The remaining amount of energy (that will be used to serve the users) needs to be used in the most effective way possible so as to satisfy the demand of the users.

Each user, among the L users, is assumed to have a load of data given by β_i bits that needs to be satisfied. A drone j can transmit data to a user i with a rate b_{ij} bits/second that depends on γ_{ij}. The time spent by a drone j to serve a user i can then be calculated as:

$$t_{ij} = \frac{\beta_i}{b_{ij}}. \tag{8.9}$$

The total hover time of a drone j can then be calculated as the summation of the times spent to serve all the users connected to this drone. Let $\mathscr{N}_j$ be the set of all users connected to drone $j,\ \forall j \in \mathscr{M}$. Then, the hover time for a drone $j \in \mathscr{M}$ will be is given by:

$$t_j = \sum_{i\in\mathscr{N}_j} \frac{\beta_i}{b_{ij}}. \tag{8.10}$$

The goal in this third scenario is to find the optimal locations of the drones to minimize the overall hover time of all drones given that the load of each user needs to be satisfied. The problem can be formulated as:

$$\min_{\mathscr{M}} \sum_{j \in \mathscr{M}} \sum_{i \in \mathscr{N}_j} \frac{\beta_i}{b_{ij}} \tag{8.11}$$

$$\text{s. t.} \quad \sum_{j \in \mathscr{M}} \mathbb{1}_{ij} = 1, \forall i \in \mathscr{L} \tag{8.12}$$

$$\sum_{i \in \mathscr{L}} \mathbb{1}_{ij} = L. \tag{8.13}$$

The constraints are similar to the previous scenario. In this scenario, when every user is connected to a drone, then every user will be in a set $\mathscr{N}_j$ of a specific drone j such that:

$$\bigcup_{j \in \mathscr{M}} \mathscr{N}_j = \mathscr{L}. \tag{8.14}$$

Then, the problem formulation of (8.11) will minimize the overall hover time while ensuring that the total load of users is satisfied.

8.3 Exploiting Earth Engine to Determine Locations of Obstacles

Various building detection algorithms have been developed, with cited precision ranging from 80-90% [186–188]. Accurate algorithms rely on a combination of feature extraction techniques and machine learning. For our application, we circumvent the time and resources needed to train such programs by exploiting the "map view" imagery supplied by Google. In this view, satellite imagery is simplified, wherein features like buildings are identified in the same color. This greatly facilitates automated building identification, under the assumption that Google's own identification techniques are accurate.

To extract building locations from map view, we use edge detection. This is implemented most readily in Earth Engine through Canny edge detection, a reliable and very common algorithm [189, 190]. Canny detection applies separate filters to detect horizontal, vertical, and diagonal edges, and computes the gradient magnitude. Finally, non-maximum magnitudes are suppressed, thinning the detected edges. In general applications of edge detection, image noise must be accounted for through the application of Gaussian filters; even then, error is expected. However, the simple, noiseless images provided by map view are ideal candidates for edge detection, and edge detection yields accurate results.

To extract lines from this output, we apply the Hough transform to the Canny image [189]. This step is important to correct imperfections in the Canny output. The Hough transform uses an accumulator to detect the presence of a line, then implements a voting algorithm to identify its parameters. Now, we sample and trace each line, noting changes in direction which correspond to building corners. At this point, we can also manually adjust the locations of any vertex. To examine the accuracy of this process we outlined buildings on map view and overlaid them onto the corresponding satellite imagery, shown in Figure 8.2.

Evidently, through this method, buildings are approximated fairly well. Over five test cases, this process correctly outlined about 95% of each building's correct area, and falsely identified an additional 12%. These figures are consistent with the 80-90% accuracy bounds given in the studies cited above. Additionally, we note that this method tends to overestimate building area. This is permissible, and possibly preferable, for UAV simulation, in which drones should not be deployed within a buffer area around buildings, due to the threat of collision. Limitations of geometric approximation of buildings in this manner include irregular building shapes, specifically ones with rounded sides. Earth Engine only supports polygons; thus, rounded edges must be represented by some number of vertices, adding inherent error. Thus, we have shown that for building location identification, analysis of Google map data is consistent in accuracy with rigorous processing of satellite imagery, but can be performed at reduced computational cost.

8.4 Simulation Results

For our simulations, we consider a 200 m $\times$ 200 m area over which users are randomly distributed. Users are assumed to be at ground level, at which $z = 0$. The locations of buildings are known, defined by their vertices at $\{V_1, V_2, ..., V_N\}$, where each vertex consists of an x and y coordinates. For these simulations, we consider a three-building configuration derived from Figure 2. As we did not estimate building height during image processing, we model the buildings' z-coordinates as random variables, constrained between 10 and 20 meters, heights appropriate for five-story buildings. Other simulation parameters are listed in Table 1.

To evaluate any arrangement of M drones over N_C candidate points, $\binom{N_C}{M}$ calculations are required. As N_C correlates directly with simulation precision, and hence a large N_C is desirable, the computational complexity can quickly become infeasible. To circumvent this, the following heuristic is implemented. We first discretize the target area into some number NC of UAV candidate points, where NC is sufficiently small to enable rapid evaluation. We form the binary power threshold matrix $\boldsymbol{T}$ in which entry (m, n) indicates whether the user at location (xn, yn) receives above a given power P^t_{min} from candidate point m. Note that we do not yet account for interference, noise, or line-of-sight; our current goal is to establish starting points for further optimization. We incrementally place drones at the candidate points is maximized; in other words, at points with the most potential links.

Now, we further discretize the area around each chosen candidate point. Given $\{V_1, V_2, ..., V_N\}$,

Figure 8.2: Results of building identification imposed over satellite imagery.

we calculate whether LoS exists by sampling the line segment connecting user i and each candidate point and checking whether any sample point lies within the bounds of a building. If so, we introduce an additional attenuation factor, η, to that potential channel. Finally, we consider interference and noise, and simultaneously solve for the optimal locations of each UAV such that number of users above a given SINR threshold, γ, is maximized.

Figure 8.3 shows the percentage of covered users as the SINR threshold needed for connectivity varies (this result corresponds to Scenario A deployment). Clearly, as the SINR threshold or equivalently the receivers' sensitivity increases, the coverage performance of drones decreases. This due to the fact that satisfying a higher SINR requirement is more challenging thus fewer number of users can be covered by the drones. For instance, when increasing the SINR threshold from 2 dBm to 8 dBm, the number of covered users decreases by 63% in the proposed approach. In Figure 8.3, we also compare the performance of the proposed deployment approach with a case in which deployment is done based a probabilistic path loss model. In the probabilistic model, a drone can have a LoS link to a ground user with a specific probability, which is given by [191]:

$$P_{\text{LoS},i} = b_1 \left(\frac{180}{\pi} \theta_i - 15 \right)^{b_2}, \tag{8.15}$$

where θ_i is the elevation angle (in radians) between the drone i and a user located at (x, y). Also,

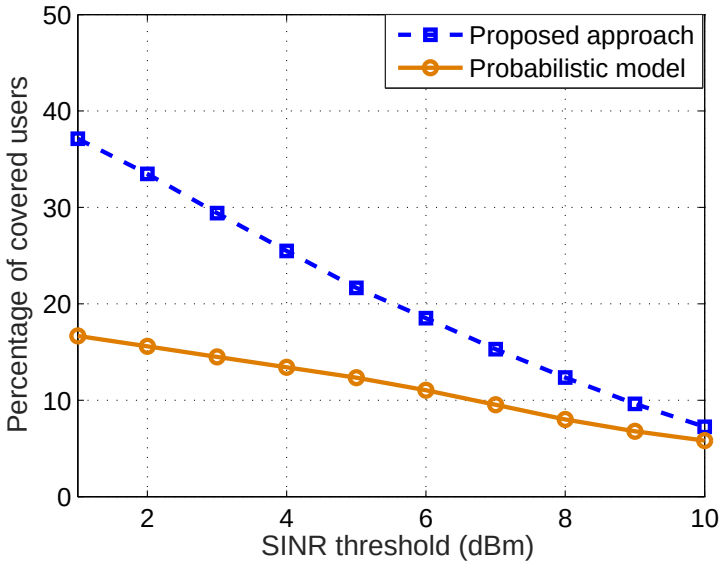


Figure 8.3: Percentage of covered users versus SINR threshold.

b_1 and b_2 are constant values which depend on the environment.

As we can see from Figure 8.3, our approach outperforms the probabilistic case. In our approach, the locations of buildings are known and deterministic as they are obtained from Google Earth Engine. In the probabilistic case, however, we do not have a complete information about the buildings. Therefore, by exploiting additional information about the environment, our deployment approach leads to a higher coverage performance than the probabilistic-based deployment. As shown in Figure 8.3, the number of covered users can be increased by up to a factor of 2 while adopting the environment-aware deployment strategy. As an illustrative example, in Figure 8.4, we show visual output of drone placement, using known building locations.

Figure 8.5 shows the impact of the number of drones on the coverage performance for various number of users (this result corresponds to Scenario B deployment). Clearly, the coverage performance decreases as the number ground users increases. For a higher number of users, it will be more likely that drone-users communication links become blocked by obstacles. Consequently, the communication reliability and, hence, the coverage performance degrades. Figure 8.5 also shows that how the number of covered users varies by changing the number of drones. In this case, there is a tradeoff in deploying more drone base stations for providing wireless connectivity. By increasing the number of drones, the coverage can be improved as the drones are placed closer the ground users. However, while using more drones, the aggregated interference increases which reduces SINR for the users. Therefore, there can be an optimal number of drones for which the coverage is maximized. For instance, as we can see from Figure 8.5, the optimal number of drones for serving 100 users is 6. This figure allow us to determine the minimum number of drones needed to meet a certain coverage requirement. For example, here, a full coverage for 50 users can be achieved by

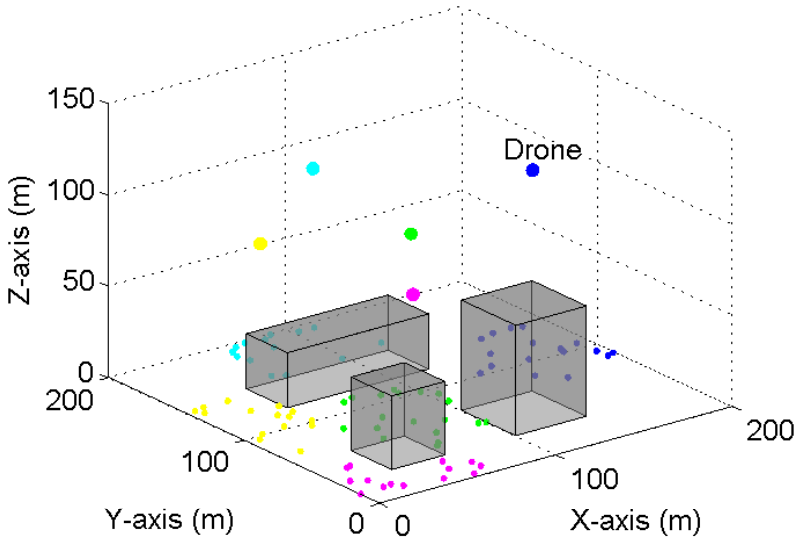


Figure 8.4: An illustrative figure for drones' deployment.

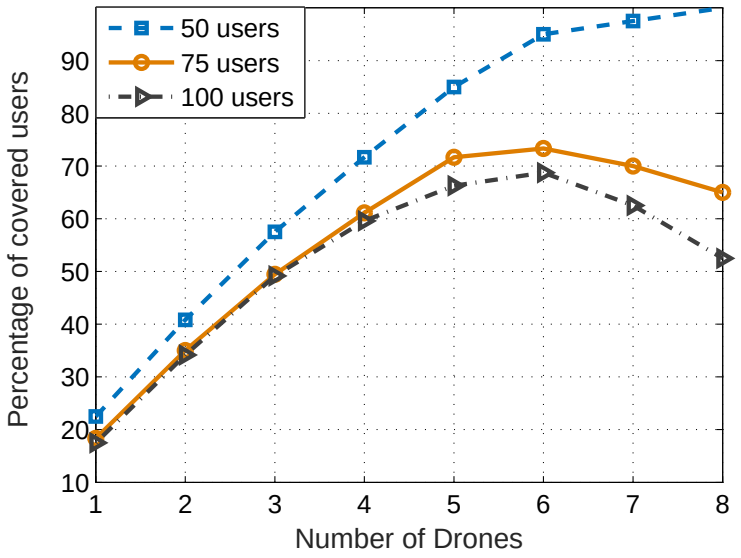


Figure 8.5: Percentage of covered users versus number of drones.

optimally deploying 8 drones over the considered geographical area.

Figure 8.6 shows the total flight time of drones needed for completely servicing the users (this result corresponds to Scenario C deployment). From this figure, we can see that the flight time of drones increases by when the number of buildings (i.e., obstacles) increases. With more obstacles in the environment, drone-to-user communications will experience lower SINR due to the blockage

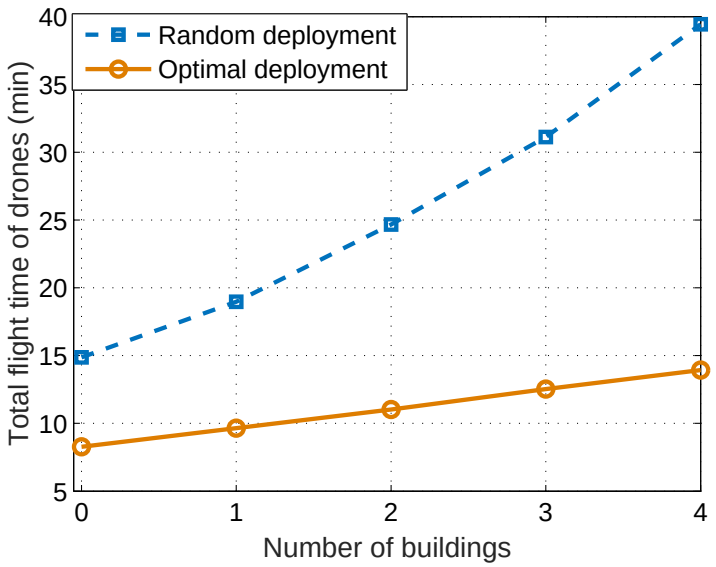


Figure 8.6: Total flight time of drones versus number of buildings (i.e., obstacles).

Table 8.1: Simulation parameters.

Parameter	**Description**	**Value**
f_c	Carrier frequency	2 GHz
P_i	Drone transmit power	1 W
N_o	Total noise power spectral density	-170 dBm/Hz
N	Number of ground users	200
B	Bandwidth	1 MHz
b_1, b_2	Parameters in probabilistic channel model	$0.36, 0.21$ [182]
β	Load per ground user	10 MHz

and shadowing effects. As a result, the transmission rate will decreases and the drones must fly longer in order to transmit a required amount of data to each user. From Figure 8.6, it can be seen that the total flight time of drones increases by 45%, in the proposed deployment case, when the number of buildings increases from 1 to 4. Hence, servicing users located in a harsh environment requires longer flight time, more energy consumption, and thus using more capable drones.

Here, we compare the performance of our proposed environment-aware deployment approach with a random deployment case in which drones are randomly deployed over the geographical area. As we can see from Figure 8.6, the proposed optimal deployment can yield up to a 65% flight time reduction compared to the random deployment case. Therefore, the proposed approach enhances energy-efficiency of the considered drone-enabled wireless network.

8.5 Summary

In this chapter we have investigated environment-aware deployment of drone base stations that provide wireless connectivity to ground users. To this end, first, we have utilized Google Earth Engine in or to extract key information about buildings in the considered geographical area. Then, we have studied the optimal deployment of drones in three practical scenarios. In the first scenario, we have determined the optimal locations of drones such that the number of covered ground users is maximized. In the second scenario, we have minimized the number of drones needed to ensure a full coverage for all users. Finally, we have minimized the flight time of drones required to completely service the users by satisfying their load requirements. Our results have shown that the proposed deployment framework significantly enhances the drone wireless system performance in terms of coverage and energy efficiency. Moreover, our simulation results have demonstrated existence of an optimal number of drones for which the wireless coverage is maximized.

Chapter 9

Conclusions and Open Problems

In this dissertation, we have identified and addressed a number of key challenges in CPSs security and in CI resilience with the aim of improving the cyber security and the resilience of critical CPSs. Towards achieving these goals, we have developed a number of analytical frameworks that capture different attack models against critical CPSs as well as other models to evaluate and improve the resilience of CIs. The formulated problems and their proposed solutions covered a number of CPS application domains. To this end, the developed cyber security solutions addressed various security problems such as: 1) Developing a defense mechanism to mitigate the effects of GPS spoofing attacks which can target UAVs with the goal of capturing the attacked UAVs, 2) Devising a moving target defense mechanism to harden attacker's mission of revealing the encryption keys used in a wireless communications, 3) Designing a moving target defense mechanism for m-health connected IoT devices. In addition, the developed solutions have also addressed CI resilience problems in different CI categories such as: 1) Developing a general framework for evaluating and improving the resilience of CIs against internal components failure, 2) Designing a resource allocation framework to improve the resilience of interconnected CIs under complete and incomplete information scenarios, 3) Formulating a solution approach to mitigate the effects of flooding on transportation networks, and, hence, improve their resilience, 4) Devising an environment-aware UAV deployment framework to improve the communication resilience under different disaster scenarios. In formulating and proposing solutions to the aforementioned problems, a number of mathematical tools have been used such as game theory, contract theory, dynamic programming, and optimization techniques.

In this regard, we next present a summary of the research work which have been performed in this dissertation.

9.1 Summary

Thus far, the main body of our work in this dissertation can be summarized as follows.

9.1.1 Protecting UAVs against GPS spoofing attacks

In Chapter 2, we have proposed a novel framework to analyze the problem of GPS spoofing attacks against UAVs. In particular, we have proposed a mathematical framework to mitigate the effects of capture attacks via GPS spoofing that target UAVs. We have used systems dynamics to model the UAVs' optimal routes towards their real destinations. Then, the effect of a GPS spoofer on these optimal routes is studied using the same concepts of system dynamics. To this end, we have mathematically derived the spoofer's optimal imposed locations on any UAV. These locations represent the optimal locations that when imposed on a UAV, will cause the UAVs to deviate from their planned routes towards the spoofer's desired destinations. At this point, the victim UAV will not realize that it is under attack. To detect the attacks, we have then proposed a countermeasure defense mechanism to allow UAVs to determine their real locations in case they are attacked. This countermeasure is built on the premise of cooperative localization, in which a UAV communicate with neighboring UAVs to request their locations, measure their relative locations, and then compute its real location. We have, then, formulated a dynamic Stackelberg game problem to allow the UAVs to better utilize the proposed defense mechanism. In particular, the game is a two-player game between a GPS spoofer (the attacker) and a drone operator (the defender) that manages a number of UAVs. Considering the hierarchical architecture of the Stackelberg games, we considered the drone operator as the leader that determines its strategies first and the spoofer as the follower that responds by choosing its strategies. We have mathematically derived the Stackelberg equilibrium strategies, for the formulated game, through a computationally efficient approach that reduces complexity of the original problem significantly. Results have shown that the proposed defense mechanism along with the dynamic Stackelberg equilibrium strategies outperform other strategy selection techniques in terms of reducing the possibility of UAV capture. In particular, two strategy selection techniques have been examined which are random and deterministic. We have then examined the effect of the instance drifted distance, update distance, and the average distances between the real and the attacker's destinations on the UAVs' deviation indices and on the possibility of UAV capture. Simulation results have shown that the UAV update distance has the most effect on both the deviation indices and UAVs capture possibility.

Limitations: The GPS spoofing model that we proposed in Chapter 2 has the following limitations:

- The velocity of the UAVs was assumed to be constant. This assumption will be valid only if the wind has a constant direction and speed. However, under dynamic changes in the wind, the velocity needs to be a function in time. To address this limitation, one can extend the UAV's travel model to capture the updated velocity after each time step. In this case, the time steps of the Stackelberg game will need to be synchronized with the velocity changes, in order to capture the accurate locations.

- The value of $e_{\max}$ is assumed to be constant and its range is determined based on the results of [104], which depends on using a Kalman filter to detect the spoofing attacks. However, if the attacker is able to manipulate the output of this Kalman filter, the value of $e_{\max}$ will be time dependent, and, hence, the model needs to be modified to capture this change. To

capture such a modification, one can extend the proposed Stackelberg game to a stochastic Stackelberg game in which the value of $e_{\max}$ is seen as a time-varying state of the game.

- The proposed defense mechanism requires the presence of four neighboring UAVs. If the number of neighboring UAVs, at any point of time, is less than four, the mechanism will fail to determine the real locations. Similarly, if the number of UAVs performing a mission is less than five, the proposed defense mechanism cannot be used. One approach to overcome this limitation is by allowing the UAVs to update their locations using the locations of fixed ground base stations. These ground base stations can send their location updates using frequencies different from the GPS signals. The defense mechanism will then need to be updated accordingly to capture these changes.

9.1.2 Single Controller Stochastic Games for Moving Target Defense

In Chapter 3, we have introduced a general framework to model and analyze the use of moving target defense (MTD) techniques. In particular, we have applied the framework on a cyber security problem in a wireless network security. The proposed MTD mechanism has been modeled using a non-zero sum stochastic game theory model. In this formulated problem, the defender was allowed to control the states transition, solely. Thus, the next state of the game is unilaterally determined by defender's actions. This property is supported by the concept of MTD in which the defender randomizes the system parameters to harden the attacker's mission and these changes should occur before the attacker can reveal the system parameters. This property of the system allowed to formulate the game as a single-controller stochastic game, which was shown in literature to always have an equilibrium point. To this end, we have provided and analyzed the mathematical model for deriving such equilibrium point in MTD games. We then provided a novel way to define the cost of applying MTD techniques in stochastic games which depends on the number of consecutive changes in system parameters. We have used two different functions to define the cost and have proved that the game will still have equilibrium when any of these cost functions apply to the system. Simulation results have shown that this framework can help the defender to get higher expected utilities in all system state than a baseline case of assigning equal probabilities over different actions.

9.1.3 Cyber Security of m-Health IoT systems

In Chapter 4, we have proposed a general security mechanism for m- Health IoT systems to protect users' information and their psrivacy. The mechanism depends on using secret keys between the devices and the gateway. This is because of the resource constrained nature of the m-Health IoT devices used around the patients. The framework also proposed to use public encryption keys between the (powerful) gateway and the remote locations, to where the data is shard. The framework, then, uses a MTD mechanism by frequently changing the encryption keys used in the network. The new keys are calculated locally in the network by encrypting the old keys using other keys known

as MTD keys which are pre-shared secret keys. Hence, only one key needs to be shared between the gateway and each device. We have, then, applied this MTD mechanism to a practical system that uses a light-weight encryption technique, LED. In particular, we have also designed a performance improvement technique for LED using bitslicing. To this end, we have formulated a 32-bit and 64-bit bitslicing implementations for LED. These bitslicing implementations represent the first 32-bit and 64-bit implementations of LED on ARM architecture. The different functions of LED have been carefully analyzed and equivalent functions have been designed for the custom bistliced representations. We have used a virtual processor, ARM cortex A-53, to evaluate our bitslicing implementations, and the results have shown a considerable performance improvement over the original LED implementation. We have, then, defined a cost for using the modified LED implementation with the MTD when packets are lost. The cost has been studied through simulations and the optimal number of packets to be used with the bitslicing implementation, have been derived. This optimal number ensures that the cost of missing packets to be bounded on the system. Results have shown that the worst-case cost for applying MTD is bounded by the number of instructions in the original LED implementation.

9.1.4 Evaluating and Improving Critical Infrastructure Resilience

In Chapter 5, we have proposed a novel framework to study and optimize the resilience of CIs against internal components failures. In particular, we have introduced a novel resilience index that is derived from the transition probabilities of a Markov chain representing the infrastructure's performance state. The CI state is defined to be either success, warning, or failure. Success represent the case when a CI is providing its designated service correctly, failure is the inability of a CI to provide the service, and warning is a newly defined state representing component failures that did not lead to the CI failure, yet. The resilience index, is then, derived based on the CI probability of failure on the long run, evaluated based on the transition probability from warning to failure states. We have then proposed a Bayesian network to model the hierarchical interaction between the infrastructure's physical components. The failure probability is then computed from the Bayesian network and is used on the resilience index calculation. Then, to prioritize the infrastructure's components in the resilience improvement process, we have introduced a Bayesian network algorithm that captures the effect of improving each component on the infrastructure's probability of failure. CI's components have been sorted based on their effect on the probability of failure, and, this sorting is used to determine the priority of fixing CI's components. To this end, we have evaluated the proposed framework in a case study of hydropower dams. Hydropower dams have been used because they capture both dams as CI and also the power grid (another CI) through the generated electricity. We have, then, defined a problem of allocating resources to a system of multiple CIs (hydropower dams) and studied it within the context of the case study. The problem has been modeled using contract theory in which a system operator wants to maximize the economic benefit from allocating the resources to multiple hydropower dams. In the formulated problem, the effects of one hydropower dam failure is used to evaluate its effect on the power grid. Dynamic programming optimization has been used to derive the optimal solution for the problem of resource

allocation allowing the operator to achieve the maximum economical benefit from allocating the resources. Results have shown that the proposed framework outperforms other allocation methods both in the economic reward for the system operator as well as the average resilience utility of all hydropower dams in the system.

9.1.5 Resource Allocation for Critical Infrastructure Protection

In Chapter 6, we have extended the problem of resource allocation to the case of asymmetric information. We have derived the analytical solution of the resource allocation problem in which the CIs are classified according to their vulnerability and criticality levels. The problem has been formulated using contract theory and the necessary and sufficient conditions for such resource allocation, under asymmetric information, have been derived. The problem has been, then, relaxed and solved to reach the optimal solution.

9.1.6 Transportation Networks Resilience against Flooding

In Chapter 7, we have studied the resilience of transportation networks against flooding. In particular, we have analyzed the effect of flooding on roads' capacities and on their free-flow travel times under different rain intensities. We have, then, proposed a framework to improve the transportation network resilience by reducing the overall system travel time after flooding. Computing the system travel times has been done through the traffic assignment problem based on the concept of Wardrop equilibrium. To this end, we have proposed to reduce the system travel time by shifting capacities (available lanes) between same road sides. The problem has been formulated as bi-level optimization in which the upper level solves for possible capacity shifts and the lower level provides feedback about users' travel times.

9.1.7 UAVs Deployment to Improve the Communication Resilience

In Chapter 8, we have proposed an environment-aware deployment framework of drone base stations that can provide wireless connectivity to ground users. To this end, we have utilized Google Earth Engine to extract key information about buildings that can affect the line of sight communications between the users and the drone base stations. We have, then, studied the optimal deployment of drones in three practical scenarios pertaining to natural-disaster's situations. Simulation results have shown that the proposed deployment framework significantly enhances the wireless coverage.

9.2 Open Problems

Despite the considerable number of studies in CPS security and in CI resilience, yet, there are a number of new directions that can be explored in both CI resilience and CPS security. As such, a number of key open problems, including possible extensions to the approaches in this dissertation, must be investigated as follows:

9.2.1 Deception as an effective Cyber Security Defense

Deceptive defense mechanisms are emerging novel approaches to thwart advanced attacks against CPSs. One approach is to use signaling [192] where the defender can transmit crafted signals to cause misconception at the attacker's side. This can be used for example, to deliver false information about the status of the CPS components. Thus, causing the attacker to target less valuable or honeypot components instead of the critical components. One approach to model deception using game theory is by using hyper game theory [193]. In hyper games, each player considers its actions by its belief or view of the game. Thus, the goal of the defender is to use signals to change the attacker's view of the game to achieve better outcomes. Another type of games that can be used is the newly defined Cyber Deception Games [194]. Although solving problems formulated using this type of games can be NP hard, algorithms can be formulated to find exact solutions of the game. Finally, we note that the problem of GPS spoofing introduced in Chapter 2 can be extended to utilize a deception approach. Thus, the defender can change the attacker's view of the protected UAVs in order achieve better outcome.

9.2.2 Artificial Intelligence Techniques CPSs Security

As discussed earlier, securing CPSs against cyber attacks is challenging due to the various factors. Of these factors is the interconnection between different CPSs and between large-scale CPSs components. As the number of these interconnections increase, mathematical models can become analytically intractable which compromises the ability of reaching optimal or closed form solutions for the problem under study. Artificial intelligence algorithms such as deep neural networks or reinforcement learning can thus help CPSs operators to learn the optimal strategies that can be applied to secure their CPSs. New security algorithms can then be developed, using machine learning, to better understand the complex interactions within the CPSs which, in turn, will help CPSs to be more secure against attacks.

9.2.3 Considering Additional Aspects for CI Resilience Evaluation

The problem of modeling CI resilience has been thoroughly addressed in this work. In particular, Chapter 5 proposes a complete framework for analyzing and improving the resilience of CIs. Yet,

some additional aspects can be considered in evaluating the resilience as follows:

Temporal aspects One possible extension for this framework is to consider the temporal effect of fixing CIs components. In this aspect, the physical components within a CI can have different times to be fixed. Thus, the CI components need to be prioritized based on both their effect on the CI probability of failure and on the time required to fix them. For example, if a component has a significant effect on the probability of failure, however, it is not the most affecting component; on the same time, this component requires less time to be fixed, then, it will be better for the CI to fix this component to gain a better short-term outcome. Considering the time required to fix components, the current Bayesian network analysis will not be able to model these changes, and, hence, a new model and analysis need to be developed.

Different warning states: Another future direction is to study our proposed resilience framework under multiple "warning" states where each state defines some degree of warning that range from small warnings to the complete failure.

Risk: Analyzing the notions of risk that can be incorporated in the principal's utility function, is another future direction. In particular, one can adopt the tools of behavioral contract theory [195] that relates the psychological behavior of the agents to their economic actions. Such a behavioral approach can help better evaluate the principal's risk in designing contracts for agents that possess different risk orientation, e.g., risk neutral or risk averse.

9.2.4 Cyber Resilience of Moving Target Defense (MTD)-enabled Critical Infrastructure

In Chapters 3 and 4, we have discussed the use of moving target defense as a security mechanism that can help improve the cyber security and, hence, the resilience of CI. The notions of MTD benefits and costs were also discussed in both chapters. Typically, in MTD, the system can switch between a number of configurations to increase its security. However, some of these configurations might cause the system to fail, if they are targeted by an attacker. As a future direction, it might be interesting to study the resilience of such MTD-enabled CI under certain attacks with the goal to find the optimum configuration for each CI to switch to, if it fails due to an attack. Choosing such a configuration is challenging as the CI should minimize the cost of configuration change plus ensuring longer stability on the new configuration in order to improve its resilience.

Bibliography

[1] M. Pregnolato, A. Ford, S. M. Wilkinson, and R. J. Dawson, "The impact of flooding on road transport: a depth-disruption function," *Transportation research part D: transport and environment*, vol. 55, pp. 67–81, 2017.

[2] R. Rajkumar, I. Lee, L. Sha, and J. Stankovic, "Cyber-physical systems: the next computing revolution," in *Design Automation Conference*. IEEE, 2010, pp. 731–736.

[3] Department of Homeland Security, "Critical infrastructure sectors," 2014. [Online]. Available: http://www.dhs.gov/critical-infrastructure-sectors

[4] B. A. Baalbaki, Y. Al-Nashif, S. Hariri, and D. Kelly, "Autonomic critical infrastructure protection (acip) system," in *ACS International Conference on Computer Systems and Applications (AICCSA)*, 2013, pp. 1–4.

[5] G. Sandaruwan, P. Ranaweera, and V. Oleshchuk, "Plc security and critical infrastructure protection," in *IEEE International Conference on Industrial and Information Systems (ICIIS)*, Peradeniya, Sri Lanka, Dec. 2013, pp. 81–85.

[6] J. McCausland, G. D. Nardo, R. Falcon, R. Abielmona, V. Groza, and E. Petriu, "A proactive risk-aware robotic sensor network for critical infrastructure protection," in *IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*, 2013, pp. 132–137.

[7] L. P. Beltran, M. Merabti, and Q. Shi, "Multiplayer game technology to manage critical infrastructure protection," in *IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2012, pp. 549–556.

[8] J. Moteff and P. Parfomak, "Critical infrastructure and key assets: definition and identification." LIBRARY OF CONGRESS WASHINGTON DC CONGRESSIONAL RESEARCH SERVICE, 2004.

[9] S. M. Condron, "Getting it right: Protecting american critical infrastructure in cyberspace," *Harv. JL & Tech.*, vol. 20, p. 403, 2006.

[10] T. G. Lewis, *Critical infrastructure protection in homeland security: defending a networked nation*. John Wiley & Sons, 2014.

[11] M. Ouyang, "Review on modeling and simulation of interdependent critical infrastructure systems," *Reliability engineering & System safety*, vol. 121, pp. 43–60, 2014.

[12] J. M. Yusta, G. J. Correa, and R. Lacal-Arántegui, "Methodologies and applications for critical infrastructure protection: State-of-the-art," *Energy Policy*, vol. 39, no. 10, pp. 6100–6119, 2011.

[13] A. Eldosouky, W. Saad, and N. Mandayam, "Resilient critical infrastructure: Bayesian network analysis and contract-based optimization," *arXiv preprint arXiv:1709.00303*, Aug. 2017.

[14] M. D. Bruijne and M. V. Eeten, "Systems that should have failed: critical infrastructure protection in an institutionally fragmented environment," *Journal of contingencies and crisis management*, vol. 15, no. 1, pp. 18–29, 2007.

[15] International Committee on Global Navigation Satellite Systems, *Critical Infrastructure Security and Resilience*. White House, November 2014.

[16] Q. Zhu and T. Başar, "Robust and resilient control design for cyber-physical systems with an application to power systems," in *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, 2011, pp. 4066–4071.

[17] A. Boin and A. McConnell, "Preparing for critical infrastructure breakdowns: the limits of crisis management and the need for resilience," *Journal of Contingencies and Crisis Management*, vol. 15, no. 1, pp. 50–59, 2007.

[18] D. A. Reed, K. C. Kapur, and R. D. Christie, "Methodology for assessing the resilience of networked infrastructure," *IEEE Systems Journal*, vol. 3, no. 2, pp. 174–180, 2009.

[19] A. French, M. Mozaffari, A. Eldosouky, and W. Saad, "Environment-aware deployment of wireless drones base stations with Google Earth simulator," in *Proceedings of UNAGI'19 - Workshop on UNmanned aerial vehicle Applications in the Smart City*, Kyoto,Japan, Mar 2019.

[20] A. Humayed, J. Lin, F. Li, and B. Luo, "Cyber-physical systems security—a survey," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 1802–1831, 2017.

[21] H. He and J. Yan, "Cyber-physical attacks and defences in the smart grid: a survey," *IET Cyber-Physical Systems: Theory & Applications*, vol. 1, no. 1, pp. 13–27, 2016.

[22] R. G. Little, "Toward more robust infrastructure: observations on improving the resilience and reliability of critical systems," in *Proc. of the 36th Annual Hawaii International Conference on System Sciences*. IEEE, Jan. 2003, pp. 1–9.

[23] T. Hashimoto, J. R. Stedinger, and D. P. Loucks, "Reliability, resiliency, and vulnerability criteria for water resource system performance evaluation," *Water resources research*, vol. 18, no. 1, pp. 14–20, Feb. 1982.

[24] D. R. W. Group, "1366-2012 - ieee guide for electric power distribution reliability indices," IEEE, Tech. Rep., 2012.

[25] Y. Iida, "Basic concepts and future directions of road network reliability analysis," *Journal of advanced transportation*, vol. 33, no. 2, pp. 125–134, 1999.

[26] J. D. Moteff, *Critical infrastructure resilience: the evolution of policy and programs and issues for congress*. Congressional Research Service US, Aug. 2012.

[27] National Infrastructure Advisory Council (US), *Critical Infrastructure Resilience: Final Report and Recommendations*, Aug. 2009.

[28] M. Bruneau, S. E. Chang, R. T. Eguchi, G. C. Lee, T. D. O'Rourke, A. M. Reinhorn, M. Shinozuka, K. Tierney, W. A. Wallace, and D. von Winterfeldt, "A framework to quantitatively assess and enhance the seismic resilience of communities," *Earthquake spectra*, vol. 19, no. 4, pp. 733–752, Nov. 2003.

[29] H. Chourabi, T. Nam, S. Walker, J. R. Gil-Garcia, S. Mellouli, K. Nahon, T. A. Pardo, and H. J. Scholl, "Understanding smart cities: An integrative framework," in *45th Hawaii International Conference on System Science (HICSS)*. IEEE, 2012, pp. 2289–2297.

[30] R. E. Hall, B. Bowerman, J. Braverman, J. Taylor, H. Todosow, and V. U. Wimmersperg, "The vision of a smart city," Brookhaven National Lab., Upton, NY (US), Tech. Rep., 2000.

[31] D. Washburn, U. Sindhu, S. Balaouras, R. A. Dines, N. Hayes, and L. E. Nelson, "Helping cios understand "smart city" initiatives," *Growth*, vol. 17, no. 2, pp. 1–17, 2009.

[32] F. Pasqualetti, F. Dörfler, and F. Bullo, "Cyber-physical attacks in power networks: Models, fundamental limitations and monitor design," in *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC),*, 2011, pp. 2195–2201.

[33] Y. Mo, T. H.-J. Kim, K. Brancik, D. Dickinson, H. Lee, A. Perrig, and B. Sinopoli, "Cyber–physical security of a smart grid infrastructure," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 195–209, 2012.

[34] F. Pasqualetti, F. Dörfler, and F. Bullo, "Attack detection and identification in cyber-physical systems," *IEEE Transactions on Automatic Control*, vol. 58, no. 11, pp. 2715–2729, 2013.

[35] A.-R. Sadeghi, C. Wachsmann, and M. Waidner, "Security and privacy challenges in industrial internet of things," in *52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2015, pp. 1–6.

[36] A. Sajid, H. Abbas, and K. Saleem, "Cloud-assisted iot-based scada systems security: A review of the state of the art and future challenges," *IEEE Access*, vol. 4, pp. 1375–1384, 2016.

[37] Y. Ashibani and Q. H. Mahmoud, "Cyber physical systems security: Analysis, challenges and solutions," *Computers & Security*, vol. 68, pp. 81–97, 2017.

[38] A. Eldosouky, W. Saad, and D. Niyato, "Single controller stochastic games for optimized moving target defense," in *Proceedings of IEEE International Conference on Communications (ICC)*, Kuala Lumpur, Malaysia, May 2016, pp. 1–6.

[39] S. Karnouskos, "Stuxnet worm impact on industrial cyber-physical system security," in *IECON 2011-37th Annual Conference of the IEEE Industrial Electronics Society*. IEEE, 2011, pp. 4490–4494.

[40] Y. Yang, K. McLaughlin, T. Littler, S. Sezer, E. G. Im, Z. Yao, B. Pranggono, and H. Wang, "Man-in-the-middle attack test-bed investigating cyber-security vulnerabilities in smart grid scada systems," 2012.

[41] Y. Yuan, Q. Zhu, F. Sun, Q. Wang, and T. Başar, "Resilient control of cyber-physical systems against denial-of-service attacks," in *2013 6th International Symposium on Resilient Control Systems (ISRCS)*. IEEE, 2013, pp. 54–59.

[42] Y. Mo and B. Sinopoli, "Integrity attacks on cyber-physical systems," in *Proceedings of the 1st international conference on High Confidence Networked Systems*. ACM, 2012, pp. 47–54.

[43] Y. Li, L. Shi, P. Cheng, J. Chen, and D. E. Quevedo, "Jamming attacks on remote state estimation in cyber-physical systems: A game-theoretic approach," *IEEE Transactions on Automatic Control*, vol. 60, no. 10, pp. 2831–2836, 2015.

[44] Y. Mo, T. H.-J. Kim, K. Brancik, D. Dickinson, H. Lee, A. Perrig, and B. Sinopoli, "Cyber–physical security of a smart grid infrastructure," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 195–209, 2012.

[45] Z. Li, M. Shahidehpour, A. Alabdulwahab, and A. Abusorrah, "Bilevel model for analyzing coordinated cyber-physical attacks on power systems," *IEEE Transactions on Smart Grid*, vol. 7, no. 5, pp. 2260–2272, 2016.

[46] R. Mitchell and R. Chen, "Effect of intrusion detection and response on reliability of cyber physical systems," *IEEE Transactions on Reliability*, vol. 62, no. 1, pp. 199–210, 2013.

[47] A.-R. Sadeghi, C. Wachsmann, and M. Waidner, "Security and privacy challenges in industrial internet of things," in *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. IEEE, 2015, pp. 1–6.

[48] I. Guvenc, F. Koohifar, S. Singh, M. L. Sichitiu, and D. Matolak, "Detection, tracking, and interdiction for amateur drones," *IEEE Communications Magazine*, vol. 56, no. 4, pp. 75–81, 2018.

[49] S. Sridhar, A. Hahn, and M. Govindarasu, "Cyber–physical system security for the electric power grid," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 210–224, 2012.

[50] A. Banerjee, K. K. Venkatasubramanian, T. Mukherjee, and S. K. S. Gupta, "Ensuring safety, security, and sustainability of mission-critical cyber–physical systems," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 283–299, 2012.

[51] E. K. Wang, Y. Ye, X. Xu, S.-M. Yiu, L. C. K. Hui, and K.-P. Chow, "Security issues and challenges for cyber physical system," in *2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing*. IEEE, 2010, pp. 733–738.

[52] H. Okhravi, T. Hobson, D. Bigelow, and W. Streilein, "Finding focus in the blur of moving-target techniques," *IEEE Security & Privacy,*, vol. 12, no. 2, pp. 16–26, 2014.

[53] J. Xu, P. Guo, M. Zhao, R. F. Erbacher, M. Zhu, and P. Liu, "Comparing different moving target defense techniques," in *Proceedings of the First ACM Workshop on Moving Target Defense*, Scottsdale, AZ, November 2014, pp. 97–107.

[54] P. McDaniel, T. Jaeger, T. F. L. Porta, N. Papernot, R. J. Walls, A. Kott, L. Marvel, A. Swami, P. Mohapatra, S. V. Krishnamurthy *et al.*, "Security and science of agility," in *Proceedings of the First ACM Workshop on Moving Target Defense*, Scottsdale, AZ, November 2014, pp. 13–19.

[55] R. Zhuang, S. A. DeLoach, and X. Ou, "Towards a theory of moving target defense," in *Proceedings of the First ACM Workshop on Moving Target Defense*, Scottsdale, AZ, November 2014, pp. 31–40.

[56] V. Casola, A. D. Benedictis, and M. Albanese, "A moving target defense approach for protecting resource-constrained distributed devices," in *IEEE 14th International Conf. on Information Reuse and Integration (IRI)*, San Francisco, CA, August 2013, pp. 22–29.

[57] A. Marttinen, A. M. Wyglinski, and R. Jantti, "Moving-target defense mechanisms against source-selective jamming attacks in tactical cognitive radio manets," in *IEEE Conf. on Communications and Network Security (CNS)*, San Francisco, CA, October 2014, pp. 14–20.

[58] J. H. Jafarian, E. Al-Shaer, and Q. Duan, "Openflow random host mutation: transparent moving target defense using software defined networking," in *Proceedings of the first workshop on Hot topics in software defined networks*, Helsinki, Finland, August 2012, pp. 127–132.

[59] S. Jajodia, A. K. Ghosh, V. Subrahmanian, V. Swarup, C. Wang, and X. S. Wang, *Moving Target Defense II*. Springer, 2013.

[60] Q. Zhu and T. Başar, "Game-theoretic approach to feedback-driven multi-stage moving target defense," in *Decision and Game Theory for Security*. Springer, 2013, pp. 246–263.

[61] K. M. Carter, J. F. Riordan, and H. Okhravi, "A game theoretic approach to strategy determination for dynamic platform defenses," in *Proceedings of the First ACM Workshop on Moving Target Defense*, Scottsdale, AZ, November 2014, pp. 21–30.

[62] M. Sherburne, R. Marchany, and J. Tront, "Implementing moving target ipv6 defense to secure 6lowpan in the internet of things and smart grid," in *Proc. of the 9th Annual Cyber and Information Security Research Conference*. ACM, 2014, pp. 37–40.

[63] P. K. Manadhata and J. M. Wing, "An attack surface metric," *IEEE Transactions on Software Engineering*, vol. 37, no. 3, pp. 371–386, 2011.

[64] L. Mili, "Taxonomy of the characteristics of power system operating states," in *2nd NSF-VT Resilient and Sustainable Critical Infrastructures (RESIN) Workshop*, , Tucson, AZ.

[65] R. R. Rajkumar, I. Lee, L. Sha, and J. Stankovic, "Cyber-physical systems: the next computing revolution," in *Proceedings of the 47th Design Automation Conference*. ACM, 2010, pp. 731–736.

[66] Y. Mo, T. H.-J. Kim, K. Brancik, D. Dickinson, H. Lee, A. Perrig, and B. Sinopoli, "Cyber–physical security of a smart grid infrastructure," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 195–209, 2012.

[67] D. Kyriazis, T. Varvarigou, D. White, A. Rossi, and J. Cooper, "Sustainable smart city iot applications: Heat and electricity management & eco-conscious cruise control for public transportation," in *2013 IEEE 14th International Symposium and Workshops on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2013, pp. 1–5.

[68] A. Eldosouky and W. Saad, "On the cybersecurity of m-health iot systems with led bitslice implementation," in *Proc. IEEE International Conference on Consumer Electronics (ICCE)*, Las Vegas, NV, USA, January 2018.

[69] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[70] W. He, G. Yan, and L. D. Xu, "Developing vehicular data cloud services in the iot environment," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1587–1595, 2014.

[71] A. Sanjab and W. Saad, "Data injection attacks on smart grids with multiple adversaries: a game-theoretic perspective," *IEEE Transactions on Smart Grid*, vol. 7, no. 4, pp. 2038–2049, 2016.

[72] A. Ferdowsi, A. Sanjab, W. Saad, and N. B. Mandayam, "Game theory for secure critical interdependent gas-power-water infrastructure," *arXiv preprint arXiv:1707.04589*, 2017.

[73] M. P. Scaparra and R. L. Church, "A bilevel mixed-integer program for critical infrastructure protection planning," *Computers & Operations Research*, vol. 35, no. 6, pp. 1905–1923, 2008.

[74] A. Eldosouky, A. Ferdowsi, and W. Saad, "Drones in distress: A game-theoretic countermeasure for protecting uavs against gps spoofing," *arXiv preprint arXiv:1904.11568*, 2019.

[75] W. H. Ip and D. Wang, "Resilience and friability of transportation networks: evaluation, analysis and optimization," *IEEE Systems Journal*, vol. 5, no. 2, pp. 189–198, 2011.

[76] E. D. Vugrin, D. E. Warren, and M. A. Ehlen, "A resilience assessment framework for infrastructure and economic systems: Quantitative and qualitative resilience analysis of petrochemical supply chains to a hurricane," *Process Safety Progress*, vol. 30, no. 3, pp. 280–290, 2011.

[77] M. Panteli and P. Mancarella, "The grid: Stronger, bigger, smarter?: Presenting a conceptual framework of power system resilience," *IEEE Power and Energy Magazine*, vol. 13, no. 3, pp. 58–66, May 2015.

[78] O. Yagan, D. Qian, J. Zhang, and D. Cochran, "Optimal allocation of interconnecting links in cyber-physical systems: Interdependence, cascading failures, and robustness," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 9, pp. 1708–1720, 2012.

[79] G. Brunner, "Using hec-ras for dam break studies," *US Army Corps of Engineers, Hydrologic Engineering Centre*.

[80] National Research Council and others, *Safety of dams: flood and earthquake criteria*. National Academies Press, 1985.

[81] A. Decò, P. Bocchini, and D. M. Frangopol, "A probabilistic approach for the prediction of seismic resilience of bridges," *Earthquake Engineering & Structural Dynamics*, vol. 42, no. 10, pp. 1469–1487, 2013.

[82] M. Pregnolato, A. Ford, S. M. Wilkinson, and R. J. Dawson, "The impact of flooding on road transport: a depth-disruption function," *Transportation research part D: transport and environment*, vol. 55, pp. 67–81, 2017.

[83] F. Petit, G. Bassett, R. Black, W. Buehring, M. Collins, D. Dickinson, R. Fisher, R. Haffenden, A. Huttenga, M. Klett, J. Phillips, M. Thomas, S. Veselka, K. Wallace, R. Whitfield, and J. Peerenboom, "Resilience measurement index: An indicator of critical infrastructure resilience," Argonne National Laboratory (ANL), Tech. Rep., Apr. 2013.

[84] M. Ouyang, L. Dueñas-Osorio, and X. Min, "A three-stage resilience analysis framework for urban infrastructure systems," *Structural Safety*, vol. 36, pp. 23–31, July 2012.

[85] Y.-P. Fang, N. Pedroni, and E. Zio, "Resilience-based component importance measures for critical infrastructure network systems," *IEEE Transactions on Reliability*, vol. 65, no. 2, pp. 502–512, June 2016.

[86] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Unmanned aerial vehicle with underlaid device-to-device communications: Performance and tradeoffs," *IEEE Transactions on Wireless Communications*, vol. 15, no. 6, pp. 3949–3963, Jun. 2016.

[87] Z. Zhang, L. Li, W. Liang, X. Li, A. Gao, W. Chen, and Z. Han, "Downlink interference management in dense drone small cells networks using mean-field game theory," in *Proceedings of the 10th International Conference on Wireless Communications and Signal Processing (WCSP)*, Hangzhou, China, Oct. 2018, pp. 1–6.

[88] J. C. Hodgson, S. M. Baylis, R. Mott, A. Herrod, and R. H. Clarke, "Precision wildlife monitoring using unmanned aerial vehicles," *Scientific reports*, vol. 6, p. 22574, Mar. 2016.

[89] M. Mozaffari, A. T. Z. Kasgari, W. Saad, M. Bennis, and M. Debbah, "Beyond 5G with UAVs: Foundations of a 3D wireless cellular network," *IEEE Transactions on Wireless Communications*, vol. 18, no. 1, pp. 357–372, Jan. 2019.

[90] R. Altawy and A. M. Youssef, "Security, privacy, and safety aspects of civilian drones: A survey," *ACM Transactions on Cyber-Physical Systems*, vol. 1, no. 2, p. 7, Feb. 2017.

[91] D. P. Shepard, J. A. Bhatti, T. E. Humphreys, and A. A. Fansler, "Evaluation of smart grid and civilian UAV vulnerability to GPS spoofing attacks," in *Radionavigation Laboratory Conference Proceedings*, 2012.

[92] A. J. Kerns, D. P. Shepard, J. A. Bhatti, and T. E. Humphreys, "Unmanned aircraft capture and control via GPS spoofing," *Journal of Field Robotics*, vol. 31, no. 4, pp. 617–636, Apr. 2014.

[93] J. S. Warner and R. G. Johnston, "Gps spoofing countermeasures," *Homeland Security Journal*, vol. 25, no. 2, pp. 19–27, Dec. 2003.

[94] M. U. Iqbal and S. Lim, "Legal and ethical implications of GPS vulnerabilities," *J. Int'l Com. L. & Tech.*, vol. 3, p. 178, 2008.

[95] B. W. O'Hanlon, M. L. Psiaki, T. E. Humphreys, and J. A. Bhatti, "Real-time spoofing detection using correlation between two civil GPS receiver," in *Proceedings of the ION GNSS Meeting*, Nashville, TN, USA, Sep. 2012.

[96] D. Schmidt, K. Radke, S. Camtepe, E. Foo, and M. Ren, "A survey and analysis of the GNSS spoofing threat and countermeasures," *ACM Computing Surveys (CSUR)*, vol. 48, no. 4, p. 64, May 2016.

[97] K. Jansen, M. Schäfer, D. Moser, V. Lenders, C. Pöpper, and J. Schmitt, "Crowd-GPS-Sec: Leveraging crowdsourcing to detect and localize GPS spoofing attacks," in *IEEE Symposium on Security and Privacy (SP)*, San Francisco, CA, May 2018, pp. 1018–1031.

[98] A. Ranganathan, H. Ólafsdóttir, and S. Capkun, "Spree: a spoofing resistant gps receiver," in *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*. ACM, 2016, pp. 348–360.

[99] D. M. Akos, "Who's afraid of the spoofer? GPS/GNSS spoofing detection via automatic gain control (agc)," *Navigation: Journal of the Institute of Navigation*, vol. 59, no. 4, pp. 281–290, Oct. 2012.

[100] P. Y. Montgomery, "Receiver-autonomous spoofing detection: Experimental results of a multi-antenna receiver defense against a portable civil gps spoofer," in *Radionavigation Laboratory Conference Proceedings*, 2011.

[101] K. Jansen, N. O. Tippenhauer, and C. Pöpper, "Multi-receiver gps spoofing detection: error models and realization," in *Proceedings of the 32nd Annual Conference on Computer Security Applications*. ACM, 2016, pp. 237–250.

[102] L. Heng, D. B. Work, and G. X. Gao, "Gps signal authentication from cooperative peers," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 4, pp. 1794–1805, 2015.

[103] Y. Qu and Y. Zhang, "Cooperative localization against GPS signal loss in multiple UAVs flight," *Journal of Systems Engineering and Electronics*, vol. 22, no. 1, pp. 103–112, Mar. 2011.

[104] J. Su, J. He, P. Cheng, and J. Chen, "A stealthy GPS spoofing strategy for manipulating the trajectory of an unmanned aerial vehicle," *IFAC-PapersOnLine*, vol. 49, no. 22, pp. 291–296, Sep. 2016.

[105] K. C. Zeng, Y. Shu, S. Liu, Y. Dou, and Y. Yang, "A practical GPS location spoofing attack in road navigation scenario," in *Proceedings of the 18th International Workshop on Mobile Computing Systems and Applications*. Sonoma, CA, USA: ACM, Feb. 2017, pp. 85–90.

[106] H. S. M. Coxeter, H. S. M. Coxeter, H. S. M. Coxeter, and H. S. M. Coxeter, *Introduction to geometry*. Wiley New York, 1969, vol. 136.

[107] A. Sinha, P. Malo, and K. Deb, "A review on bilevel optimization: from classical to evolutionary approaches and applications," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 2, pp. 276–295, Apr. 2018.

[108] Z. Han, D. Niyato, W. Saad, T. Başar, and A. Hjørungnes, *Game theory in wireless and communication networks: theory, models, and applications*. Cambridge University Press, 2012.

[109] M. Simaan and J. B. Cruz, "Additional aspects of the stackelberg strategy in nonzero-sum games," *Journal of Optimization Theory and Applications*, vol. 11, no. 6, pp. 613–626, Dec. 1973.

[110] T. Başar and G. J. Olsder, *Dynamic noncooperative game theory*. Siam, 1999, vol. 23.

[111] J. Lee, K. Kapitanova, and S. H. Son, "The price of security in wireless sensor networks," *Computer Networks*, vol. 54, no. 17, pp. 2967–2978, 2010.

[112] J. A. Filar and T. Raghavan, "A matrix game solution of the single-controller stochastic game," *Mathematics of Operations Research*, vol. 9, no. 3, pp. 356–362, 1984.

[113] J.-F. çois Mertens, "Stochastic games," *Handbook of game theory with economic applications*, vol. 3, pp. 1809–1832, 2002.

[114] A. Nowak and T. Raghavan, "A finite step algorithm via a bimatrix game to a single controller non-zero sum stochastic game," *Mathematical Programming*, vol. 59, no. 1-3, pp. 249–259, 1993.

[115] C. E. Lemke and J. T. Howson, Jr, "Equilibrium points of bimatrix games," *Journal of the Society for Industrial & Applied Mathematics*, vol. 12, no. 2, pp. 413–423, 1964.

[116] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Unmanned aerial vehicle with underlaid device-to-device communications: Performance and tradeoffs," *IEEE Transactions on Wireless Communications*, vol. 15, no. 6, pp. 3949–3963, June 2016.

[117] R. S. Istepanian, A. Sungoor, A. Faisal, and N. Philip, "Internet of m-health things 'm-iot'," in *IET Seminar on Assisted Living 2011*. IET, 2011, pp. 1–3.

[118] R. Istepanian, S. Laxminarayan, and C. S. Pattichis, *M-health*. Springer, 2006.

[119] C. Doukas, I. Maglogiannis, V. Koufi, F. Malamateniou, and G. Vassilacopoulos, "Enabling data protection through pki encryption in iot m-health devices," in *IEEE 12th International Conference on Bioinformatics & Bioengineering (BIBE)*, 2012, pp. 25–29.

[120] T. Park, N. Abuzainab, and W. Saad, "Learning how to communicate in the internet of things: Finite resources and heterogeneity," *IEEE Access*, vol. 4, pp. 7063–7073, Nov. 2016.

[121] H. Suo, J. Wan, C. Zou, and J. Liu, "Security in the internet of things: a review," in *International Conference on Computer Science and Electronics Engineering (ICCSEE)*, vol. 3, 2012, pp. 648–651.

[122] W. Saad, X. Zhou, B. Maham, T. Başar, and H. V. Poor, "Tree formation with physical layer security considerations in wireless multi-hop networks," *IEEE Transactions on Wireless Communications*, vol. 11, no. 11, pp. 3980–3991, Nov. 2012.

[123] J.-Y. Lee, W.-C. Lin, and Y.-H. Huang, "A lightweight authentication protocol for internet of things," in *International Symposium on Next-Generation Electronics (ISNE)*. IEEE, 2014, pp. 1–2.

[124] X. Wang, J. Zhang, E. M. Schooler, and M. Ion, "Performance evaluation of attribute-based encryption: Toward data privacy in the iot," in *IEEE International Conference on Communications (ICC)*, 2014, pp. 725–730.

[125] X. Yao, Z. Chen, and Y. Tian, "A lightweight attribute-based encryption scheme for the internet of things," *Future Generation Computer Systems*, vol. 49, pp. 104–112, 2015.

[126] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[127] Y. Sharaf-Dabbagh and W. Saad, "On the authentication of devices in the internet of things," in *Proceedings of 17th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, Coimbra, Portugal, june 2016, pp. 1–3.

[128] S. L. Albuquerque and P. R. Gondim, "Security in cloud-computing-based mobile health," *IT Professional*, vol. 18, no. 3, pp. 37–44, 2016.

[129] M. Sherburne, R. Marchany, and J. Tront, "Implementing moving target ipv6 defense to secure 6lowpan in the internet of things and smart grid," in *Proceedings of the 9th Annual Cyber and Information Security Research Conference*. ACM, 2014, pp. 37–40.

[130] J. Guo, T. Peyrin, A. Poschmann, and M. Robshaw, "The led block cipher," in *Cryptographic Hardware and Embedded Systems–CHES 2011*. Springer, 2011, pp. 326–341.

[131] H. Lipmaa, D. Wagner, and P. Rogaway, "Comments to nist concerning aes modes of operation: Ctr-mode encryption," 2000.

[132] ARM-Development-Tools. (2016) Arm ds-5 development studio. [Online]. Available: http://ds.arm.com/ds-5/

[133] J. Guo, T. Peyrin, A. Poschmann, and M. Robshaw. (2014) Led reference implementation. [Online]. Available: http://led.crypto.sg/downloads

[134] R. F. B. Bekera, "A metric and frameworks for resilience analysis of engineered and infrastructure system," *Reliability Engineering & System Safety*, vol. 121, pp. 90–103, Jan. 2014.

[135] E. D. Vugrin and R. C. Camphouse, "Infrastructure resilience assessment through control design," *International Journal of Critical Infrastructures*, vol. 7, no. 3, pp. 243–260, Jan. 2011.

[136] O. Yagan, D. Qian, J. Zhang, and D. Cochran, "Optimal allocation of interconnecting links in cyber-physical systems: Interdependence, cascading failures, and robustness," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 9, pp. 1708–1720, Sep. 2012.

[137] C. Liu, Y. Fan, and F. Ordóñez, "A two-stage stochastic programming model for transportation network protection," *Computers & Operations Research*, vol. 36, no. 5, pp. 1582–1590, May 2009.

[138] A. Eldosouky, W. Saad, C. Kamhoua, and K. Kwiat, "Contract-theoretic resource allocation for critical infrastructure protection," in *Proc. IEEE Global Communications Conference (GLOBECOM)*, Dec. 2015, pp. 1–6.

[139] P. Bolton and M. Dewatripont, *Contract Theory*. Cambridge, MA, USA: MIT Press, 2004.

[140] C. M. Grinstead and J. L. Snell, *Introduction to probability*. American Mathematical Soc., 2012.

[141] A. Lisnianski, D. Elmakias, D. Laredo, and H. B. Haim, "A multi-state markov model for a short-term reliability analysis of a power generating unit," *Reliability Engineering & System Safety*, vol. 98, no. 1, pp. 1–6, 2012.

[142] W. Dent and R. Ballintine, "A review of the estimation of transition probabilities in markov chains," *Australian journal of agricultural and resource economics*, vol. 15, no. 2, pp. 69–81, 1971.

[143] E. Charniak, "Bayesian networks without tears," *AI magazine*, vol. 12, no. 4, p. 50, Dec. 1991.

[144] G. F. Cooper, "The computational complexity of probabilistic inference using bayesian belief networks," *Artificial intelligence*, vol. 42, no. 2-3, pp. 393–405, Mar. 1990.

[145] J. Pearl, "Fusion, propagation, and structuring in belief networks," *Artificial Intelligence*, vol. 29, no. 3, pp. 241–288, Sep. 1986.

[146] W.-S. Lee, D. L. Grosh, F. A. Tillman, and C. H. Lie, "Fault tree analysis, methods, and applications: A review," *IEEE transactions on reliability*, vol. 34, no. 3, pp. 194–203, 1985.

[147] N. Khakzad, F. Khan, and P. Amyotte, "Safety analysis in process facilities: Comparison of fault tree and bayesian network approaches," *Reliability Engineering & System Safety*, vol. 96, no. 8, pp. 925–932, 2011.

[148] C.-G. Jong and S.-S. Leu, "Bayesian-network-based hydro-power fault diagnosis system development by fault tree transformation," *Journal of Marine Science and Technolog*, vol. 21, no. 4, pp. 367–379, 2013.

[149] A. Darwiche, *Modeling and reasoning with Bayesian networks*. Cambridge University Press, 2009.

[150] B. C. Yen and Y.-K. Tung, *Reliability and uncertainty analyses in hydraulic design*. ASCE Publications, 1993.

[151] J. Irizarry, M. Gheisari, and B. N. Walker, "Usability assessment of drone technology as safety inspection tools," *Journal of Information Technology in Construction*, vol. 17, pp. 194–212, Sep. 2012.

[152] T. Özaslan, S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar, "Inspection of penstocks and featureless tunnel-like environments using micro uavs," in *Field and Service Robotics*. Springer, 2015, pp. 123–136.

[153] P. Ridao, M. Carreras, D. Ribas, and R. Garcia, "Visual inspection of hydroelectric dams using an autonomous underwater vehicle," *Journal of Field Robotics*, vol. 27, no. 6, pp. 759–778, Nov. 2010.

[154] Federal Emergency Management Agency. (2016) Dam ownership in the united states. [Online]. Available: https://www.fema.gov/dam-ownership-united-states

[155] Federal Aviation Administration. (2016) Unmanned aircraft systems. [Online]. Available: https://www.faa.gov/uas/

[156] F. Qi and B.-N. Guo, "Monotonicity of sequences involving convex function and sequence," *RGMIA research report collection*, vol. 3, no. 2, 2000.

[157] S. Bradley, A. Hax, and T. Magnanti, "Applied mathematical programming," 1977.

[158] R. Bellman, "On the theory of dynamic programming," *Proc. of the National Academy of Sciences*, vol. 38, no. 8, pp. 716–719, Aug. 1952.

[159] A. Darwiche, "Samiam," *Software available from http://reasoning. cs. ucla. edu/samiam*, 2010.

[160] V. M. Bier, N. Haphuriwat, J. Menoyo, R. Zimmerman, and A. M. Culpen, "Optimal resource allocation for defense of targets based on differing measures of attractiveness," *Risk Analysis*, vol. 28, no. 3, pp. 763–770, 2008.

[161] Y. Huang, Y. Fan, and R. L. Cheu, "Optimal allocation of multiple emergency service resources for protection of critical transportation infrastructure," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2022, no. 1, pp. 1–8, 2007.

[162] M. Harris and R. M. Townsend, "Resource allocation under asymmetric information," *Econometrica: Journal of the Econometric Society*, pp. 33–64, 1981.

[163] Y. Zhang, L. Song, W. Saad, Z. Dawy, and Z. Han, "Contract-based incentive mechanisms for device-to-device communications in cellular networks," *IEEE Journal on Selected Areas in Communications (JSAC), Special issue on Heterogeneous Networks*, p. to appear, 2015.

[164] L. Duan, L. Gao, and J. Huang, "Cooperative spectrum sharing: A contract-based approach," *IEEE Transactions on Mobile Computing*, vol. 13, no. 1, pp. 174–187, Jan. 2014.

[165] L. Stole, "Lectures on the theory of contracts and organizations," *Unpublished monograph*, 2001.

[166] Y.-S. Kim, B. F. S. Jr, and A. S. Elnashai, "Seismic loss assessment and mitigation for critical urban infrastructure systems," Newmark Structural Engineering Laboratory. University of Illinois at Urbana-Champaign., Tech. Rep., 2008.

[167] M. G. Karlaftis, K. L. Kepaptsoglou, and S. Lambropoulos, "Fund allocation for transportation network recovery following natural disasters," *Journal of Urban Planning and Development*, vol. 133, no. 1, pp. 82–89, 2007.

[168] P. Bocchini and D. M. Frangopol, "Restoration of bridge networks after an earthquake: Multicriteria intervention optimization," *Earthquake Spectra*, vol. 28, no. 2, pp. 426–455, 2012.

[169] National Geographic, "Sea level rise will flood hundreds of cities in the near future," 2017. [Online]. Available: https://news.nationalgeographic.com/2017/07/sea-level-rise-flood-global-warming-science/

[170] Miami Herald, "Miami beach flooding spiked over last decade, um study finds," 2016. [Online]. Available: http://www.miamiherald.com/news/local/environment/article70145652.html

[171] M. Othman and A. A. Hamid, "Impact of flooding on traffic route choices," in *SHS Web of Conferences*, vol. 11. EDP Sciences, 2014, p. 01002.

[172] K. Pyatkova, A. S. Chen, S. Djordjevic, D. Butler, Z. Vojinović, Y. A. Abebe, and M. Hammond, "Flood impacts on road transportation using microscopic traffic modelling technique," 2015.

[173] J. G. Wardrop, "Some theoretical aspects of road traffic research." *Proceedings of the institution of civil engineers*, vol. 1, no. 3.

[174] Transportation Research Board, "Highway capacity manual," *National Research Council, Washington, DC*, vol. 113, 2000.

[175] H. Rakha, M. Farzaneh, M. Arafeh, and E. Sterzin, "Inclement weather impacts on freeway traffic stream behavior," *Transportation Research Record: Journal of the Transportation Research Board*, no. 2071, pp. 8–18, 2008.

[176] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Unmanned aerial vehicle with underlaid device-to-device communications: Performance and tradeoffs," *IEEE Transactions on Wireless Communications*, vol. 15, no. 6, pp. 3949–3963, June 2016.

[177] A. R. Khan, S. M. Bilal, and M. Othman, "A performance comparison of open source network simulators for wireless networks," in *Proc. of IEEE International Conference on Control System, Computing and Engineering*, Nov. 2012.

[178] J. Lessmann, P. Janacik, L. Lachev, and D. Orfanus, "Comparative study of wireless network simulators," in *Proc. of International Conference on Networking*, Apr. 2008.

[179] S. Kang, M. Aldwairi, and K.-I. Kim, "A survey on network simulators in three-dimensional wireless ad hoc and sensor networks," *International Journal of Distributed Sensor Networks*, vol. 12, no. 10, p. 1550147716664740, 2016.

[180] B. Newton, J. Aikat, and K. Jeffay, "Simulating large-scale airborne networks with ns-3," in *Proc. of the 2015 Workshop on ns-3*. ACM, 2015, pp. 32–39.

[181] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Efficient deployment of multiple unmanned aerial vehicles for optimal wireless coverage," *IEEE Communications Letters*, vol. 20, no. 8, pp. 1647–1650, Aug. 2016.

[182] A. Al-Hourani, S. Kandeepan, and S. Lardner, "Optimal LAP altitude for maximum coverage," *IEEE Wireless Communication Letters*, vol. 3, no. 6, pp. 569–572, Dec. 2014.

[183] E. Kalantari, H. Yanikomeroglu, and A. Yongacoglu, "On the number and 3D placement of drone base stations in wireless cellular networks," in *Proc. of IEEE Vehicular Technology Conference*, 2016.

[184] M. M. Azari, F. Rosas, K. C. Chen, and S. Pollin, "Joint sum-rate and power gain analysis of an aerial base station," in *Proc. of IEEE GLOBECOM Workshops*, Dec. 2016.

[185] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Wireless communication using unmanned aerial vehicles (UAVs): Optimal transport theory for hover time optimization," *IEEE Transactions on Wireless Communications*, vol. 16, no. 12, pp. 8052–8066, Dec. 2017.

[186] A. Zhang, X. Liu, A. Gros, and T. Tiecke, "Building detection from satellite images on a global scale," *available online: arxiv.org/abs/1707.08952*, 2017.

[187] M. Cote and P. Saeedi, "Automatic rooftop extraction in nadir aerial imagery of suburban regions using corners and variational level set evolution," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 51, no. 1, pp. 313–328, Jan. 2013.

[188] J. P. Cohen, W. Ding, C. Kuhlman, A. Chen, and L. Di, "Rapid building detection using machine learning," *Applied Intelligence*, vol. 45, no. 2, pp. 443–457, 2016.

[189] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.

[190] R. O. Duda and P. E. Hart, "Use of the hough transformation to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, no. 1, pp. 11–15, 1972.

[191] A. Hourani, S. Kandeepan, and A. Jamalipour, "Modeling air-to-ground path loss for low altitude platforms in urban environments," in *Proc. of IEEE Global Communications Conference (GLOBECOM)*, Austin, TX, USA, Dec. 2014.

[192] M. O. Sayin and T. Basar, "Deception-as-defense framework for cyber-physical systems," *arXiv preprint arXiv:1902.01364*, 2019.

[193] M. Wang, K. W. Hipel, and N. M. Fraser, "Modeling misperceptions in games," *Behavioral Science*, vol. 33, no. 3, pp. 207–223, 1988.

[194] A. Schlenker, O. Thakoor, H. Xu, F. Fang, M. Tambe, L. Tran-Thanh, P. Vayanos, and Y. Vorobeychik, "Deceiving cyber adversaries: A game theoretic approach," in *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2018, pp. 892–900.

[195] B. Kőszegi, "Behavioral contract theory," *Journal of Economic Literature*, vol. 52, no. 4, pp. 1075–1118, 2014.