RESILIENT AND SAFE CONTROL OF CYBER-PHYSICAL SYSTEMS UNDER UNCERTAINTIES AND ADVERSARIES

By

Aquib Mustafa

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Mechanical Engineering – Doctor of Philosophy

2020
RESILIENT AND SAFE CONTROL OF CYBER-PHYSICAL SYSTEMS UNDER UNCERTAINTIES AND ADVERSARIES

By

Aquib Mustafa

The recent growth of cyber-physical systems with a wide range of applications such as smart grids, healthcare, search and rescue and traffic monitoring, to name a few, brings new challenges to control systems due to the presence of significant uncertainties and undesired signals (i.e., disturbances and cyber-physical attacks). Thus, it is of vital importance to design resilient and safe control approaches that can adapt to the situation and mitigate adversaries to ensure an acceptable level of functionality and autonomy despite uncertainties and cyber-physical attacks.

This dissertation begins with the analysis of adversaries and design of resilient distributed control mechanisms for multi-agent cyber-physical systems with guaranteed performance and consensus under mild assumptions. More specifically, the adverse effects of cyber-physical attacks are first analyzed on the synchronization of the multi-agent cyber-physical systems. Then, information-theoretic based detection and mitigation methods are presented by equipping agents with self-belief about the trustworthiness of their own information and trust about their neighbors. Then, the effectiveness of the developed approach is certified by applying it to distributed frequency and voltage synchronization of AC microgrids under data manipulation attacks. In the next step, to relax some connectivity assumptions in the network for the resilient control design, a distributed adaptive attack compensator is developed by estimating the normal expected behavior of agents. The adaptive attack compensator is augmented with the controller and it is shown that the proposed controller achieves resilient synchronization in the presence of the attacks on sensors and actuators. Moreover, this approach recovers compromised agents under actuator attacks and avoids propagation of attacks on sensors without discarding information from the compromised agents.
Then, the problem of secure state estimation for distributed sensor networks is considered. More specifically, the adverse effects of cyber-physical attacks on distributed sensor networks are analyzed and attack mitigation mechanism for the event-triggered distributed Kalman filter is presented. It is shown that although event-triggered mechanisms are highly desirable, the attacker can leverage the event-triggered mechanism to cause triggering misbehaviors which significantly harms the network connectivity and performance. Then, an entropy estimation-based attack detection and mitigation mechanisms are designed.

Finally, the safe reinforcement learning framework for autonomous control systems under constraints is developed. Reinforcement learning agents with pre-specified reward functions cannot provide guaranteed safety across variety of circumstances that an uncertain system might encounter. To guarantee performance while assuring the satisfaction of safety constraints across variety of circumstances, an assured autonomous control framework is designed by empowering reinforcement learning algorithms with meta-cognitive learning capabilities. More specifically, adapting the reward function parameters of the reinforcement learning agent is performed in a meta-cognitive decision-making layer to assure the feasibility of the reinforcement learning agent.
ACKNOWLEDGEMENTS

I would like to express sincere thanks to my advisor Prof. Hamidreza Modares for his guidance, constant encouragement, and impeccable support during my doctoral research. I have been fortunate to learn the art of research under his exceptional tutelage. Apart from research, I have learned various things from him in my life during this journey.

I am thankful to my committee members, Prof. Ranjan Mukherjee, Prof. George Zhu, and Prof. Zhaojian Li, for their help and insightful comments. I thank all my collaborators for all their help throughout my doctoral work. I thank Prof. Ali Bidram for his guidance and immense support for Chapter 3 of this dissertation. I thank Dr. Majid Mazouchi for all his help and discussions in collaborative projects.

I believe the outcome of my year-long work was, by and large, a product of the excellent lab ambiance and support. I sincerely thank all my lab mates and friends. I am highly thankful to my close friends from Kanpur, Aligarh, Missouri, Colorado and of course Michigan for their help and support throughout this journey.

Finally, I would like to thank all my family members for their unconditional support and encouragement during this journey. Especially, I would like to thank my parents and elder brothers for their immense support and sacrifice. I shall remain ever indebted to them.
# TABLE OF CONTENTS

**LIST OF TABLES** ......................................................... ix

**LIST OF FIGURES** ....................................................... x

**CHAPTER 1  INTRODUCTION** ............................................. 1

1.1 Motivation ............................................................... 1
1.2 Literature Synopsis .................................................. 4
   1.2.1 Resilient Control Design for Distributed Multi-Agent Systems .... 4
   1.2.2 Secure Distributed State Estimation .................................. 6
   1.2.3 Safe Reinforcement Learning ........................................ 7
1.3 Contributions and outline ........................................... 8
1.4 Publications resulted from this work ................................ 11

**CHAPTER 2  RESILIENT SYNCHRONIZATION OF DISTRIBUTED MULTI-AGENT SYSTEMS UNDER ATTACKS** .......................... 13

2.1 Introduction ............................................................. 13
2.2 Preliminaries ........................................................... 13
2.3 Overview of Consensus in DMASs ..................................... 15
2.4 Attack Modelling and Analysis for DMASs ............................. 16
   2.4.1 Attack Modelling ..................................................... 16
   2.4.2 Attack Analysis ....................................................... 19
   2.4.3 Extension of Analysis Results to the Case of Noisy Communication 26
2.5 An Attack Detection Mechanism ....................................... 27
   2.5.1 Attack detection for IMP-based attacks ............................... 28
   2.5.2 Attack detection for non-IMP-based attacks .......................... 33
2.6 An Attack Mitigation Mechanism ...................................... 35
   2.6.1 Self-belief of agents about their outgoing information ............... 35
   2.6.2 Trust of agents about their incoming information .................... 37
   2.6.3 The mitigation mechanism using trust and self-belief values .......... 38
2.7 Simulation Results ..................................................... 41
   2.7.1 IMP-based attacks ................................................... 41
   2.7.2 Non-IMP-based attacks ............................................... 43
2.8 Conclusion ............................................................... 44
2.9 Appendix ................................................................. 45

**CHAPTER 3  DETECTION AND MITIGATION OF DATA MANIPULATION ATTACKS IN AC MICROGRIDS** ................................. 48

3.1 Introduction ............................................................. 48
3.2 Preliminaries ........................................................... 49
3.3 Conventional Distributed Secondary Control .......................... 49
3.4 Attack Modeling and Detection Mechanism ............................ 52

vi
LIST OF TABLES

Table 6.1: Vehicle Parameters .................................................. 149
LIST OF FIGURES

Figure 2.1: Schematic representation of the proposed resilient approach for DMASs. 38
Figure 2.2: Communication topology. ................................. 41
Figure 2.3: The state of agents when Agent 1 is under an IMP-based attack. .... 42
Figure 2.4: Agent 5 is under IMP-based attack. The state of agents. ............. 42
Figure 2.5: Agent 5 is under IMP-based attack. The local neighborhood tracking error of agents. ........................................ 42
Figure 2.6: Divergence for state of agents when Agent 5 is under a IMP-based attack. 43
Figure 2.7: The state of agents using the proposed attack detection and mitigation approach for IMP-based attack. ........................ 43
Figure 2.8: The state of agents when Agent 5 is under a non-IMP-based attack. ... 43
Figure 2.9: Divergence for state of agents when Agent 5 is under a non-IMP based attack. ...................................................... 44
Figure 2.10: The state of agents after attack detection and mitigation for non-IMP based attack. ................................................. 44
Figure 3.1: The flowchart of proposed attack detection and mitigation approach. ... 57
Figure 3.2: Single line diagram of the microgrid test system in Case A. ............. 62
Figure 3.3: Communication graph of the microgrid test system in Case A. ........... 62
Figure 3.4: Case A: Effect of attack on DSFC: (a) frequency; (b) active power ratio. 64
Figure 3.5: Case A: Relative entropy based on frequency of DERs. ...................... 64
Figure 3.6: Case A: Resilient DSFC: (a) frequency; (b) active power ratio. ........... 64
Figure 3.7: Case A: Resilient DSFC: (a) relative entropy; (b) self-believes of DERs. 64
Figure 3.8: Effect of periodic attack on DSFC with 0.05 s duration: (a) frequency; (b) active power ratio. ........................................... 66
Figure 3.9: Effect of periodic attack on DSFC with 0.05 s duration: (a) relative entropy; (b) self-believes of DERs. 66

Figure 3.10: Effect of periodic attack on DSFC with 0.5 s duration: (a) frequency; (b) active power ratio. 67

Figure 3.11: Fig. 12. Effect of periodic attack on DSFC with 0.5 s duration: (a) relative entropy; (b) self-believes of DERs. 67

Figure 3.12: Effect of attack on DER 2 in DSVC: (a) voltage (V); (b) reactive power ratio. 67

Figure 3.13: Case A: Relative entropy based on voltage of DERs. 68

Figure 3.14: Case A: Resilient DSVC: (a) voltage (V); (b) reactive power ratio. 68

Figure 3.15: Case A: Resilient DSVC: (a) relative entropy; (b) self-belief of DERs. 69

Figure 3.16: Microgrid testbed with 20 DERs. 70

Figure 3.17: Communication graph of the microgrid testbed in Case B. 70

Figure 3.18: Case B: Effect of attack on DSFC: (a) frequency; (b) active power ratio. 70

Figure 3.19: Case B: Relative entropy based on frequency of DERs. 71

Figure 3.20: Case B: Resilient DSFC: (a) frequency; (b) active power ratio. 71

Figure 3.21: Case B: Resilient DSFC: (a) relative entropy; (b) self-belief of DERs. 71

Figure 3.22: Microgrid test system for HIL testing. 73

Figure 3.23: HIL Setup. 73

Figure 3.24: Case C: Effect of attack on DSFC: (a) frequency; (b) active power ratio. 73

Figure 3.25: Case C: Relative entropy based on frequency of DERs. 74

Figure 3.26: Case C: Resilient DSFC: (a) frequency; (b) active power ratio. 74

Figure 4.1: Graph topology. 92

Figure 4.2: The Agents depth trajectory under the influence of the attacks on AUVs 2 and 3. Depth without adaptive compensator 93
Figure 4.3: The Agents depth trajectory under the influence of the attacks on AUVs 2 and 3. Depth with adaptive compensator ........................................ 93

Figure 5.1: Effect of non-triggering misbehavior on sensor nodes \{5,6\} cluster the graph \(G\) in the two isolated graphs \(G_1\) and \(G_2\). ........................................ 105

Figure 5.2: Communication topology ........................................ 120

Figure 5.3: Sensor network without any attack. (a) State estimation errors (b) Transmit function for sensor 2 ........................................ 120

Figure 5.4: Sensor node 2 under continuous-triggering misbehavior. (a) State estimation errors (b) Transmit function for sensor 2 ........................................ 121

Figure 5.5: Sensor node 2 under non-triggering misbehavior. (a) State estimation errors (b) Transmit function for sensor 2 ........................................ 122

Figure 5.6: Sensor node 2 under attack. (a) Estimated KL divergence (b) Confidence of sensors ........................................ 123

Figure 5.7: State estimation errors under attack on sensor 2 using proposed resilient state estimator ........................................ 123

Figure 6.1: Proposed metacognitive control scheme. \(S\): The system to be controlled; \(K\): low-level RL controller; \(C\): high-level metacognitive layer scheme. .... 133

Figure 6.2: Lane changing scenario for steering control of autonomous vehicle. .... 150

Figure 6.3: Lane changing with fixed value of hyperparameter without any change in dynamics ........................................ 150

Figure 6.4: Constraint violation during lane changing with fixed value of hyperparameter and change in dynamics ........................................ 151

Figure 6.5: Predicted fitness corresponding to vehicle trajectory under normal operation. 152

Figure 6.6: Predicted fitness corresponding to vehicle trajectory under constraint violation ........................................ 152

Figure 6.7: Overall fitness value under desired STL constraint violation for the vehicle trajectory ........................................ 152

Figure 6.8: Vehicle trajectory with hyperparameter adaptation based on Algorithm 1 for lane changing scenario ........................................ 153
Figure 6.9: Predicted fitness corresponding to vehicle trajectory with hyperparameter adaptation based on Algorithm 1. 153

Figure 6.10: Overall fitness value under desired STL constraint for adapted vehicle trajectory. 154
CHAPTER 1
INTRODUCTION

This chapter presents motivation, literature synopsis and contributions of this dissertation.

1.1 Motivation

A cyber-physical system (CPS) refers to a class of engineering systems that integrates the cyber aspect of computation and communication elements with physical entities. Based on their control objectives CPSs can be categorized into two classes, namely distributed multi-agent systems (DMASs) and networked control systems (NCSs). The control objective in DMAS is to achieve a coordinated or synchronized motion or behavior through the exchange of local information among agents [1]-[4]. On the other hand, the control objective in NCS, for which the feedback loops are closed through a communication network, is to regulate the system’s output to the desired value or trajectory [86]. Despite their numerous applications in a variety of disciplines, DMASs and NCSs are cyber-physical systems that bring new challenges to control systems due to the presence of significant uncertainties and undesired signals (i.e., disturbances and cyber-physical attacks). Thus, it is of vital importance to design resilient and safe control approaches that can adapt to the situation and mitigate adversaries to ensure an acceptable level of functionality and autonomy despite uncertainties and cyber-physical attacks.

The first part of this dissertation focus on attack analysis and resilient designs for DMASs. In the case of synchronization of DMASs, the coordination objective is to guarantee that all agents reach agreement on a common value or trajectory of interest. DMASs are cyber-physical systems that incorporate communication as a cyber component to facilitate the exchange of information among agents. This, however, makes them vulnerable to variety of attacks. In contrast to other undesirable inputs, such as disturbances and noises, attacks are intentionally planned to maximize the damage to the network. Therefore, it is important to
analyze the adverse effects of attacks on the performance and then design resilient DMASs that can mitigate attacks and guarantee an acceptable level of functionality despite attacks. To address the problem of interest, Chapters 2, 3 and 4 of this dissertation focus on attack analysis and resilient designs for DMASs.

Next, to perform monitoring and to successfully design a controller for different systems where state measurements are not available, one needs to do the state estimation over wireless sensor networks (WSNs). WSNs are a class of multi-agent CPSs for which a set of sensors are spatially distributed to monitor and estimate a variable of interest (e.g., location of a moving target, state of a large-scale system, etc.), and have various applications such as surveillance and monitoring, target tracking, and active health monitoring [103]. In centralized WSNs, all sensors broadcast their measurements to a center at which the information is fused to estimate the state [104]. These approaches, however, are communication demanding and prone to single-point-of-failure. To estimate the state with reduced communication burden, a distributed Kalman filter (DKF) is presented in [105]-[111], in which sensors exchange their information only with their neighbors, not with all agents in the network or a central agent. Cost constraints on sensor nodes in a WSN result in corresponding constraints on resources such as energy and communications bandwidth. Sensors in a WSN usually carry limited, irreplaceable energy resources and lifetime adequacy is a significant restriction of almost all WSNs. Therefore, it is important to design event-triggered DKF to reduce the communication burden which consequently improves energy efficiency. To this end, several energy-efficient event-triggered distributed state estimation approaches are presented for which sensor nodes intermittently exchange information [112]-[115]. Moreover, the importance of event-triggered state estimation problem is also reported for several practical applications such as smart grids and robotics [116]-[119]. Although event-triggered distributed state estimation is resource-efficient, it provides an opportunity for an attacker to harm the network performance and its connectivity by corrupting the information that is exchanged among sensors, as well as to mislead the event-triggered mechanism. Thus, it
is of vital importance to design a resilient event-triggered distributed state estimation approach that can perform accurate state estimation despite attacks. To address this problem, Chapter 5 of this dissertation first analyzes the adverse effects of attacks and then presents the secure state estimator for distributed sensor networks.

Finally, the safe control design problem for autonomous CPSs under uncertainties is addressed. More specifically, the safe reinforcement learning (RL) framework for autonomous control systems under constraints is developed. RL is a goal-oriented learning approach, inspired by biological systems, and is concerned with designing agents that can take actions in an environment to maximize some notion of cumulative reward [138]-[141]. Despite tremendous success of RL in a variety of applications, including robotics [142], control [143], and human-computer interaction [144], existing results are categorized as weak artificial intelligence (AI) [145]. That is, current RL practice has mainly been used to achieve pre-specified goals in structured environments by handcrafting a cost or reward function for which its minimization guarantees reaching the goal. Strong AI, on the other hand, holds the promise of designing agents that can learn to achieve goals across multiple circumstances by generalizing to unforeseen and novel situations. As the designer cannot foresee all the circumstances that the agent might encounter, pre-specifying and handcrafting the reward function cannot guarantee reaching goals in an uncertain and non-stationary environment. Reward shaping [146]-[147] has been presented in the literature with the primary goal of speeding up the learning without changing the outcome of the solution. Intrinsic motivated RL [148] has also been presented so that agents learn to shape their reward function to a better trade-off between exploration and exploitation or learn faster for the applications that the external environmental reward is sparse. Thus, in Chapter 6 of this dissertation, to guarantee performance while assuring the satisfaction of safety constraints across variety of circumstances, we present an assured autonomous control framework is designed by empowering RL algorithms with meta-cognitive learning capabilities.
1.2 Literature Synopsis

In this section, we review the literature in areas relevant to this dissertation. We organize the literature according to the broad topics of interest in this dissertation.

1.2.1 Resilient Control Design for Distributed Multi-Agent Systems

There has been a surge of interest in developing attack detection/identification and mitigation approaches for cyber-physical systems [5]-[25], including DMASs [16]-[25]. In [5]-[6], conditions under which an attacker can remain unnoticed are presented, followed by detection and identification mechanisms for attacks on sensors and actuators of cyber-physical systems. Resilient state estimation and control algorithms for cyber-physical systems under attacks are reported in [7]-[8]. A passivity-based attack mitigation mechanism is proposed in [9]. Teixeira et al. in [10] categorized attacks based on the attacker’s knowledge, disclosure, disruption resources, and characterized their impact using the concept of safe sets for cyber-physical systems. Various game-theoretic frameworks based resilient state estimation approaches are presented in [11]-[14]. Recently, in [15], authors presented proactive and reactive defense mechanisms to mitigate attacks on sensors and actuators. Although elegant, these aforementioned results do not apply to DMASs for which the overall objective is to synchronize the agents’ states to some value of interest.

For DMASs, attack detection and mitigation algorithms are presented in [16]-[18]. Mean square subsequence based resilient distributed control protocols for consensus of DMASs are presented in [19]-[21]. In these approaches, agents discard neighbors’ information based on the discrepancy between their neighbors’ values and their own values. Moreover, the maximum number of agents under attack is assumed known and a network connectivity assumption is made based on that. Adaptive local resilient control protocols based attack mitigation are designed to directly mitigate attack without identifying them using an observer-based approach [22]. An attacker-defender game framework is presented in [23] on networks with
unknown graph topology in which the defender injects control inputs to reach a consensus while attenuating the attack signal from compromised agents. Similarly, a controllability Gramian based game-theoretic approach is presented for resilient distributed consensus in [24]. A comprehensive survey on security of the cyber-physical system is presented in [25], categorizing the reported results for DMASs in three classes, called, prevention, resilience, and detection isolation. Despite tremendous and welcoming progress, most of the mentioned mitigation approaches for DMASs use the discrepancy among agents and their neighbors to detect and mitigate the effect of an attack. However, as shown in Chapters 2 and 4, a stealthy attack can make all agents unstable simultaneously, and thus misguide existing mitigation approaches. Moreover, this discrepancy could be caused by a legitimate change in the state of an agent, and rejecting this useful information can decrease the speed of convergence to the desired consensus and harm connectivity of the network.

Several remarkable results are presented for resilient control designs for important applications such as robotics and power systems. In [87]-[88], authors presented resilient algorithms for flocking and active target tracking applications in robotics, respectively. The work presented in [87] ensures a resilient consensus if the network connectivity is greater than a resilience threshold with the assumption that a compromised agent can share wrong information, but the actuator always works properly. The network connectivity constraints on the graph topology are relaxed in [89] by including trusted nodes. Similarly, the bulk of the research in cybersecurity of power systems focuses mainly on attack detection techniques [66]-[73]. Different techniques, including, adaptive cumulative sum using Markov-chain analysis [66], Kalman filter [67], graphical method [68], model-based scheme [69], matrix separation technique [70], Chi-square detector and cosine similarity matching approach [71], and nonlinear internal observer [72] are introduced for the attack detection in power systems with centralized control structure. The proposed attack detection filter in [16] and systematic detection and localization strategy in [73] tackle the attack detection in distributed control systems. In [74], signal temporal logic has been utilized for attack detection in a distributed
control system. Attack mitigation has also recently been considered in power systems. In [75], sensor fault detection and mitigation schemes are proposed to mitigate the impacts of cyber-attacks in DC power systems with a centralized control structure. Reference [76] proposes a trust/confidence-based approach for cyber-attack mitigation in the distributed control system of DC microgrids. In [78], a two-fold strategy is proposed to mitigate the impacts of FDI attacks on the control system of the shipboard power system. In [79], a trust/confidence-based control protocol is proposed to mitigate the impact of attacks on the distributed secondary control of AC microgrids. This approach, however, only considers the secondary frequency control and does not address the attack mitigation of secondary voltage control. In Chapter 3, we present FDI-attack detection and mitigation approaches for distributed secondary control of microgrids that are not limited to any specific type of attack with only mild restrictions on network connectivity.

1.2.2 Secure Distributed State Estimation

In recent years, secure state estimation of CPSs have received significant attention and remarkable results have been reported for mitigation of cyber-physical attacks, including denial of service attacks [10], [120], false data injection attacks [5]-[7],[121]-[122], and bias injection attacks [36], [123]. For the time-triggered distributed scenario, several secure state estimation approaches are presented in [124]-[131]. Specifically, in [124]-[132] authors presented a distributed estimator that allows agents to perform parameter estimation in the presence of attack by discarding information from the adversarial agents. Byzantine-resilient distributed estimator with deterministic process dynamics is discussed in [126]. Then, the same authors solved the resilient distributed estimation problem with communication losses and intermittent measurements in [127]. Attack analysis and detection for distributed Kalman filters are discussed in [128]. Resilient state estimation subject to denial of service attacks for power system and robotics applications is presented in [129]-[131]. Although elegant, these aforementioned results for the time-triggered resilient state estimation do not apply to event-
triggered distributed state estimation problems. To solve this problem, we analyze the effect of adversaries and design secure event-triggered distributed Kalman filters for state estimation over wireless sensor networks in Chapter 5.

1.2.3 Safe Reinforcement Learning

In the control community, several RL-based feedback controllers have been presented for control of uncertain dynamical systems [149]-[153]. In these traditional RL-based controllers, the reinforcement signal feedback is derived through a fixed quadratic objective function [154]-[155]. A fixed reward or objective function, however, cannot guarantee to achieve desired specifications across all circumstances. To express rich specifications rather than quadratic objectives, temporal logic, as an expressive language close to human language, has been widely used. Temporal logic is well suited for specifying goals and introducing domain knowledge for the RL problem [156]-[159]. RL with temporal logic specifications has also been used recently [160]-[163]. However, defining the rewards solely based on temporal logic specifications and ignoring numerical rewards can result in sparse feedbacks in control systems and cannot include other performance objectives such as energy and time minimization. Moreover, the system pursues several objectives and as the circumstance changes, the system’s needs and priorities also change, requiring adapting the reward signal to encode these needs and priorities to the context. It is therefore desired to design a controller that provides a good enough performance across variety of circumstances while assuring that its safety-related temporal logic specifications are satisfied. To this end, Chapter 6 of this dissertation takes a step towards strong AI for feedback control design by presenting a notion of adaptive reward function by introducing a metacognitive layer that decides on what reward function to optimize depending on the circumstance. More specifically, a metacognitive assured RL framework is presented to learn control solutions with good performances while satisfying desired specifications.
1.3 Contributions and outline

In this section, we outline the organization of the chapters in this dissertation and provide the contributions of each chapter. The key contributions of the dissertation are listed as follows.

- **Attack Analysis and Resilient Designs for Multi-agent CPSs**

  **Chapter 2:** In this chapter, we first address adverse effects of attacks on distributed synchronization of multi-agent systems, by providing conditions under which an attacker can destabilize the underlying network, as well as another set of conditions under which local neighborhood tracking errors of intact agents converge to zero. Based on this analysis, we propose a Kullback-Liebler divergence based criterion in view of which each agent detects its neighbors’ misbehavior and, consequently, forms a self-belief about the trustworthiness of the information it receives. Agents continuously update their self-beliefs and communicate them with their neighbors to inform them of the significance of their outgoing information. Moreover, if the self-belief of an agent is low, it forms trust on its neighbors. Agents incorporate their neighbors’ self-beliefs and their own trust values on their control protocols to slow down and mitigate attacks. We show that using the proposed resilient approach, an agent discards the information it receives from a neighbor only if its neighbor is compromised, and not solely based on the discrepancy among neighbors’ information, which might be caused by legitimate changes, and not attacks. The proposed approach is guaranteed to work under mild connectivity assumptions.

  **Chapter 3:** This chapter presents a resilient control framework for distributed frequency and voltage control of AC microgrids under data manipulation attacks. In order for each distributed energy resource (DER) to detect any misbehavior on its neighboring DERs, an attack detection mechanism is first presented using a Kullback-Liebler divergence-based criterion. An attack mitigation technique is then proposed
that utilizes the calculated KL divergence factors to determine trust values indicating the trustworthiness of the received information. Moreover, DERs continuously generate a self-belief factor and communicate it with their neighbors to inform them of the validity level of their own outgoing information. DERs incorporate their neighbors’ self-belief and their own trust values in their control protocols to slow down and mitigate attacks. It is shown that the proposed cybersecure control effectively distinguishes data manipulation attacks from legitimate events. The performance of proposed resilient frequency and voltage control techniques is verified through simulation of microgrid test system and hardware-in-the-loop (HIL) set-up using Opal-RT as a real-time digital simulator.

Chapter 4: This chapter analyzes the adverse effects of cyber-physical attacks on discrete-time distributed multi-agent systems, and proposes a mitigation approach for attacks on sensors and actuators. First, we show how an attack on a single node snowballs into a network-wide attack and even destabilizes the entire system. Next, to overcome the adversarial effects of attacks on sensors and actuators, a distributed adaptive attack compensator is designed by estimating the normal expected behavior of agents. The adaptive attack compensator is augmented with the controller and it is shown that the proposed controller achieves secure consensus in the presence of the attacks on sensors and actuators. No restrictive assumption on the number of agents under adversarial input is assumed. Moreover, it recovers compromised agents under actuator attacks and avoids propagation of attacks on sensors without discarding information from the compromised agents. Finally, numerical simulations validate the effectiveness of the presented theoretical contributions on a network of Sentry autonomous underwater vehicles.

Chapter 5: In this chapter, we analyze the adverse effects of cyber-physical attacks as well as mitigate their impacts on the event-triggered distributed Kalman filter (DKF). We first show that although event-triggered mechanisms are highly desirable, the attacker can leverage the event-triggered mechanism to cause non-triggering misbehav-
ior which significantly harms the network connectivity and its collective observability.
We also show that an attacker can mislead the event-triggered mechanism to achieve continuous-triggering misbehavior which not only drains the communication resources but also harms the network’s performance. An information-theoretic approach is presented next to detect attacks on both sensors and communication channels. In contrast to the existing results, the restrictive Gaussian assumption on the attack signal’s probability distribution is not required. To mitigate attacks, a meta-Bayesian approach is presented that incorporates the outcome of the attack detection mechanism to perform second-order inference. The proposed second-order inference forms confidence and trust values about the truthfulness or legitimacy of sensors’ own estimates and those of their neighbors, respectively. Each sensor communicates its confidence to its neighbors. Sensors then incorporate the confidence they receive from their neighbors and the trust they formed about their neighbors into their posterior update laws to successfully discard corrupted information. Finally, the simulation result validates the effectiveness of the presented resilient event-triggered DKF.

• Safe Reinforcement Learning for Autonomous Systems: A Metacognitive Framework

Chapter 6: This chapter presents a safe reinforcement learning framework for autonomous control systems under constraints. As RL agents with pre-specified reward functions cannot provide guaranteed safety across variety of circumstances that an uncertain system might encounter. To guarantee performance while assuring satisfaction of safety constraints across variety of circumstances, an assured autonomous control framework is presented by empowering RL algorithms with metacognitive learning capabilities. More specifically, adapting the reward function parameters of the RL agent is performed in a metacognitive decision-making layer to assure the feasibility of RL agent. That is, to assure that the learned policy by the RL agent satisfies safety constraints specified by signal temporal logic while achieving as much performance as
possible. The metacognitive layer monitors any possible future safety violation under the actions of the RL agent and employs a higher-layer Bayesian RL algorithm to proactively adapt the reward function for the lower-layer RL agent. To minimize the higher-layer Bayesian RL intervention, a fitness function is leveraged by the metacognitive layer as a metric to evaluate success of the lower-layer RL agent in satisfaction of safety and liveness specifications, and the higher-layer Bayesian RL intervenes only if there is a risk of lower-layer RL failure. Finally, a simulation example is provided to validate the effectiveness of the proposed approach.

1.4 Publications resulted from this work

Journal Articles:


Conferences:


CHAPTER 2
RESILIENT SYNCHRONIZATION OF DISTRIBUTED MULTI-AGENT SYSTEMS UNDER ATTACKS

2.1 Introduction

In this chapter, we present attack analysis, detection, and mitigation mechanisms for distributed multi-agent systems (DMASs). First, the adverse effects of cyber-physical attacks on the synchronization of the DMAS are described, supported with analysis. Specifically, conditions under which an attack can destabilize the entire network are provided. Moreover, conditions under which the local neighborhood tracking error of intact agents becomes zero while they are far from the synchronization are provided. The results of this analysis enables us to design detection mechanisms for sophisticated and threatening attacks. Then, two attack detectors are designed based on Kullback-Leibler (KL) divergence metrics to detect attacks that make the local neighborhood tracking error of intact agents zero and those for which this error cannot be zero. These detectors are then combined to detect variety of deception attacks. Finally, to mitigate attacks, self-belief (i.e., the belief on the trustworthiness of agents’ own information) and trust (i.e., the belief on trustworthiness of neighbors information) metrics are introduced based on the results from the detection mechanism. A weighted local neighborhood tracking error is introduced in which each agent incorporates trusts on its neighbors as well as the self-belief of its neighbors.

2.2 Preliminaries

A directed graph (digraph) \( \mathcal{G} \) consists of a pair \((\mathcal{V}, \mathcal{E})\) in which \( \mathcal{V} = \{v_1, \ldots, v_N\} \) is a set of nodes and \( \mathcal{E} \subseteq \mathcal{V} \times \mathcal{V} \) is a set of edges. We denote the directed link (edge) from \( v_j \) to \( v_i \) by the ordered pair \((v_j, v_i)\). The adjacency matrix is defined as \( \mathcal{A}^d = [a_{ij}] \), with \( a_{ij} > 0 \) if \((v_j, v_i) \in \mathcal{E}\), and \( a_{ij} = 0 \) otherwise. The nodes in the set \( \mathcal{N}_i = \{v_j : (v_j, v_i) \in \mathcal{E}\} \) are said to
be neighbors of node $v_i$. The in-degree of $v_i$ is the number of edges having $v_i$ as a head. The out-degree of a node $v_i$ is the number of edges having $v_i$ as a tail. If the in-degree equals the out-degree for all nodes $v_i \in V$ the graph is said to be balanced. The graph Laplacian matrix is defined as $L = D - A^d$, where $D = \text{diag}(d_i)$ is the in-degree matrix, with $d_i = \sum_{j \in N_i} a_{ij}$ as the weighted in-degree of node $v_i$. A node is called as a root node if it can reach all other nodes of the digraph $G$ through a directed path. A leader is a root node with no incoming link. A (directed) tree is a connected digraph where every node except one, called the root, has in-degree equal to one. A spanning tree of a digraph is a directed tree formed by graph edges, which connects all the nodes of the graph.

Throughout the chapter, we denote the set of integers by $\mathbb{Z}$. The set of integers greater than or equal to some integer $q \in \mathbb{Z}$ is denoted by $\mathbb{Z}_{\geq q}$. The cardinality of a set $S$ is denoted by $|S|$. $\lambda(A)$ and $\text{tr}(A)$ denote, respectively, the eigenvalues and trace of the matrix $A$. Furthermore, $\lambda_{\min}(A)$ represents the minimum eigenvalue of the matrix $A$. The Kronecker product of matrices $A$ and $B$ is denoted by $A \otimes B$, and $\text{diag}(A_1, \ldots, A_n)$ represents a block diagonal matrix with matrices $A_i, \forall i \in N$ as its diagonal entries. $\mathbf{1}_N$ is the $N$-vector of ones and $\mathbf{I}_N$ is the $N \times N$ identity matrix. $\|A\|$ denotes Euclidean norm of $A$. $\text{span}(a_1, \ldots, a_n)$ represents the set of all linear combinations of the vectors $a_1, \ldots, a_n$. A Gaussian distribution with mean $\mu$ and covariance $\Sigma$ is denoted by $\mathcal{N}(\mu, \Sigma)$. Moreover, $\mathcal{FN}(\bar{\mu}, \bar{\sigma}^2)$ represents univariate folded Gaussian distribution with $\bar{\mu}$ and $\bar{\sigma}^2$ as mean and variance, respectively [26]. $\mathbb{E}[]$ denotes the expectation operator. The term statistical properties is used for error sequences in the chapter to denote their mean and variance.

A system dynamics is called stable (i.e., Hurwitz) if all its eigenvalues have negative real part, and called unstable if it has eigenvalues with positive real part or repeated pair of eigenvalues on the imaginary axis. In this chapter, the term destabilize is used when the attacker makes system unstable.

**Assumption 1.** The communication graph $G$ is directed and has a spanning tree.

Note that having a spanning tree is the minimum requirement to guarantee synchroniza-
tion over a directed graph [27].

**Definition 1 [27]-[28].** A square matrix $A \in \mathbb{R}^{n \times n}$ is called a singular (non-singular) M-matrix, if all its off-diagonal elements are non-positive and all its eigenvalues have non-negative (positive) real parts. □

**Lemma 1 [27]-[28].** The graph Laplacian matrix $L$ of a directed graph $G$ has at least one zero eigenvalue, and all its nonzero eigenvalues have positive real parts. Zero is a simple eigenvalue of $L$, if and only if Assumption 1 is satisfied.

### 2.3 Overview of Consensus in DMASs

In this section, we provide an overview of the consensus problem for leaderless DMAS. Consider a group of $N$ homogeneous agents with linear identical dynamics described by

$$
\dot{x}_i(t) = Ax_i(t) + Bu_i(t) \quad \forall \, i \in \mathcal{N},
$$

where $x_i \in \mathbb{R}^n$ and $u_i \in \mathbb{R}^m$ denote, respectively, the state and the control input of agent $i$. The matrices $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{n \times m}$ are, respectively, the drift dynamics and the input matrix.

**Problem 1.** Design local control protocols $u_i$ for all agents $\forall \, i \in \mathcal{N}$ in (2.1) such that all agents reach consensus or synchronization on some common value or trajectory of interest, i.e.,

$$
\lim_{t \to \infty} \|x_j(t) - x_i(t)\| = 0 \quad \forall \, i, j \in \mathcal{N}.
$$

**Assumption 2.** The system dynamics matrix $A$ in (2.1) is assumed to be marginally stable with all eigenvalues on the imaginary axis [29].

Define the local neighborhood tracking error for the agent $i$ as

$$
\eta_i(t) = \sum_{j=1}^{N} a_{ij}(x_j(t) - x_i(t)),
$$

with $a_{ij}$ as the $(i,j)$-th entry of the graph adjacency matrix $A^d$.

Consider the distributed control protocol for each agent $i$ as [27]-[28]

$$
u_i(t) = cK \eta_i(t) \quad \forall \, i \in \mathcal{N},$$

(2.4)
where $c$ and $K \in \mathbb{R}^{m \times n}$ denote, respectively, the scalar coupling gain and the feedback control gain matrix.

Several approaches are presented to design $c$ and $K$ locally to solve Problem 1 [27]-[30]. To this end, the gains $c$ and $K$ are designed such that $A - c\lambda_i BK$ is Hurwitz for all $i = 2, \ldots, N$ [27]-[30], with $\lambda_i$ as the $i^{th}$ eigenvalue of the graph Laplacian matrix $\mathcal{L}$. In the subsequent sections, we assume that $c$ and $K$ are designed locally by each agent and without using a central agent appropriately to solve problem 1 in the absence of attack. We then analyze the effect of attacks and propose mitigation approaches.

**Remark 1.** Note that the presented results subsume the leader-follower synchronization problem and the average consensus as special cases. For the leader-follower case, the leader is only root node, whereas for the average consensus case, the graph is assumed to be balanced and $A = 0$ and $B = I_m$. $\square$

### 2.4 Attack Modelling and Analysis for DMASs

In this section, attacks on agents are modelled and a complete attack analysis is provided.

#### 2.4.1 Attack Modelling

In this subsection, attacks on DMASs are modelled. Attacks on actuators of agent $i$ can be modelled as

$$u_i^c = u_i + \beta_i u_i^d,$$  \hspace{1cm} (2.5)

where $u_i$, $u_i^d$ and $u_i^c$ denote, respectively, the nominal value of the control protocol for agent $i$ in (2.1), the disrupted signal directly injected into actuators of agent $i$, and the corrupted control protocol of agent $i$. If agent $i$ is under actuator attack, then $\beta_i = 1$, otherwise $\beta_i = 0$. Similarly, one can model attacks on sensors of agent $i$ as

$$x_i^c = x_i + \alpha_i x_i^d,$$  \hspace{1cm} (2.6)
where \( x_i \), \( x^d_i \) and \( x^c_i \) denote, respectively, the nominal value of the state of agent \( i \) in (2.3), the disrupted signal directly injected into sensors of agent \( i \), and the corrupted state of agent \( i \). If agent \( i \) is under sensor attack, then \( \alpha_i = 1 \), otherwise \( \alpha_i = 0 \). Using the corrupted state (2.6) in the controller (2.4)-(2.3) with the corrupted control input (2.5) in (2.1), the system dynamics under attack becomes

\[
\dot{x}_i = A x_i + B u_i + B f_i \quad \forall \ i \in \mathcal{N},
\]

(2.7)

where \( f_i \) denotes the overall attack affecting the agent \( i \) which can be written as

\[
f_i = \beta_i u^d_i + cK \sum_{j \in \mathcal{N}_i} a_{ij} \left( \alpha_j x^d_j - \alpha_i x^d_i \right),
\]

(2.8)

with \( u^d_i \) and \( x^d_i \) as attacks directly on actuators and sensors of agent \( i \), respectively, and \( x^d_j \) as the disruption in the received state of the \( j^{th} \) neighbor due to injected attack signal either into its sensors or actuators or into the incoming communication link from agent \( j \) to agent \( i \).

The following definition categorizes all attacks into two categories. The first type of attack exploits the knowledge of the system dynamics \( A \) and use it in the design of its attack signal. That is, for the first type of attack for \( f_i \) in (2.8), one has

\[
\dot{f}_i = \Psi f_i,
\]

(2.9)

where \( \Psi \in \mathbb{R}^{m \times m} \) depends on the knowledge of the system dynamics \( A \) as discussed in Definition 2. On the other hand, for the second type of attack, the attacker has no knowledge of the system dynamics \( A \) and this can cover all other attacks that are not in the form of (2.9).

Define

\[
\begin{align*}
E_\Psi &= \{ \lambda_1(\Psi), \ldots, \lambda_m(\Psi) \} \\
E_A &= \{ \lambda_1(A), \ldots, \lambda_n(A) \},
\end{align*}
\]

(2.10)

where \( \lambda_i(\Psi) \ \forall i = 1, \ldots, m \) and \( \lambda_i(A) \ \forall i = 1, \ldots, N \) are, respectively, the set of eigenvalues of the attack signal generator dynamics matrix \( \Psi \) and the system dynamics matrix \( A \).
Definition 2 (IMP-based and non-IMP-based Attacks). If the attack signal $f_i$ in (2.7) is generated by (2.9), then the attack signal is called the internal model principle (IMP)-based attack, if $E_{\Psi} \subseteq E_A$. Otherwise, i.e., $E_{\Psi} \not\subset E_A$ or if the attacker has no dynamics (e.g. a random signal), it is called a non-IMP based attack.

Remark 2. Note that we do not limit attacks to the IMP-based attacks given by (2.9). Attacks are categorized into two classes in Definition 2 based on their impact on the system performance, as to be shown in the subsequent sections. The non-IMP based attacks cover a broad range of attacks.

Definition 3 (Compromised and Intact Agent). We call an agent that is directly under attack as a compromised agent. An agent is called intact if it is not compromised. We denote the set of intact agents as $N_{\text{Int}}$, i.e., $N_{\text{Int}} = N \setminus N_{\text{Comp}}$ where $N_{\text{Comp}}$ denotes the set of compromised agents.

Definition 4. In a graph, Agent $i$ is reachable from agent $j$ if there exists $[v_1, v_2, \ldots, v_l] \in \mathcal{V}$ such that $a_j v_1 a v_1 v_2 \ldots a v_l i \neq 0$ for some $l \geq 0$, i.e., there is a directed path of length $l + 1$ from node $j$ to node $i$.

Using (2.4)-(2.3), the global form of control input, i.e., $u = [u_1^T, \ldots, u_N^T]^T$ can be written as

$$u = (-cL \otimes K)x,$$

where $L$ denotes the graph Laplacian matrix.

By using (2.11) in (2.7), the global dynamics of agents under attack becomes

$$\dot{x}(t) = \left(I_N \otimes A\right)x(t) + \left(I_N \otimes B\right)u^c(t),$$

where

$$u^c(t) = u(t) + f(t) \overset{\Delta}{=} (-cL \otimes K)x(t) + f(t),$$

with $f(t) = [f_1^T(t), \ldots, f_N^T(t)]^T$ and $x(t) = [x_1^T(t), \ldots, x_N^T(t)]^T$ denote, respectively, the overall vector of attacks on agents and the global vector of the states of agents. Note that
based on (2.8), and since corrupted sensory information is used by the controller, \( f(t) \) in (2.13) not only captures attacks on actuators, but also on sensors.

If agents are not under attack, i.e., \( f(t) = 0 \), then, the control input (2.11) eventually compensates for the difference between the agents’ initial conditions and becomes zero once they reach an agreement. That is, in the absence of attack, \( u^c(t) = u(t) \) goes to zero (i.e., \( u^c(t) \rightarrow 0 \)), and the global dynamics of agents become

\[
\dot{x}_{ss}(t) = (I_N \otimes A)x_{ss}(t),
\]

(2.14)

where \( x_{ss} = \lim_{t \rightarrow \infty} x(t) \) is called the global steady state of agents.

**Definition 5 (Steady State and Reaching Consensus).** We say that agents with the dynamics given by (2.7) and the global dynamics given by (2.12) reach a steady state if (2.14) is satisfied, i.e., if \( u^c(t) \rightarrow 0 \) in (2.13). In the absence of attack, if agents reach a steady state, then, agents reach consensus, i.e., \( \lim_{t \rightarrow \infty} ||x_j(t) - x_i(t)|| = 0 \) \( \forall i, j \in \mathcal{N} \). □

**Remark 3.** In the presence of attack, whether agents reach a steady state or not, i.e., whether \( u^c(t) \rightarrow 0 \) or \( u^c(t) \not\rightarrow 0 \), plays an important role in the attack analysis and mitigation to follow. Reaching a steady state is necessary for agents to achieve consensus based on Definition 5. However, we show that even if agents reach a steady state, they do not achieve consensus if the system is under attack. □

### 2.4.2 Attack Analysis

In this subsection, a graph theoretic-based approach is utilized to analyze the effect of attacks on DMASs. To this end, the following notation and lemmas are used.

Let the graph Laplacian matrix \( \mathcal{L} \) be partitioned as [27]

\[
\mathcal{L} = \begin{bmatrix}
\mathcal{L}_{r \times r} & 0_{r \times nr} \\
0_{nr \times r} & \mathcal{L}_{nr \times nr}
\end{bmatrix},
\]

(2.15)

where \( r \) and \( nr \) in (2.15) denote, respectively, the number of root nodes and non-root nodes. Moreover, \( \mathcal{L}_{r \times r} \) and \( \mathcal{L}_{nr \times nr} \) are, respectively, the sub-graph matrices corresponding to the
sub-graphs of root nodes and non-root nodes. The result of the following lemma is used in the proof of Theorem 1 to show that the local neighborhood tracking error goes to zero even in the presence of attack.

**Lemma 2.** Consider the partitioned graph Laplacian matrix (2.15). Then, $\mathcal{L}_{r \times r}$ is a singular M-matrix and $\mathcal{L}_{nr \times nr}$ is a non-singular M-matrix.

**Proof.** We first prove that the subgraph of root nodes is strongly connected. According to the definition of a root node, there always exists a directed path from a root node to all other nodes of the graph $\mathcal{G}$, including other root nodes. Therefore, in the graph $\mathcal{G}$, there always exists a path from each root node to all other root nodes. We now show that removing non-root nodes from the graph $\mathcal{G}$ does not affect the connectivity of the subgraph comprised of only root nodes. In the graph $\mathcal{G}$, if a non-root node is not an incoming neighbor of a root node, then its removal does not harm the connectivity of the subgraph of the root nodes. Suppose that removing a non-root node affects the connectivity of the subgraph of root nodes. This requires the non-root node to be an incoming neighbor of a root node. However, this makes the removed node a root node, as it can now access all other nodes through the root node it is connected to. Hence, this argument shows that the subgraph of root nodes is always strongly connected. Then, based on Lemma 1, $\mathcal{L}_{r \times r}$ has zero as one of its eigenvalues, which implies that $\mathcal{L}_{r \times r}$ is a singular M-matrix according to Definition 1. On the other hand, from (2.15), since $\mathcal{L}$ is a lower triangular matrix, the eigenvalues of $\mathcal{L}$ are the union of the eigenvalues of $\mathcal{L}_{r \times r}$ and $\mathcal{L}_{nr \times nr}$. Moreover, as stated in Lemma 1, $\mathcal{L}$ has a simple zero eigenvalue and, as shown above, zero is the eigenvalue of $\mathcal{L}_{r \times r}$. Therefore, all eigenvalues of $\mathcal{L}_{nr \times nr}$ have positive real parts only, and thus based on Definition 1, $\mathcal{L}_{nr \times nr}$ is a non-singular M-matrix.

In the following Lemmas 3-4 and Theorem 1, we now provide the conditions under which the agents can reach a steady state.

**Lemma 3.** Consider the global dynamics of DMAS (2.12) under attack. Let the attack
signal \( f(t) \) be a non-IMP based attack and \( f(t) \neq 0 \). Then, agents cannot reach a steady state, i.e., \( u^c(t) \not\rightarrow 0 \).

**Proof.** We prove this result by contradiction. Assume that the attack signal \( f(t) \) is a non-IMP based attack, i.e., \( E_\Psi \not\subset E_A \), but \( u^c(t) \rightarrow 0 \) in (2.12), which implies \( \dot{x}_i \rightarrow Ax_i \) for all \( i \in \mathcal{N} \). Using the modal decomposition, one has

\[
x_i(t) \rightarrow \sum_{j=1}^{n} (r_j x_i(0)) e^{\lambda_j (A)t} m_j,
\]

where \( r_j \) and \( m_j \) denote, respectively, the left and right eigenvectors associated with the eigenvalue \( \lambda_j (A) \). On the other hand, based on (2.13) \( u^c(t) \rightarrow 0 \) implies \( f(t) \rightarrow (cL \otimes K)x(t) \) or equivalently

\[
f_i(t) \rightarrow \sum_{j \in N_i} a_{ij} (x_j(t) - x_i(t)) \quad \forall \ i \in \mathcal{N}.
\]

As shown in (2.16), the right-hand side of (2.17) is generated by the natural modes of the system dynamics whereas the left-hand side is generated by the natural modes of the attack signal generator dynamics in (2.9). By the prior assumption, \( E_\Psi \not\subset E_A \), the attacker’s natural modes are different from those of the system dynamics. Therefore, (2.17) cannot be satisfied which contradicts the assumption. This completes the proof.

Equation (2.17) in Lemma 3 also shows that for non-IMP based attacks, the local neighborhood tracking error is nonzero for a compromised agent. The following results show that under IMP-based attack, either agents’ states diverge, or they reach a steady state while their local neighborhood tracking errors converge to zero, despite attack. The following lemma is needed in Theorem 1, which gives conditions under which agents reach a steady state under IMP-based attack. Then, Theorem 2 shows that under what conditions an IMP-based attack makes the entire network of agents unstable.

Define

\[
\begin{align*}
S_A(t) &= [e^{\lambda A_1 t}, \ldots, e^{\lambda A_n t}] \\
S_\psi(t) &= [e^{\lambda_\Psi_1 t}, \ldots, e^{\lambda_\Psi_n t}].
\end{align*}
\]
where $e^{\lambda_A t} \forall i = 1, \ldots, n$ and $e^{\lambda_{\Psi} t} \forall i = 1, \ldots, m$ are, respectively, the set of natural modes of agent dynamics $A$ in (2.1) and the attacker dynamics $\Psi$ in (2.9).

**Lemma 4.** Consider the global dynamics of DMAS (2.12) under attack on non-root nodes. Then, for an IMP-based attack, agents reach a steady state, i.e., $u^c(t) \to 0$.

**Proof.** According to (2.14), in steady state, one has $\dot{x}_{ss}(t) \to (I_N \otimes A)x_{ss}(t)$ since $u^c(t) \to 0$. This implies that $x_{ss}(t) \in \text{span}(S_A)$ where $S_A$ is defined in (2.18). On the other hand, if agents reach a steady state, then based on (2.13), one has

$$(cL \otimes K)x_{ss}(t) = f(t).$$  

(2.19)

Define the global steady state vector $x_{ss}(t) = [\bar{x}_{rs}^T, \bar{x}_{nrs}^T]^T$, where $\bar{x}_{rs}$ and $\bar{x}_{nrs}$ are, respectively, the global steady states of root nodes and non-root nodes. Since attack is only on non-root nodes, $f(t)$ can be written as $f(t) = [0_r, \bar{f}_{nr}^T]^T$, where $\bar{f}_{nr} = [f_{r+1}, \ldots, f_N]^T$ represents the attack vector on non-root nodes.

Then, using (2.15) and (2.19), one has

$$
\begin{cases}
(cL_{r \times r} \otimes K)\bar{x}_{rs} = 0 \\
(cL_{nr \times r} \otimes K)\bar{x}_{rs} + (cL_{nr \times nr} \otimes K)\bar{x}_{nrs} = \bar{f}_{nr}.
\end{cases}
$$

(2.20)

As stated in Lemma 2, $L_{r \times r}$ is a singular M-matrix with zero as an eigenvalue and $1_r$ is its corresponding right eigenvector and, thus, the solution to the first equation of (2.20) becomes $\bar{x}_{rs} = c_1 1_r$ for some positive scalar $c_1$. Using $\bar{x}_{rs} = c_1 1_r$ in the second equation of (2.20), the global steady states of non-root nodes becomes

$$\bar{x}_{nrs} = (cL_{nr \times nr} \otimes K)^{-1} [- (cL_{nr \times r} \otimes K)c_1 1_r + \bar{f}_{nr}].$$  

(2.21)

Equation (2.21) shows that the steady states of non-root nodes are affected by the attack signal $f(t)$. If $E_{\Psi} \nsubseteq E_A$, it results in $\bar{x}_{nrs} \in \text{span}(S_A, S_{\Psi})$ where $S_A$ and $S_{\Psi}$ are defined in (2.18) which contradicts $x_{ss}(t) \in \text{span}(S_A)$. Therefore, the condition $E_{\Psi} \subset E_A$ is necessary to conclude that for any $f = [0_r, \bar{f}_{nr}^T]^T$, there exists a steady state solution $x_{ss}(t)$, i.e., $u^c(t) \to 0$ holds true. This completes the proof. 

\[\square\]
The following theorem provides necessary and sufficient conditions for IMP-based attacks to make an agent reach a steady state, i.e., \( u^c(t) \to 0 \) in (2.13).

**Theorem 1.** Consider the global dynamics of DMAS (2.12) with the control protocol (2.13), where the attack signal \( f(t) \) is generated based on an IMP-based attack. Then, agents reach a steady state, i.e., \( u^c(t) \to 0 \) if and only if the attack signals satisfy

\[
\sum_{k=1}^{N} p_k f_k = 0, 
\]

where \( p_k \) are the nonzero elements of the left eigenvector of the graph Laplacian matrix \( \mathcal{L} \) associated with its zero eigenvalue.

**Proof.** It was shown in the Lemma 4 that for the IMP-based attack on non-root nodes, agents reach a steady state, i.e., \( u^c(t) \to 0 \). Therefore, whether agents reach a steady state or not depends solely upon attacks on root nodes. Let \( f(t) = [\bar{f}_r, \bar{f}_{nr}] \) where \( \bar{f}_r \) represents the vector of attacks for root nodes given by \( \bar{f}_r = [f_1^T, \ldots, f_r^T]^T \). Now, we first prove the necessary condition for root nodes. If \( u^c(t) \to 0 \), then, using (2.15) and (2.19), there exists a nonzero vector \( \bar{x}_{rs} \) for root nodes such that

\[
(c \mathcal{L}_{r \times r} \otimes K) \bar{x}_{rs} = \bar{f}_r,
\]

where \( \bar{x}_{rs} \) can be considered as the global steady state of the root nodes. Moreover, based on Lemma 3, (2.23) does not hold, if \( E_\Psi \not\subseteq E_A \) which implies that (2.23) is true only for \( E_\Psi \subseteq E_A \). As stated in Lemma 2, \( \mathcal{L}_{r \times r} \) is a strongly connected graph of root nodes and, therefore, it is a singular M-matrix. Let \( \bar{w}^T = [p_1, \ldots, p_r] \) be the left eigenvector associated with the zero eigenvalue of \( \mathcal{L}_{r \times r} \). Now, pre-multiplying both sides of (2.23) by \( \bar{w}^T \) and using the fact that \( \bar{w}^T \mathcal{L}_{r \times r} = 0 \) yield

\[
\bar{w}^T (c \mathcal{L}_{r \times r} \otimes K) \bar{x}_{rs} = \bar{w}^T \bar{f}_r = 0.
\]

This states that IMP-based attacks on root nodes have to satisfy \( \sum_{k=1}^{N} p_k f_k = 0 \) to ensure agents reach a steady state, i.e., \( u^c(t) \to 0 \). Note that \( p_k = 0 \) for \( k = r + 1, \ldots, N \), i.e., the
elements of the left eigenvector of the graph Laplacian matrix $L$, corresponding to its zero eigenvalue, are zero for non-root nodes [27]-[28]. This proves the necessity part.

Now, we prove the sufficient part by contradiction for root nodes. Assume agents reach a steady state, i.e., $u^c(t) \to 0$, but $\sum_{k=1}^{N} p_k f_k \neq 0$. Note that, agents reach a steady state implies that there exists a nonzero vector $\bar{x}_{rs}$ such that (2.23) holds. Using (2.24) and $\sum_{k=1}^{N} p_k f_k \neq 0$, one can conclude that $\bar{w}^T (cL_{r\times r} \otimes K) \bar{x}_{rs} \neq 0$. This can happen only when $L_{r\times r}$ does not have any zero eigenvalue, which violates the fact in Lemma 2 that $L_{r\times r}$ is a strongly connected graph. Therefore, $\bar{w}^T (cL_{r\times r} \otimes K) \bar{x}_{rs} = 0$ which results in $\sum_{k=1}^{N} p_k f_k = 0$ and contradicts the assumption made. This completes the proof. □

The following theorem provides conditions for IMP-based attack under which the network becomes unstable.

**Theorem 2.** Consider the global dynamics of DMAS (2.12) with the control protocol (2.13) under IMP-based attack. If (2.22) is not satisfied and $E_{\Psi} \cap E_{A} \neq \emptyset$, then the dynamics of agents become unstable.

**Proof.** Since it is assumed that the condition in (2.22) is not satisfied, then based on Theorem 1, $u^c(t) \not\to 0$ even under IMP-based attack. Thus, the attack signal $f(t)$ does not vanish over time and eventually acts as an input to the system in (2.12). Assume that there exists at least one common marginal eigenvalue between the system dynamics matrix $A$ in (2.1) and the attacker dynamics matrix $\Psi$ in (2.9), i.e., $E_{\Psi} \cap E_{A} \neq \emptyset$. Then, the multiplicity of at least one marginally stable pole becomes greater than 1. Therefore, the attacker destabilizes the state of the agent in (2.12). Moreover, since (2.22) is not satisfied, then the attack is on root nodes, and root nodes have a path to all other nodes in the network, the state of the all agents become unstable. This completes the proof. □

**Theorem 3.** Consider the global dynamics of DMAS (2.12) under attack $f(t)$. Then, the local neighborhood tracking error (2.3) converges to zero for all intact agents if $u^c(t) \to 0$. 24
Moreover, intact agents that are reachable from the compromised agents do not converge to the desired consensus trajectory.

Proof. In the presence of attacks, the global dynamics of the DMAS (2.12) with (2.13) can be written as

\[ \dot{x}(t) = (I_N \otimes A)x(t) + (I_N \otimes B)((-cL \otimes K)x(t) + f(t)), \]

where \( x(t) = [x_1^T(t), \ldots, x_N^T(t)]^T \) is the global vector of the state of agents and \( f(t) = [f_1^T(t), \ldots, f_N^T(t)]^T \) denotes the global vector of attacks. As shown in (2.14) that if \( u^c(t) \to 0 \), agents reach a steady state. That is,

\[ cK\eta_i \to -f_i \quad \forall i \in \mathcal{N}, \]

where \( \eta_i \) denotes the local neighborhood tracking error of agent \( i \) defined in (2.3). For an intact agent, by definition one has \( f_i = 0 \), and thus (2.26) implies that the local neighborhood tracking error (2.3) converges to zero. Now, we show that intact agents that are reachable from the compromised agent do not synchronize to the desired consensus behavior. To do this, let agent \( j \) be under attack. Assuming that all intact agents synchronize, one has \( x_k = x_i \ \forall i, k \in \mathcal{N} - \{j\} \). Now, consider the intact agent \( i \) as an immediate neighbor of the compromised agent \( j \). Then using (2.13), if \( u^c(t) \to 0 \), for intact agent \( i \), i.e., \( f_i = 0 \), one has

\[ \sum_{k \in \mathcal{N}_i - \{j\}} a_{ij}(x_k - x_i) + (x_j - x_i) \to 0, \]

where \( x_k \) denotes the state of the all intact neighbors of agent \( i \). On the other hand, (2.7) shows that the state of the compromised agent \( j \), i.e., \( x_j \), is deviated from the desired consensus value with a value proportional to \( f_j \). Therefore, (2.27) results in deviating the state of the immediate neighbor of the compromised agent \( j \) from the desired consensus behavior, which contradicts the assumption. Consequently, intact agents that have a path to the compromised agent do not reach consensus, while their local neighborhood tracking error is zero. This completes the proof. \( \square \)
Remark 4. An attacker can exploit the security of the network by eavesdropping and monitoring the transmitted data to identify at least one of the marginal eigenvalues of the agent dynamics by identifying the system dynamics using data-based approaches. Eavesdropping based design of attack signal is also discussed in [25]. □

2.4.3 Extention of Analysis Results to the Case of Noisy Communication

Up to now, the presented analysis has been under the assumption that the communication is noise free. We now briefly discuss what changes if the communication noise is present, and propose attack detection and mitigation in the presence of communication noise. In the presence of Gaussian distributed communication noise, the local neighborhood tracking error in (2.3) becomes

\[ \bar{\eta}_i = \eta_i + \omega_i, \]  
(2.28)

where \( \omega_i \sim \mathcal{N}(0, \Sigma_{\omega_i}) \) denotes the aggregate Gaussian noise affecting the incoming information to agent \( i \) and is given as

\[ \omega_i = \sum_{j \in \mathcal{N}_i} a_{ij} \omega_{ij}, \]  
(2.29)

where \( \omega_{ij} \) denotes noise in measurement from agent \( j \) to agent \( i \). In such situations, the DMAS consensus problem defined in Problem 1 changes to the mean square consensus problem [37].

In the presence of Gaussian noise, based on (2.28), the control protocol in (2.4)-(2.3) becomes [37]

\[ u_i(t) = cK a(t) \sum_{j=1}^{N} a_{ij} (x_j(t) - x_i(t)) + \omega_i \quad \forall \ i \in \mathcal{N}, \]  
(2.30)

with \( a(t) \) as time-dependent consensus gain and \( \omega_i \sim \mathcal{N}(0, \Sigma_{\omega_i}) \) is defined in (2.29). Based on mean square consensus, one has

\[ \lim_{t \to \infty} \mathbb{E}[u_i(t)] \to 0 \quad \forall \ i \in \mathcal{N}, \]  
(2.31)
and thus, based on (2.1), the steady state of agents converge to a consensus trajectory in mean square sense and its global form in (2.14) becomes

\[ \dot{x}_{ss}^{m} = (I_N \otimes A) x_{ss}^{m}, \]  

(2.32)

where \( x_{ss}^{m} = \lim_{t \to \infty} \mathbb{E}[x(t)] \) denotes the global steady state of agents in mean square sense. Then, following the same procedure as Lemmas 3-4 and Theorems 1-3, one can show that an IMP-based attack does not change the statistical properties of the local neighborhood tracking error, while a non-IMP based attack does. Moreover, the local neighborhood tracking error converges to zero in mean for an IMP-based attack, and it does not converges to zero in mean for a non-IMP based attack.

**Remark 5.** In general, the noise associated with electronic circuits at the receiver end lies under the category of thermal noise and statistically modeled as Gaussian [38]. Therefore, it is standard to assume \( \omega_{ij} \) in (2.29) is Gaussian [37]. However, if the assumption is violated, the same attack detection mechanism can be still developed using the divergence estimation approach presented in [39]-[40] for all distributions. □

### 2.5 An Attack Detection Mechanism

In this section, Kullback-Liebler (KL)-based attack detection and mitigation approaches are developed for both IMP-based and non-IMP-based attacks.

**Definition 6 (Kullback-Leibler Divergence) [41]-[42].** The Kullback-Leibler divergence between two probability densities \( P_X \) and \( P_Z \) of a random variable \( \Theta \) is defined as

\[ D_{KL}(X||Z) = \int_{\theta \in \Theta} P_X(\theta) \log \left( \frac{P_X(\theta)}{P_Z(\theta)} \right). \]  

(2.33)

In the following subsections, KL-divergence is used to detect IMP-based and non-IMP-based attacks on DMASs.
2.5.1 Attack detection for IMP-based attacks

In this subsection, an attack detector is designed to identify IMP-based attacks. To this end, two error sequences $\tau_i$ and $\varphi_i$ are defined based on only local exchanged information for agent $i$ as

$$\tau_i = \left\| \sum_{j \in N_i} a_{ij} d_{ij} \right\|$$  \hspace{1cm} (2.34)

and

$$\varphi_i = \sum_{j \in N_i} \left\| a_{ij} d_{ij} \right\|,$$  \hspace{1cm} (2.35)

where the measured discrepancy $d_{ij}$ between agent $i$'s state and its neighbor $j$'s state under attack becomes

$$d_{ij} = x^c_j - x^c_i + \omega_{ij} \quad \forall j \in N_i,$$  \hspace{1cm} (2.36)

where $\omega_{ij} \sim \mathcal{N}(0, \Sigma_{\omega_{ij}})$ denotes the Gaussian incoming communication noise from agent $j$ to agent $i$. Moreover, $x^c_i$ is the measured state of agent $i$ under attack and $x^c_j$ is the possibly corrupted information it receives from its $j^{th}$ neighbor. If agent $i$ is not compromised, then $x^c_i = x_i$, and, similarly, if agent $j$ is not compromised, then $x^c_j = x_j$. In fact, (2.34) is the norm of the summation of the measured discrepancy of agent $i$ and all its neighbors, and (2.35) is the summation of norms of those measured discrepancies. In the absence of attack, these two signals show the same behavior in the sense that their means converge to zero.

**Remark 6.** In the presence of an IMP-based attack and in the absence of noise, based on Theorem 3, $\tau_i$ goes to zero for intact agents, but $\varphi_i$ does not converge to zero, as its convergence implies consensus, which cannot not happen based on Theorem 3. On the other hand, for an IMP-based attack in the presence of noise, based on Theorem 3, $\tau_i$ converges to zero in mean because the local neighborhood tracking error converges to zero in mean for all agents. In contrast, the mean of $\varphi_i$ depends upon not only on the mean of the noise signal, but also of the attack signal. Therefore, the behavior of these two signals significantly
diverges in the presence of attacks which can be captured by KL-divergence based detection mechanism.

Note that one can measure $\tau_i$ and $\varphi_i$ based on the exchanged information among agents, which might be corrupted by the attack signal. Existing KL-divergence methods are, nevertheless, developed for Gaussian signals. However, while the communication noise is assumed to be Gaussian, error sequences (2.34) and (2.35) are norms of some variable with Gaussian distributions, thus, they have univariate folded Gaussian distributions given by [26]

$$\varphi_i \sim \mathcal{F}\mathcal{N}(\mu_{1i}, \sigma_{1i}^2)$$

and

$$\tau_i \sim \mathcal{F}\mathcal{N}(\mu_{2i}, \sigma_{2i}^2).$$

That is,

$$P_{\varphi_i}(q_i, \mu_{1i}, \sigma_{1i}) = \frac{1}{\sqrt{2\pi} \sigma_{1i}} e^{-\frac{(q_i-\mu_{1i})^2}{2\sigma_{1i}^2}} + \frac{1}{\sqrt{2\pi} \sigma_{1i}} e^{-\frac{(q_i+\mu_{1i})^2}{2\sigma_{1i}^2}},$$

$$P_{\tau_i}(q_i, \mu_{2i}, \sigma_{2i}) = \frac{1}{\sqrt{2\pi} \sigma_{2i}} e^{-\frac{(q_i-\mu_{2i})^2}{2\sigma_{2i}^2}} + \frac{1}{\sqrt{2\pi} \sigma_{2i}} e^{-\frac{(q_i+\mu_{2i})^2}{2\sigma_{2i}^2}},$$

where $\mu_{1i}$ and $\sigma_{1i}$ are the mean and variance of the error sequences $\varphi_i$ and $\mu_{2i}$ and $\sigma_{2i}$ are the mean and variance of the error sequences $\tau_i$. Using (2.33), the KL divergence in terms of the local error sequences $\varphi_i$ and $\tau_i$ can be defined as

$$D_{KL}(\varphi_i || \tau_i) = \int P_{\varphi_i}(q_i) \log \left( \frac{P_{\varphi_i}(q_i)}{P_{\tau_i}(q_i)} \right) dq_i = \mathbb{E}_{P_{\varphi_i}} \left( \log \frac{P_{\varphi_i}(q_i)}{P_{\tau_i}(q_i)} \right),$$

where $\mathbb{E}_{P_{\varphi_i}}[\cdot]$ represents the expectation value with respect to the distribution of the first sequence [26].

A KL divergence formula for the folded Gaussian distributions is now developed in the following lemma.

**Lemma 5.** Consider the error sequences $\tau_i$ and $\varphi_i$ in (2.34)-(2.35) with folded Gaussian distributions $P_{\varphi_i}$ and $P_{\tau_i}$ in (2.37). Then, the KL divergence between error sequences $\tau_i$ and $\varphi_i$, i.e., $D_{KL}(\varphi_i || \tau_i)$, becomes

$$D_{KL}(\varphi_i || \tau_i) \approx \frac{1}{2} \left( \log \frac{\sigma_{2i}^2}{\sigma_{1i}^2} - 1 + \frac{\sigma_{2i}^2}{\sigma_{1i}^2} + \frac{\sigma_{2i}^2}{\sigma_{1i}^2} (\mu_{2i} - \mu_{1i})^2 + 1 \right)$$

$$+ \frac{4\mu_{1i}^2}{\sigma_{1i}^2} \left( \frac{8\mu_{1i}^2}{\sigma_{1i}^2} + \frac{\mu_{1i}^2}{2\sigma_{1i}^2} \right) + e^{\frac{\mu_{1i}^2}{2\sigma_{1i}^2}} \left( \frac{1}{2} \left( \frac{\rho_{1i}^2}{e^{2\sigma_{1i}^2}} + e^{2\sigma_{1i}^2} \right) - \left( \frac{\rho_{1i}^2}{e^{2\sigma_{1i}^2}} + \frac{\rho_{1i}^2}{e^{2\sigma_{1i}^2}} \right) \right),$$

(2.39)
for some $\rho_1 = (\mu_{1i} - 2\mu_{2i}\sigma_{1i}^2\sigma_{2i}^2)$, $\rho_2 = (\mu_{1i} + 2\mu_{2i}\sigma_{1i}^2\sigma_{2i}^2)$, $\rho_3 = (\mu_{1i} - 4\mu_{2i}\sigma_{1i}^2\sigma_{2i}^2)$ and $\rho_4 = (\mu_{1i} + 4\mu_{2i}\sigma_{1i}^2\sigma_{2i}^2)$.

Proof. See Appendix A. \qed

Note that in (2.39), $\tau_i$ and $\phi_i$ are error sequences and the divergence between their distributions depends on their means and variances. One can calculate the value of these error sequences based on the equations (1.36) and (1.37) at each time instant. Then, based on previous $m$ samples one can determine the mean and the variance of distributions. Therefore, we do not explicitly need to know these statistical parameters.

In the following theorem, we show that the effect of IMP-based attacks can be captured using the KL divergence defined in (2.39).

**Theorem 4.** Consider the DMAS (2.1) along with the controller (2.13), and under the IMP-based attacks. Assume that the communication noise sequences are i.i.d. Then, for an intact agent $i$ reachable from the compromised agent,

$$
\frac{1}{T} \int_{k}^{k+T-1} D_{KL}(\phi_i || \tau_i) d\kappa > \gamma_i,
$$

(2.40)

where $\phi_i$ and $\tau_i$ are defined in (2.34) and (2.35), respectively, and $T$ and $\gamma_i$ represent the window size and the predesigned threshold parameter.

Proof. According to Theorem 3, the local neighborhood tracking error goes to zero for intact agents in the presence of an IMP-based attack when there is no communication noise. In the presence of communication noise with Gaussian distribution, i.e., $\omega_{ij} \sim (0, \Sigma_{\omega_{ij}})$ and IMP-based attack, the expectation value of the local neighborhood tracking error for intact agent $i$ becomes

$$
E[\eta_i] = E\left[ \sum_{j \in N_i} a_{ij} d_{ij} \right] \to 0,
$$

(2.41)
where the measured discrepancy $d_{ij}$ is defined in (2.36). Using (2.41), one can write (2.34) as

$$
\tau_i = \left\| \sum_{j \in N_i} a_{ij} d_{ij} \right\| \sim \mathcal{F}\mathcal{N}(0, \bar{\sigma}_i^2),
$$

(2.42)

which represents a folded Gaussian distribution with mean zero and variance $\bar{\sigma}_i^2$. Note that the mean and variance of the distribution $P_{\tau_i}$ in (2.37) become $\mu_{2i} = 0$ and $\sigma_{2i}^2 = \bar{\sigma}_i^2$.

Since noise signals are independent and identically distributed, from (2.35), one can infer that the folded Gaussian distribution $P_{\varphi_i}$ in (2.37) has the following statistical properties

$$
\varphi_i \sim \mathcal{F}\mathcal{N}(\mu_{fd_i}, \bar{\sigma}_i^2 + \hat{\sigma}_i^2 + \bar{\sigma}_i^2),
$$

(2.43)

where $\mu_{fd_i}$ and $\bar{\sigma}_i^2 + \hat{\sigma}_i^2 + \bar{\sigma}_i^2$ represent the overall mean and covariance, respectively, due to the communication noise and overall deviation from the desired behavior in intact neighbors reachable from the compromised agent.

In the absence of attack, the statistical properties corresponding to sequences $\tau_i$ and $\varphi_i$ become $\mathcal{F}\mathcal{N}(0, \bar{\sigma}_i^2)$ and $\mathcal{F}\mathcal{N}(0, \bar{\sigma}_i^2 + \hat{\sigma}_i^2)$, respectively, and the corresponding KL divergence in (2.39) becomes

$$
D_{KL}(\varphi_i | | \tau_i) \approx \frac{1}{2} \left( \log \frac{\bar{\sigma}_i^2}{\bar{\sigma}_i^2 + \hat{\sigma}_i^2} + \bar{\sigma}_i^2 \right),
$$

(2.44)

where $\hat{\sigma}_i^2$ represents additional variance in sequence $\varphi_i$, which depends on the communication noise.

Note that $\tau_i$ in (2.34) is the norm of the summation of the measured discrepancy of agent $i$ and all its neighbors whereas $\varphi_i$ in (2.35) is the summation of norms of those measured discrepancies. Even in the absence of attack, they represent folded Gaussian distributions with zero means and different covariances due to application of norm on measured discrepancies.

Now, in the presence of IMP-based attacks, using the derived form of KL divergence for
folded Gaussian distributions from Lemma 5, one can simplify (2.39) using (2.42)-(2.43) as

\[
D_{KL}(\varphi_i || \tau_i) \approx \frac{1}{2} \left( \log \frac{\bar{v}_i^2}{\bar{v}_i^2 + \bar{v}_{\omega_i}^2 + \bar{v}_{f_i}^2} + \frac{4(\mu_{f_i})^2}{\bar{v}_i^2 + \bar{v}_{\omega_i}^2 + \bar{v}_{f_i}^2} + \frac{8(\mu_{f_i})^2}{1 - e^{4(\mu_{f_i})^2}} \right) + \frac{1}{2} \bar{v}_i^2 (\mu_{f_i})^2 + \frac{1}{2} \bar{v}_i^2 (\mu_{f_i})^2 \right).
\]

(2.45)

Then, one can design the threshold parameter \( \gamma_i \) such that

\[
\frac{1}{T} \int_k^{k+T-1} D_{KL}(\varphi_i || \tau_i) d\kappa > \gamma_i, \tag{2.46}
\]

where \( T \) denotes the sliding window size. This completes the proof.

Based on Theorem 4, one can use the following conditions for attack detection [31]

\[
\begin{cases}
\frac{1}{T} \int_k^{k+T-1} D_{KL}(\varphi_i || \tau_i) d\kappa < \gamma_i & : H_0 \\
\frac{1}{T} \int_k^{k+T-1} D_{KL}(\varphi_i || \tau_i) d\kappa > \gamma_i & : H_1,
\end{cases}
\]

(2.47)

where \( \gamma_i \) denotes the designed threshold for detection, the null hypotheses \( H_0 \) represents the intact mode and \( H_1 \) denotes the compromised mode of an agent.

**Remark 7.** Note that several existing results on the faults/attack’s detection employ log-likelihood or generalized likelihood ratio-based test statistics [31]-[34]. Similarly, to get (1.47), we consider a log-likelihood ratio-based test statistic given by

\[
\Lambda(q_i) = \log \left( \frac{P_{\varphi_i}(q_i)}{P_{\tau_i}(q_i)} \right)
\]

(2.48)

where observations \( q_i \) are drawn from probability distribution functions \( P_{\varphi_i} \) and \( P_{\tau_i} \) defined in (2.37). Based on (2.38), the expectation of \( \Lambda(q_i) \) with respect to the probability distribution function \( P_{\varphi_i} \) becomes

\[
\mathbb{E}_{P_{\varphi_i}} [\Lambda(q_i)] = \mathbb{E}_{P_{\varphi_i}}[\log(P_{\varphi_i}(q_i)/P_{\tau_i}(q_i))] = D_{KL}(\varphi_i || \tau_i)
\]

(2.49)
In the absence of attack, i.e., when the intact mode hypothesis $H_0$ is true, based on Remark 6 and Theorem 4, $D_{KL}(\varphi_i||\tau_i)$ in (2.49) becomes a small positive value. This is because $P_{\varphi_i}$ and $P_{\tau_i}$ both have zero mean in the absence of attack. Moreover, in the presence of attack, i.e., when the compromised mode hypothesis $H_1$ is true, based on Remark 6 and Theorem 4, $D_{KL}(\varphi_i||\tau_i)$ in (2.49) becomes large because $P_{\varphi_i}$ and $P_{\tau_i}$ will have different mean and covariance values. Therefore, based on log-likelihood ratio-based test statistic, $D_{KL}(\varphi_i||\tau_i)$ is used in (2.47) for the attack detection over a sliding window. Note that the notion of sliding window is employed in order to avoid the false detection which may occur due to legitimate change, i.e., transient behavior of agents in the DMAS. The designed threshold $\gamma_i$ and the sliding window size $T$ in (2.47) are predefined parameters. The threshold $\gamma_i$ is typically designed based on the knowledge of the bound on the communication noise $\omega_{ij}$ in (36). This design choice, even though ensures that the compromised mode $H_1$ in (2.47) will not be active in the absence of attacks, can imply that the threshold is conservative. Selection of the predefined parameters for the adversary detection and stealthiness based on the system knowledge is reported in the literature [35]-[36]. □

2.5.2 Attack detection for non-IMP-based attacks

This subsection presents the design of a KL-based attack detector for non-IMP based attacks. It was shown in Theorem 3 that the local neighborhood tracking error goes to zero if agents are under IMP-based attacks. Therefore, for the case of non-IMP-based attacks, one can identify these types of attacks using the changes in the statistical properties of the local neighborhood tracking error. In the absence of attack, since the Gaussian noise, i.e., $\omega_i \sim \mathcal{N}(0, \Sigma_{\omega_i})$, is considered in the communication link, the local neighborhood tracking error $\eta_i$ in (2.28) has the following statistical properties

$$\eta_i \sim \mathcal{N}(0, \Sigma_{\omega_i}) \quad (2.50)$$

and it represents the nominal behavior of the system.
In the presence of attacks, using (2.28), the local neighborhood tracking error $\eta_i^a$ can be written as

$$\eta_i^a = \sum_{j \in N_i} a_{ij} d_{ij},$$

(2.51)

where measured discrepancy under attacks $d_{ij}$ is defined (2.36). From (2.51), one has

$$\eta_i^a \sim N(\mu_{fi}, \Sigma_{fi} + \Sigma_{\omega_i}),$$

(2.52)

where $\mu_{fi}$ and $\Sigma_{fi}$ are, respectively, mean and covariance of the overall deviation due to corrupted states under attacks as given in (2.36).

Now, since both $\eta_i^a$ and $\eta_i$ have normal Gaussian distributions, the KL divergence in the terms of $\eta_i^a$ and $\eta_i$ as $D_{KL}(\eta_i^a || \eta_i)$ can be written as [42]

$$D_{KL}(\eta_i^a || \eta_i) = \frac{1}{2} \left( \log \left| \frac{\Sigma_{\eta_i}}{\Sigma_{\eta_i^a}} \right| - n + tr(\Sigma_{\eta_i}^{-1} \Sigma_{\eta_i^a}) \right)$$

$$+ \frac{1}{2} (\mu_{\eta_i} - \mu_{\eta_i^a})^T \Sigma_{\eta_i}^{-1} (\mu_{\eta_i} - \mu_{\eta_i^a}),$$

(2.53)

where $\mu_{\eta_i}$ and $\Sigma_{\eta_i}$ denote the mean and covariance of $\eta_i$ and $\mu_{\eta_i^a}$ and $\Sigma_{\eta_i^a}$ denote the mean and covariance of $\eta_i^a$. Moreover, $n$ denotes the dimension of the error sequence. Define the average of KL divergence over a window $T$ as

$$\bar{D}_i = \frac{1}{T} \int_k^{k+T-1} D_{KL}(\eta_i^a || \eta_i) d\kappa.$$ 

(2.54)

The following theorem shows that the effect of non-IMP based attacks can be detected using the KL divergence between the two error sequences $\eta_i^a$ and $\eta_i$.

**Theorem 5.** Consider the DMAS (2.1) along with the controller (2.13). Then,

1. in the absence of attack, $\bar{D}_i$ defined in (2.54) tends to zero.

2. in the presence of a non-IMP-based attack, $\bar{D}_i$ defined in (2.54) is greater than a predefined threshold $\gamma_i$. 

34
Proof. In the absence of attacks, the statistical properties of sequences $\eta_i$ and $\eta_i^a$ are the same as in (2.50). Therefore, the KL divergence $D_{KL}(\eta_i^a||\eta_i)$ in (2.53) becomes zero and this makes $\bar{D}_i$ in (2.54) zero. This completes the proof of part 1.

To prove Part 2, using (2.50)-(2.52) in (2.53) and the fact that $(\Sigma_{\omega_i}^{-1}(\Sigma f_i + \Sigma \omega_i) - n = tr(\Sigma_{\omega_i}^{-1}\Sigma f_i)$, one can write the KL divergence between $\eta_i^a$ and $\eta_i$ as

$$D_{KL}(\eta_i^a||\eta_i) = \frac{1}{2} \log \frac{|\Sigma_{\omega_i}|}{|\Sigma f_i + \Sigma \omega_i|} + tr(\Sigma_{\omega_i}^{-1}\Sigma f_i) + \mu_f^T \Sigma_{\omega_i}^{-1} \mu_f).$$

(2.55)

Then, using (2.54), one has

$$\bar{D}_i = \frac{1}{T} \int_k^{k+T-1} \frac{1}{2} \log \frac{|\Sigma_{\omega_i}|}{|\Sigma f_i + \Sigma \omega_i|} + tr(\Sigma_{\omega_i}^{-1}\Sigma f_i) + \mu_f^T \Sigma_{\omega_i}^{-1} \mu_f) > \gamma_i,$$

(2.56)

where $T$ and $\gamma_i$ denote the sliding window size and the predefined design threshold, respectively. This completes the proof. $\square$

Based on Theorem 5, one can use the following conditions for attack detection

$$\begin{align*}
\bar{D}_i &< \gamma_i : H_0 \\
\bar{D}_i &> \gamma_i : H_1
\end{align*}$$

(2.57)

where $\gamma_i$ denotes the designed threshold for detection, $H_0$ represents the intact mode of the system and $H_1$ denotes the compromised mode of the system.

2.6 An Attack Mitigation Mechanism

In this section, both IMP-based and non-IMP-based attacks are mitigated using the presented detection mechanisms in the previous section.

2.6.1 Self-belief of agents about their outgoing information

To determine the level of trustworthiness of each agent about its own information, a self-belief value is presented.
Using the $D_{KL}(\varphi_i||\tau_i)$ and $D_{KL}(\eta_i^a||\eta_i)$ from Theorems 4 and 5, we define $c_i^1(t)$ as the self-belief of agent $i$ calculated under IMP-based attack and $c_i^2(t)$ as its self-belief under non-IMP based attack as

$$c_i^j(t) = \kappa_j \int_0^t e^{\kappa_j(t-\tau)} \chi_i^j(\tau) d\tau, \quad j = 1, 2, \quad (2.58)$$

where $0 \leq c_i^j(t) \leq 1$ with

$$\chi_i^1(t) = \frac{\Delta_i}{\Delta_i + D_{KL}(\varphi_i||\tau_i)} \quad (2.59)$$

and

$$\chi_i^2(t) = \frac{\Delta_i}{\Delta_i + D_{KL}(\eta_i^a||\eta_i)}, \quad (2.60)$$

Moreover, $\Delta_i$ represents the threshold to account for the channel fading and other uncertainties and $\kappa_j > 0$ denotes the discount factor. Note that $\chi_i^j(t), \ j = 1, 2$ in (2.58) depends on divergence between error sequences which are functions of time and consequently, $\chi_i^j(t)$ is also time-dependent. However, in order to maintain consistency with detection part, we have avoided time-indexing for divergence terms $D_{KL}(\varphi_i||\tau_i)$ and $D_{KL}(\eta_i^a||\eta_i)$. Based on Leibniz integral rule [43], equation (2.58) can be implemented by the following differential equation

$$\dot{c}_i^j(t) + \kappa_j c_i^j(t) = \kappa_j \chi_i^j(t), \quad j = 1, 2.$$

According to Theorems 4 (Theorem 5), for IMP-based attack (non-IMP based attack), the divergence term $D_{KL}(\varphi_i||\tau_i)$ ( $D_{KL}(\eta_i^a||\eta_i)$) increases, which makes $\chi_i^1(t)$ ($\chi_i^2(t)$) approach zero and consequently makes the value of $c_i^1(t)$ ($c_i^2(t)$) close to zero. On the other hand, without an attack, divergence term tend to zero, making $\chi_i^1(t)$ ($\chi_i^2(t)$) approach 1 and, consequently, $c_i^1(t)$ ($c_i^2(t)$) becomes close to 1. The larger the value of $c_i^j(t)$ is, the more confident the agent is about the trustworthiness of its broadcasted information.

Then, using $c_i^j(t), \ j = 1, 2$ defined in (2.58), the self-belief of an agent $i$ is defined as

$$\xi_i(t) = \min\{c_i^1(t), c_i^2(t)\}. \quad (2.61)$$
If an agent $i$ is under direct attack or receives corrupted information from its neighbors, then the self-belief of the agent $i$ tends to zero. In such a situation, it transmits the low self-belief value to its neighbor to put less weight on the information they receive from it and this prevents attack propagation in the distributed network.

### 2.6.2 Trust of agents about their incoming information

The trust value represents the level of confidence of an agent on its neighbors’ information. If the self-belief value of an agent is low, it forms beliefs on its neighbors (either intact or compromised) and updates its trust value which depends on the beliefs on each of its neighbors using only local information. Therefore, agents identify their compromised neighbors and discard their information.

Using the KL divergence between exchanged information of agent $i$ and its neighbor, one can define $\eta_{ij}(t)$ as

$$
\eta_{ij}(t) = \kappa_3 \int_{0}^{t} e^{\kappa_3(\tau-t)} L_{ij}(\tau) d\tau,
$$

where $0 \leq \eta_{ij}(t) \leq 1$ with

$$
L_{ij}(t) = 1 - \frac{\Lambda_1}{\Lambda_1 + e^{-\Lambda_2 D_{KL}(x_j||m_i)}} \quad \forall j \in N_i,
$$

with $m_i = \sum_{j \in N_i} x_j$ and $\Lambda_1, \Lambda_2 > 0$ represent the threshold to account for channel fading and other uncertainties, and $\kappa_3 > 0$ denotes the discount factor. For the compromised neighbor, the KL divergence $D_{KL}(x_j||m_i)$ tends to zero, which makes $L_{ij}(t)$ close to zero. Consequently, this makes the value of $\eta_{ij}(t)$ close to zero. On the other hand, if the incoming neighbor is not compromised, then $D_{KL}(x_j||m_i)$ increases and makes $\eta_{ij}(t)$ approach 1.

Based on Leibniz integral rule, equation (2.62) can be implemented using the following differential equation

$$
\dot{\eta}_{ij}(t) + \kappa_3 \eta_{ij}(t) = \kappa_3 L_{ij}(t).
$$
Now, we define the trust value of an agent on its neighbors as

$$\Omega_{ij}(t) = \max(\xi_i(t), \eta_{ij}(t)),$$

with $0 \leq \Omega_{ij}(t) \leq 1$.

In the absence of attacks, the state of agents converge to the consensus trajectory and the KL divergence $D_{KL}(x_j|m_i)$, $\forall j \in N_i$ tends to zero which results in $\Omega_{ij}(t)$ being $1 \forall j \in N_i$. In the presence of attacks, $\eta_{ij}(t)$ corresponding to the compromised agents tends to zero.

![Figure 2.1: Schematic representation of the proposed resilient approach for DMASs.](image)

### 2.6.3 The mitigation mechanism using trust and self-belief values

In this subsection, the trust and self-belief values are utilized to design the mitigation algorithm. To achieve resiliency, both self-belief and trust values are incorporated into the exchange information among agents as shown in Fig. 2.1. Consequently, the resilient form of local neighborhood tracking error (2.28) is presented as

$$\tilde{\eta}_i = \sum_{j \in N_i} \Omega_{ij}(t)\xi_j(t)a_{ij}(x_j - x_i) + \omega_i,$$

(2.65)
where \( \Omega_{ij}(t) \) and \( \xi_j(t) \) denote, respectively, the trust value in (2.65) and the self-belief of neighboring agents in (2.61). Based on the controller in (2.4) with resilient form of local neighborhood tracking error in (2.65), the resilient control protocol can be written as

\[
\tilde{u}_i = cK \tilde{\eta}_i, \quad \forall \ i \in \mathcal{N}.
\]  

(2.66)

According to (2.65), the topology of the graph changes over time due to incorporation of the trust and the self-belief values of agents, thus we denote the time-varying graph as \( G(t) = (\mathcal{V}, \mathcal{E}(t)) \) with \( \mathcal{E}(t) \subseteq \mathcal{V} \times \mathcal{V} \) representing the set of time-varying edges.

Now, based on following definitions and lemma, we formally present Theorem 6 to illustrate that the trust and self-belief based proposed resilient control protocol (2.66) solves Problem 1 for all intact agents, i.e., \( \mathcal{N}_{Int} = \mathcal{N} \setminus \mathcal{N}_{Comp} \) as defined in Definition 3 achieve the final desired consensus, regardless of attacks in the distributed network.

**Definition 7 (r-reachable set)** [20]. Given a directed graph \( G \) and a nonempty subset \( \mathcal{V}_s \subset \mathcal{V} \), the set \( \mathcal{V}_s \) is r-reachable if there exists a node \( i \in \mathcal{V}_s \) such that \( |\mathcal{N}_i \setminus \mathcal{V}_s| \geq r \), where \( r \in \mathbb{Z}_{\geq 0} \).

**Definition 8 (r-robust graph)** [20]. A directed graph \( G \) is called an r-robust graph with \( r \in \mathbb{Z}_{\geq 0} \) if for every pair of nonempty, disjoint subsets of \( \mathcal{V} \), at least one of the subsets is r-reachable.

**Assumption 3.** If at most \( q \) neighbors of each intact agents are under attack, at least \((q + 1)\) neighbors of each intact agents are intact [15].

**Lemma 6.** [20] Consider an r-robust time-varying directed graph \( G(t) \). Then, the graph has a directed spanning tree, if and only if \( G(t) \) is 1-robust.

The following theorem shows that the proposed resilient controller (2.66) guarantees synchronization despite attack.

**Theorem 6.** Consider the DMAS (2.1) under attack with the proposed resilient control protocol \( \tilde{u}_i \) in (2.66). Let the time-varying graph \( G(t) \) be such that at each time instant \( t \), Assumption 1 and Assumption 3 are satisfied. Then, \( \lim_{t \to \infty} ||x_j(t) - x_i(t)|| = 0 \) \( \forall i, j \in \mathcal{N}_{Int} \).
Proof. The DMAS (2.1) with the proposed resilient control protocol $\tilde{u}_i$ in (2.66) without noise can be written as

$$\dot{x}_i = Ax_i + cBK \sum_{j \in N_i} a_{ij}(t) (x_j - x_i),$$

(2.67)

with $a_{ij}(t) = \Psi_{ij}(t)C_j(t)a_{ij}$, where $\Omega_{ij}(t)$ and $\xi_j(t)$ represent, respectively, the trust value in (2.65) and the self-belief of neighboring agents in (2.61). The global form of resilient system dynamics in (2.67) becomes

$$\dot{x} = (I_N \otimes A - cL(t) \otimes BK)x,$$

(2.68)

where $L(t)$ denotes the time-varying graph Laplacian matrix of the directed graph $G(t)$. Based on Assumption 3, even if $q$ neighbors of an intact agent are attacked and collude to send the corrupted value to misguide it, there still exists $q+1$ intact neighbors that communicate values different from the compromised ones. Moreover, since at least $q+1$ of the intact agent’s neighbors are intact, it can update its trust values to remove the compromised neighbors. Furthermore, since the time varying graph $G(t)$ resulting from isolating the compromised agents is 1-robust, based on Definition 8 and Lemma 6, the entire network is still connected to the intact agents. Therefore, there exists a spanning tree in the graph associated with all intact agents $\mathcal{N}_{\text{Int}}$. Hence, it is shown in [44] that the solutions of DMAS in (2.68) reach consensus on desired behavior if the time-varying graph $G(t)$ jointly contains a spanning tree as the network evolves with time. This results in $\lim_{t \to \infty} ||x_j(t) - x_i(t)|| = 0 \quad \forall \ i, j \in \mathcal{N}_{\text{Int}}$ asymptotically. This completes the proof.

Remark 8. The proposed approach discards the compromised agent only when an attack is detected, in contrast to most of the existing methods that are based solely on the discrepancy among agents. Note that discrepancy can be the result of a legitimate change in the state of one agent. Moreover, in the beginning of synchronization, there could be a huge discrepancy between agents’ states that should not cause discarding information.
2.7 Simulation Results

Consider a group of 5 homogeneous agents with the dynamics defined as

\[
\dot{x}_k = Ax_k + Bu_k \quad k = 1, \ldots, 5, \tag{2.69}
\]

where \( A = [0, -1; 1 0] \) and \( B = [1, 1] \).

The communication graph is shown in Fig. 2.2. We assume zero-mean Gaussian communication noise with following statistical properties \( \mathcal{N}(0, 0.1) \).

\[\text{Figure 2.2: Communication topology.}\]

2.7.1 IMP-based attacks

Since the eigenvalues of \( A \) in (2.69) are \( \lambda_{1,2} = \pm i \), based on Definition 2, the attack signal \( f = 20 \sin(t) \) is an IMP-based attack. Let the attack signal be injected to Agent 1 (root node) at time \( t=20 \). The results are shown in Fig. 2.3. It can be seen that the compromised agent destabilizes the entire network. This result is consistent with Theorem 2. It is shown in Fig. 2.4 that the same IMP-based attack on Agent 5 (non-root node) cannot destabilize the entire network. However, Agent 4, which is the only agent reachable from Agent 5, does not synchronize to the desired consensus trajectory. Moreover, one can see in Fig. 2.5 that the local neighborhood tracking error converges to zero for all intact agents except the compromised Agent 5. These results are in line with Theorem 3. Fig. 2.6 shows the KL divergence in the presence of IMP based attack on Agent 5 grows for corrupted agent. This
result follows Theorem 4. Then, the effect of attack is rejected using the presented belief-based detection and mitigation approach in Theorem 4 and Theorem 6. Fig. 2.7 shows that reachable agents follow the desired consensus trajectory, even in the presence of the attack.

Figure 2.3: The state of agents when Agent 1 is under an IMP-based attack.

Figure 2.4: Agent 5 is under IMP-based attack. The state of agents.

Figure 2.5: Agent 5 is under IMP-based attack. The local neighborhood tracking error of agents.
2.7.2 Non-IMP-based attacks

The attack signal is assumed to be $f = 10 + 5\sin(2t)$. The effect of this attack on Agent 5 (non-root node) is shown in Fig. 2.8. It can be seen that this non-IMP-based attack on Agent 5 only affects the reachable Agent 4. Then, Fig. 2.9 shows that the KL divergence in the presence of non-IMP based attack on Agent 5 grows for compromised agent. It is shown
Figure 2.9: Divergence for state of agents when Agent 5 is under a non-IMP based attack.

Figure 2.10: The state of agents after attack detection and mitigation for non-IMP based attack.

in Fig. 2.10 that the effect of the attack is removed for the intact Agent 4 using belief-based detection and mitigation approaches presented in Theorems 5 and 6.

2.8 Conclusion

A resilient control framework has been introduced for DMASs. First, the effects of IMP-based and non-IMP-based attacks on DMASs have been analyzed using a graph-theoretic approach. Then, a KL divergence based criterion, using only the observed local information of agents, has been employed to detect attacks. Each agent detects its neighbors’ misbehaviors, consequently forming a self-belief about the correctness of its own information, and continuously updates its self-belief and communicates it with its neighbors to inform them about the significance of its outgoing information. Additionally, if the self-belief value of an agent is low, it forms beliefs on the type of its neighbors (intact or compromised) and, consequently, updates its trust of its neighbors. Finally, agents incorporate their neighbors’
self-beliefs and their own trust values in their control protocols to slow down and mitigate attacks.

2.9 Appendix

Proof of Lemma 5

Using (2.39), the KL divergence between error sequences $\varphi_i$ and $\tau_i$ can be written as

$$D_{KL}(\varphi_i || \tau_i) = \mathbb{E}_1[\log P_{\varphi_i} - \log P_{\tau_i}],$$

(2.70)

where probability density functions $P_{\varphi_i}$ and $P_{\tau_i}$ are defined in (2.37). Using (2.37), (2.70) and the logarithm property $\log(a + b) = \log(a) + \log(1 + b/a)$, one has

$$D_{KL}(\varphi_i || \tau_i) =$$

$$= \mathbb{E}_1[\log \left( \frac{1}{\sqrt{2\pi} |\sigma_{1i}|} e^{-\frac{(q_i - \mu_{1i})^2}{2\sigma^2_{1i}}} \right) - \log \left( \frac{1}{\sqrt{2\pi} |\sigma_{2i}|} e^{-\frac{(q_i - \mu_{2i})^2}{2\sigma^2_{2i}}} \right)]$$

$$+ \mathbb{E}_1[\log \left( 1 + e^{-\frac{2q_i\mu_{1i}}{\sigma^2_{1i}}} \right) - \log \left( 1 + e^{-\frac{2q_i\mu_{2i}}{\sigma^2_{2i}}} \right)],$$

(2.71)

The first term in (2.71) is a KL divergence formula for statistical sequences with normal Gaussian distribution which is given in [42] as

$$T_1 = \frac{1}{2} \left( \log \frac{\sigma^2_{2i}}{\sigma^2_{1i}} - 1 + (\sigma^2_{2i} - 2\sigma_{1i}^2) \right) + \frac{1}{2} \sigma^2_{2i} (\mu_{2i} - \mu_{1i})^2.$$  

(2.72)

The second term $T_2$ in (2.71), using power series expansion

$$\log(1 + a) = \sum_{n \geq 0} \left( (-1)^n a^{n+1} / (n + 1) \right)$$

and ignoring higher order terms, can be approximated as

$$T_2 \approx \mathbb{E}_1[e^{\frac{-2q_i\mu_{1i}}{\sigma^2_{1i}}} - (e^{\frac{-2q_i\mu_{1i}}{\sigma^2_{1i}}})^2] - \mathbb{E}_1[e^{\frac{-2q_i\mu_{2i}}{\sigma^2_{2i}}} - (e^{\frac{-2q_i\mu_{2i}}{\sigma^2_{2i}}})^2],$$  

(2.73)
which can be expressed as

\[ T_2 \approx \int_{-\infty}^{\infty} P_{\phi_i} e^{-\frac{2q_i \mu_{1i}}{\sigma_{1i}^2}} \, dq_i - \frac{1}{2} \int_{-\infty}^{\infty} P_{\phi_i} e^{-\frac{4q_i \mu_{1i}}{\sigma_{1i}^2}} \, dq_i \]

\[ - \int_{-\infty}^{\infty} P_{\phi_i} e^{-\frac{2q_i \mu_{2i}}{\sigma_{2i}^2}} \, dq_i + \frac{1}{2} \int_{-\infty}^{\infty} P_{\phi_i} e^{-\frac{4q_i \mu_{2i}}{\sigma_{2i}^2}} \, dq_i. \]  

(2.74)

Now, the first term of \( T_2 \) can be written as

\[ \int_{-\infty}^{\infty} P_{\phi_i} e^{-\frac{2q_i \mu_{1i}}{\sigma_{1i}^2}} \, dq_i \]

\[ = \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi} |\sigma_{1i}|} e^{-\frac{(q_i + \mu_{1i})^2}{2\sigma_{1i}^2}} \, dq_i + \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi} |\sigma_{1i}|} e^{-\frac{(q_i + 5\mu_{1i})^2 + 4q_i \mu_{1i}}{2\sigma_{1i}^2}} \, dq_i. \]  

(2.75)

Using the fact that density integrates to 1, (2.75) becomes

\[ \int_{-\infty}^{\infty} P_{\phi_i} e^{-\frac{2q_i \mu_{1i}}{\sigma_{1i}^2}} \, dq_i = 1 + e^{\frac{4\mu_{1i}^2}{\sigma_{1i}^2}}. \]  

(2.76)

Similarly, second term of \( T_2 \) can be written as

\[ \int_{-\infty}^{\infty} P_{\phi_i} e^{-\frac{4q_i \mu_{1i}}{\sigma_{1i}^2}} \, dq_i \]

\[ = -\frac{1}{2} \int_{-\infty}^{\infty} P_{\phi_i} e^{-\frac{8q_i \mu_{1i}}{2\sigma_{1i}^2}} \, dq_i + e^{\frac{24\mu_{1i}^2}{12\sigma_{1i}^2}} \, dq_i. \]  

(2.77)

which yields

\[ -\frac{1}{2} \int_{-\infty}^{\infty} P_{\phi_i} e^{-\frac{4q_i \mu_{1i}}{\sigma_{1i}^2}} \, dq_i = -\frac{1}{2} \left( e^{\frac{4\mu_{1i}^2}{\sigma_{1i}^2}} + e^{\frac{12\mu_{1i}^2}{\sigma_{1i}^2}} \right). \]  

(2.78)

The third term of \( T_2 \) is

\[ \int_{-\infty}^{\infty} P_{\phi_i} e^{-\frac{2q_i \mu_{2i}}{\sigma_{2i}^2}} \, dq_i \]

\[ = \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi} |\sigma_{2i}|} e^{-\frac{(q_i - \mu_{2i})^2}{2\sigma_{2i}^2}} \, dq_i + e^{\frac{2q_i \mu_{2i}}{2\sigma_{2i}^2}} + e^{\frac{2q_i \mu_{2i}}{2\sigma_{2i}^2}} \, dq_i. \]  

(2.79)
which can be written in the form

\[
\int_{-\infty}^{\infty} P_\varphi e^{-\frac{2qi\mu_1}{\sigma_1^2}} dq_i = -\int_{-\infty}^{\infty} \left( e^{-\frac{\rho_1^2 - \rho_1^2}{2\sigma_1^2}} \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi} |\sigma_1|} e^{-\frac{(q_i - \rho_1)^2}{2\sigma_1^2}} dq_i \right),
\]

where \( \rho_1 = (\mu_1 - 2\mu_2 \sigma_1^2 \sigma_2^2) \) and \( \rho_2 = (\mu_1 + 2\mu_2 \sigma_1^2 \sigma_2^2) \) which becomes

\[
-\int_{-\infty}^{\infty} P_\varphi e^{-\frac{2qi\mu_1}{\sigma_1^2}} dq_i = -\left( e^{-\frac{\rho_1^2 - \rho_1^2}{2\sigma_1^2}} + e^{-\frac{\rho_1^2 - \rho_2^2}{2\sigma_1^2}} \right).
\]

Similarly, the last term of \( T_2 \) can be simplified as

\[
\frac{1}{2} \int_{-\infty}^{\infty} P_\varphi e^{-\frac{2qi\mu_1}{\sigma_1^2}} dq_i = \frac{1}{2} \left( e^{-\frac{\rho_2^2 - \rho_3^2}{2\sigma_1^2}} + e^{-\frac{\rho_2^2 - \rho_4^2}{2\sigma_1^2}} \right),
\]

where \( \rho_3 = (\mu_1 - 4\mu_2 \sigma_1^2 \sigma_2^2) \) and \( \rho_4 = (\mu_1 + 4\mu_2 \sigma_1^2 \sigma_2^2) \). Adding (2.76), (2.78), (2.81) and (2.82), \( T_2 \) can be written as

\[
T_2 \approx e^{-\frac{\rho_1^2}{2\sigma_1^2}} \left( \frac{1}{2} \left( e^{\frac{\rho_3^2}{2\sigma_1^2}} + e^{\frac{\rho_2^2}{2\sigma_1^2}} \right) \right) - \left( e^{\frac{\rho_1^2}{2\sigma_1^2}} + e^{\frac{\rho_2^2}{2\sigma_1^2}} \right) + 1 + \frac{1}{2} e^{\frac{8\mu_1^2}{\sigma_1^2}} \left( 1 - e^{-\frac{\rho_1^2}{2\sigma_1^2}} \right). \]

Now, using (2.72)-(2.73) and (2.83), one gets (2.39). This completes the proof.
CHAPTER 3
DETECTION AND MITIGATION OF DATA MANIPULATION ATTACKS
IN AC MICROGRIDS

3.1 Introduction

In the previous chapter, we analyzed the adverse effect of attacks and designed resilient
distributed control mechanism for DMASs with guaranteed performance and consensus un-
der mild assumptions. This chapter validates the effectiveness of the developed approach in
Chapter 2 by applying it to distributed frequency and voltage synchronization of AC micro-
grids under data manipulation attacks. The attack detection mechanism deploys Kullback-
Liebler (KL) divergence to measure the discrepancy between the Gaussian distributions of
the actual and expected local frequency/active power and voltage/reactive power neighbor-
hood tracking errors. To mitigate the negative impact of attack, a self-belief value, as an
indication of the probability of presence of attacks on neighbors of an agent, is presented
for each distributed energy resource (DER) by utilizing KL-based detectors. The self-belief
value is a measure of trustworthiness of the DER’s own outgoing information and is transmit-
ted to neighboring DERs. Moreover, the trustworthiness of the incoming information from
neighboring DERs is estimated using a trust factor. Trust for individual DERs is developed
based on the relative entropy between DER own information and its neighbor’s informa-
tion on the communication graph. The attack mitigation algorithm utilizes self-belief and
trust values to modify distributed control protocols. Finally, the performance of proposed
resilient frequency and voltage control techniques is verified through simulation of microgrid
tests system and hardware-in-the-loop (HIL) set-up using Opal-RT as a real-time digital
simulator.
3.2 Preliminaries

The communication network of a microgrid can be modeled by a graph. DERs are considered as the nodes of the communication graph and the communication links are considered as the edges. A graph is usually expressed as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$ with a nonempty finite set of $N$ nodes $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$, a set of edges or arcs $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$, and the associated adjacency matrix $\mathcal{A} = [a_{ij} \in \mathbb{R}^{N \times N}]$. $a_{ij}$ is the weight of edge $(v_j, v_i)$, and $a_{ij} > 0$ if $(v_j, v_i) \in \mathcal{E}$, otherwise $a_{ij} = 0$. The set of neighbors of node $i$ is denoted as $N_i = \{j| (v_j, v_i) \in \mathcal{E}\}$. The in-degree matrix is defined as $D = \text{diag}\{d_i\} \in \mathbb{R}^{N \times N}$ with $d_i = \sum_{j \in N_i} a_{ij}$. The Laplacian matrix is defined as $L = D - \mathcal{A}$ [80].

Assumption 1. The communication graph $\mathcal{G}$ has a spanning tree.

3.3 Conventional Distributed Secondary Control

In the microgrid hierarchical control structure, the primary control level maintains the voltage and frequency stability of the microgrid. The secondary control level restores the microgrid voltage and frequency to their nominal values. DERs are integrated to the rest of microgrid through Voltage Source Inverters (VSI). Depending on the control objectives, DERs can be of two main types, namely grid forming and grid following. Grid forming DERs utilize a Voltage Controlled VSI (VCVSI) and have the capability of dictating microgrid frequency and voltage. On the other hand, grid following DERs utilize a Current Controlled VSI (CCVSI) and follow the microgrid frequency and voltage while supplying a specific amount of active and reactive power based on external set points [54].

The primary control is locally implemented at grid forming DERs by the droop technique. This technique prescribes a relation between the frequency, $\omega_i$, and the active power, and between the voltage magnitude, $v_{o,magi}$, and the reactive power. The frequency and voltage droop characteristics are

$$\begin{align*}
\omega_i &= \omega_{ni} - m_i P_i \\
v_{o,magi} &= V_{ni} - n_i Q_i
\end{align*}$$

(3.1)
where $\omega_{ni}$ and $V_{ni}$ are the primary frequency and voltage control references and $m_{Pi}$ and $n_{Qi}$ are the active and reactive power droop coefficients, respectively. Conventionally, the active power droop coefficients are proportionally selected based on the apparent power rating of DERs. However, the reactive power droop coefficients are proportionally selected based on the maximum reactive power which is calculated using a minimum allowable power factor and apparent power rating of DER [46]. The apparent power rating is related to thermal rating of DER equipment (e.g., power electronics switches).

The objective of distributed secondary control is to mitigate the microgrid frequency and voltage deviations from their nominal values which are caused by primary control. Distributed secondary control utilizes distributed control protocols implemented on individual DERs that can communicate with each other through a distributed communication network and share their local information with neighboring DERs.

**Problem 1:** The distributed secondary control chooses $\omega_{ni}$ and $V_{ni}$ in (3.1) such that the operating frequency and terminal voltage magnitude of each DER synchronize to the reference frequency and voltage, $\omega_{ref}$ and $v_{ref}$, i.e.,

\[
\left\{ \begin{aligned}
\lim_{t \to \infty} \| \omega_i(t) - \omega_{ref} \| &= 0 \\
\lim_{t \to \infty} \| v_{o, magi}(t) - v_{ref} \| &= 0 
\end{aligned} \right. \forall i \in N. \tag{3.2}
\]

Moreover, the secondary control should guarantee the allocation of active and reactive power of DERs based on the droop coefficients [51]-[55] as

\[
m_{Pi} P_i = m_{Pj} P_j, \tag{3.3}
\]

\[
n_{Qi} Q_i = n_{Qj} Q_j, \tag{3.4}
\]

where $P_{\text{maxi}}/Q_{\text{maxi}}$ and $P_{\text{maxj}}/Q_{\text{maxj}}$ are the active and reactive power ratings of $i$–th and $j$–th DER, respectively.

The secondary control of a microgrid including $N$ DERs is described as the synchronization problem for the following first-order multi-agent system to adjust the primary control inputs

\[
\left\{ \begin{aligned}
\dot{\omega}_{ni} &= v_{\omega i}, \\
\dot{V}_{ni} &= v_{vi}, 
\end{aligned} \right. \quad i = 1, ..., N, \tag{3.5}
\]
where \( v_{\omega i} \) and \( v_{vi} \) are the distributed secondary frequency and voltage control (DSFC and DSVC) protocols that are chosen based on the local information of each DER and neighboring DERs’ information and can be written as [54]

\[
v_{\omega i} = -c_\omega \delta_{\omega i}, \tag{3.6}
\]

\[
v_{vi} = -c_v \delta_{vi}, \tag{3.7}
\]

where \( c_\omega \) and \( c_v \) are the control gains; \( \delta_{\omega i} \) and \( \delta_{vi} \) are the local frequency and voltage neighborhood tracking errors that can be written as

\[
\delta_{\omega i} = \sum_{j \in N_i} a_{ij}(\omega_i - \omega_j) + g_i(\omega_i - \omega_{ref}) + \sum_{j \in N_i} a_{ij}(m_{P_i}P_i - m_{P_j}P_j),
\]

\[
\delta_{vi} = \sum_{j \in N_i} a_{ij}(v_{o,magi} - v_{o,magj}) + g_i(v_{o,magi} - v_{ref}) + \sum_{j \in N_i} a_{ij}(n_{Q_i}Q_i - n_{Q_j}Q_j),
\]

The pinning gain \( g_i \) is assumed nonzero for only one DER.

**Remark 1.** Note that there always exists a low-level communication noise in the network of DERs. Therefore, in the presence of the communication noise, one can write the auxiliary controls \( v_{\omega i} \) and \( v_{vi} \) of \( i \)-th DER in (3.6) and (3.7) as

\[
\begin{align*}
\zeta_{\omega i} &= v_{\omega i} + \eta_{\omega i}, \\
\zeta_{vi} &= v_{vi} + \eta_{vi},
\end{align*}
\tag{3.10}
\]

where \( \eta_{\omega i} \sim \mathcal{N}(0, \Sigma_{\omega i}) \) and \( \eta_{vi} \sim \mathcal{N}(0, \Sigma_{vi}) \), respectively, denote the aggregate Gaussian noise affecting the incoming neighbors’ frequency and voltage to \( i \)-th DER. In general, the noise associated with electronic devices at the receiver end lies under the category of thermal noise and statistically modeled as Gaussian, thus we assumed communication noise to be Gaussian and it is a standard assumption in the literature [56]. In noisy scenarios, the synchronization problem for microgrid frequency and voltage as defined in Problem 1 changes to the mean square synchronization problem
and becomes
\[
\begin{align*}
\lim_{t \to \infty} E \left\| \omega_i(t) - \omega_{ref}(t) \right\|^2 &= 0 \\
\lim_{t \to \infty} E \left\| v_{o, magi}(t) - v_{ref}(t) \right\|^2 &= 0
\end{align*}
\]
(3.11)

### 3.4 Attack Modeling and Detection Mechanism

This section presents attack modelling and detection mechanism for the distributed secondary control of microgrid.

**Definition 1. (Compromised DER).** A DER that is directly under attack is called a compromised DER.

**Definition 2. (Intact DER).** A DER that is not compromised or not under direct attack is called an intact DER.

#### 3.4.1 Attack Modeling

For the direct attack on controller, one can model the DER’s frequency as
\[
\omega_{cr}^i = \omega_i + \gamma_i \omega^a_i
\]
(3.12)
with \(\omega^a_i\) as the injected attacker’s input into the controller of \(i\)-th DER and \(\omega_{cr}^i\) denotes the corrupted DER frequency with scalar \(\gamma_i\) equal to 1 in the presence of attack. Similarly, for the attack on the communication channel between two DERs, one can model the received corrupted frequency signal from \(j\)-th DER as
\[
\omega_{cr}^j = \omega_j + \gamma_j \omega^a_j
\]
(3.13)
where \(\omega^a_j\) represents the injected attacker’s input into the communication channel between two DERs and \(\omega_{cr}^j\) denotes the corrupted DER frequency of neighbor \(j\) received at \(i\)-th DER with scalar \(\gamma_i\) equal to 1 in the presence of attack.

**Remark 2.** This subsection discusses the attack model in terms of DER’s frequency which affects the auxiliary control \(v_{\omega i}\) in (3.6). Moreover, the rest of the chapter considers frequency-based attacks and presents attack detection and mitigation mechanisms. Without loss of generality, the same approach holds true for attack modelling, detection and mitigation mechanisms for voltage-based attacks.
Remark 3. Attack models in (3.12)-(3.13) represent frequency manipulation attacks on controllers. Due to the extensive deployment of communication and control technologies and the presence of Intelligent Electronic Devices (IEDs), the microgrid control system is highly vulnerable to cyber-attacks. An attack tree for FDI threat analysis is provided to illustrate the attack path. As seen, the FDI attack can tamper with either the sensors (e.g., Phasor Measurement Units (PMUs)) or actuators (control and decision-making units). Such attacks can be launched by injecting counterfeit attack signals into sensors of DER measurement units or directly by injecting a disturbance into the control units and even hijacking the entire controller. More specifically, FDI attacks on DERs can endanger microgrid voltage and frequency stability, slow down the DER control system responses, or overload DERs.

The existing firewall/intrusion detection systems (IDSs) monitor and analyze information flow in the network and detect if there exists considerable change in the information flow. However, there is no single IDS that is able to detect all different attack types [81]. Moreover, the IDSs' effectiveness highly depends on their parameters. So, if the IDS parameters are not fine-tuned, the possibility of not detecting attacks increases [81]. On the other hand, IDSs do not block the corrupted information and cannot mitigate attacks. Therefore, it is of vital importance to design a resilient control protocol for microgrids that can mitigate attacks and ensure an acceptable level of functionality for microgrid despite attacks.

3.4.2 Attack Detection Mechanism

This subsection presents a relative entropy-based attack detection approach for the distributed secondary control of microgrid. More specifically, KL divergence, a non-negative measure of the relative entropy between two probability distributions is employed to measure the discrepancy between them.

**Definition 3. (KL divergence)** [41], [82] Let $X$ and $Z$ be two random sequences with probability density function $P_X$ and $P_Z$, respectively. The KL divergence measure between $P_X$ and $P_Z$ in continuous-time is defined as

$$D_{KL}(X||Z) = \int P_X(\theta) \log \left( \frac{P_X(\theta)}{P_Z(\theta)} \right) d\theta$$

(3.14)
If the sequences $X$ and $Z$ are Gaussian distributed, then the KL divergence in (3.14) can be simplified in the terms of mean and covariance of sequences as [41]

$$D_{KL}(X||Z) = \frac{1}{2} \left( \log \frac{|\Sigma_Z|}{|\Sigma_X|} - n + tr(\Sigma_Z^{-1}\Sigma_X) \right)$$

$$+ \frac{1}{2} (\mu_Z - \mu_X)^T \Sigma_Z^{-1} (\mu_Z - \mu_X)$$

where $\mu_X$ and $\Sigma_X$ denote the mean and covariance of sequence $X$, and $\mu_Z$ and $\Sigma_Z$ denote the mean and covariance of sequence $Z$. Moreover, $n$ denotes the dimension of the sequences.

For the design of an attack detector, we first rewrite the frequency auxiliary control $\zeta_{\omega i}$ in (3.10) with statistical properties and then present an attack detection mechanism based on the KL divergence measure for distributed secondary control of AC microgrids. We show that in the presence of an attack, one can identify different sophisticated attacks based on the change in the statistical properties of the auxiliary control variables. In the absence of attack, since we consider the Gaussian noise in the communication channel, then the auxiliary control $\zeta_{\omega i}$ in (3.10) can be written as

$$\zeta_{\omega i} = -c_\omega \delta_{\omega i} + \eta_{\omega i}$$

(3.16)

where $\eta_{\omega i}$ denotes the aggregate Gaussian noise affecting the incoming neighbors’ information given by

$$\eta_{\omega i} = \sum_{j \in N_i} a_{ij} \eta_{\omega ij} \sim \mathcal{N}(0, \Sigma_{\omega i})$$

(3.17)

Due to presence of noise, the statistical properties of the auxiliary control $\zeta_{\omega i}$ in (3.10) becomes

$$\zeta_{\omega i} \sim \mathcal{N}(0, \Sigma_{\omega i})$$

(3.18)

and it represents the nominal behavior of the DSFC.

In the presence of attacks, using (3.10), the auxiliary control $\zeta_{a\omega i}$ becomes

$$\zeta_{a\omega i} = -c_\omega \delta_{cr\omega i} + \eta_{\omega i}$$

(3.19)

with the corrupted local neighborhood tracking error $\delta_{cr\omega i} = \delta_{\omega i} + f_i$ where

$$f_i = [\sum_{j \in N_i} a_{ij} + g_i] \omega_i^a - \sum_{j \in N_i} a_{ij} \omega_j^a]$$

(3.20)
denotes the overall deviation in the local neighborhood tracking error due to the attacks on controller/communication channel in the network. Note that in presence of attacks, one can observe the corrupted frequency of DERs and based on corrupted frequency, one has the corrupted auxiliary control \( \zeta^a \omega_i \). The overall attacker’s input \( f_i \) is neither measurable nor required to be known. The statistical properties of corrupted control protocol changes due to the effect of attacks. Now, from (3.19), one has the following statistical properties

\[
\zeta^a \omega_i \sim \mathcal{N}(\mu_{f_i}, \Sigma_{f_i} + \Sigma_{\omega_i})
\]

(3.21)

where \( \mu_{f_i} \) and \( \Sigma_{f_i} \) are mean and covariance of the injected overall attack signal \( f_i \), respectively. Since both \( \zeta^a \omega_i \) and \( \zeta_{\omega_i} \) have normal Gaussian distributions, according to (3.15) the KL divergence \( D_{KL}(\zeta^a \omega_i || \zeta_{\omega_i}) \) between control sequences \( \zeta^a \omega_i \) and \( \zeta_{\omega_i} \) becomes

\[
D_{KL}(\zeta^a \omega_i || \zeta_{\omega_i}) = \frac{1}{2} \left( \log \left| \frac{\Sigma_{\omega_i}}{\Sigma_{a \omega_i}} \right| - 1 + tr(\Sigma_{a \omega_i}^{-1} \Sigma_{\omega_i} \Sigma_{a \omega_i}^{-1}) \right)
\]

\[
+ \frac{1}{2} (\mu_{\zeta_{\omega_i}} - \mu_{\zeta^a \omega_i})^T \Sigma_{\omega_i}^{-1} (\mu_{\zeta_{\omega_i}} - \mu_{\zeta^a \omega_i})
\]

(3.22)

where \( \mu_{\zeta_{\omega_i}} \) and \( \Sigma_{\zeta_{\omega_i}} \) denote the mean and covariance of \( \zeta_{\omega_i} \) and \( \mu_{\zeta^a \omega_i} \) and \( \Sigma_{\zeta^a \omega_i} \) denote the mean and covariance of \( \zeta^a \omega_i \).

We define the average of KL divergence over a window \( T \) as

\[
\Omega_i = \frac{1}{T} \int_{t_k}^{t_{k+T-1}} D_{KL}(\zeta^a \omega_i || \zeta_{\omega_i}) d\tau
\]

(3.23)

to detect the change due to the adversarial input. Now, in the following theorem, we show that the effect of attacks in the secondary distributed control of the microgrid can be detected based on the discrepancy of the control sequences \( \zeta^a \omega_i \) and \( \zeta_{\omega_i} \).

**Theorem 1.** Consider the distributed auxiliary control \( \zeta_{\omega_i} \) in (3.16) under attacks. Then, a) \( \Omega_i \) defined in (3.23) becomes zero, if there is no attack on DERs. b) \( \Omega_i \) defined in (3.23) is greater than a design threshold \( \gamma_i \), if the microgrid secondary control is under attack.

**Proof.** In the absence of attacks, the statistical properties of sequences \( \zeta^a \omega_i \) and \( \zeta_{\omega_i} \), respectively, in (3.18) and (3.21) are the same because \( \mu_{f_i} \) and \( \Sigma_{f_i} \) become zero as \( f_i = 0 \). Therefore, the KL
divergence $D_{KL}(\zeta_{\omega_i}^a || \zeta_{\omega_i})$ in (3.22) becomes zero based on (3.15) which yields $\Omega_i$ in (3.23) to be zero. This complete the proof of part (a).

For the proof of Part (b), using (3.18)-(3.21) in (3.22), the KL divergence between $\zeta_{\omega_i}^a$ and $\zeta_{\omega}$ becomes

$$D_{KL}(\zeta_{\omega_i}^a || \zeta_{\omega_i}) = \frac{1}{2}(\log \frac{\|\Sigma_{\omega_i}\|}{\|\Sigma_{f_i} + \Sigma_{\omega_i}\|} + tr((\Sigma_{\omega_i}^{-1}\Sigma_{f_i}) + \mu_f^T \Sigma_{\omega_i}^{-1} \mu_f))$$  (3.24)

Then, using (3.23), one has

$$\Omega_i = \frac{1}{T} \int_k^{k+T-1} \left(\frac{1}{2} \log \frac{\|\Sigma_{\omega_i}\|}{\|\Sigma_{f_i} + \Sigma_{\omega_i}\|} + tr((\Sigma_{\omega_i}^{-1}\Sigma_{f_i}) + \mu_f^T \Sigma_{\omega_i}^{-1} \mu_f))d\tau > \gamma_i$$  (3.25)

where $T$ and $\gamma_i$ denote the sliding window size and the predefined positive design threshold, respectively. This completes the proof.

Based on the presented Theorem 1, effect of attacks on the distributed secondary control of microgrids can be detected using the predefined design threshold $\gamma_i$. Attack detection in (3.25) uses the idea of average over a fixed length moving window to avoid false detection. If there is a short-period anomaly rather than attack (such as disturbance or packet dropout), it vanishes in a few time steps and such anomalies are not detected as attacks.

### 3.5 Resilient Distributed Control Mechanism

This section presents a resilient distributed control mechanism for distributed secondary control of microgrids based on the proposed attack detection algorithm in the previous section.
Figure 3.1: The flowchart of proposed attack detection and mitigation approach.
To this end, first, we introduce the notion of self and external-belief of DERs about trustworthiness of their own information and their neighbor’s information, respectively. Then the presented beliefs are incorporated in the distributed secondary control protocols.

### 3.5.1 Belief of DERs About Their Own Observed Frequency

To measure the level of trustworthiness of each DER about its own observed frequency, which depends on the proximity to the source of the attack in the network, a self-belief is presented. In the presence of the adversary, a DER reduces its level of trustworthiness about its own observed frequency and transmits its self-belief to its immediate neighbors which prevent the propagation of attack in the microgrid.

Using the $D_{KL}(\zeta^{a}_{\omega i}||\zeta_{\omega i})$ from Theorem 1, self-belief of $i-th$ DER about its own observed frequency is defined as

$$I_{Bel}^{i}(t) = \kappa_{1} \int_{0}^{t} e^{\kappa_{1}(\tau-t)} \psi_{i}(\tau)d\tau$$ (3.26)

where $0 \leq I_{Bel}^{i}(t) \leq 1$ with

$$\psi_{i}(t) = \frac{\Delta_{1}}{\Delta_{1} + D_{KL}(\zeta^{a}_{\omega i}||\zeta_{\omega i})}$$ (3.27)

where $\Delta_{1}$ represents the threshold to account for the channel fading and other uncertainties and $0 < \kappa_{1} < 1$ denotes the discount factor. Equation (3.26) can be implemented by the following differential equation

$$\dot{I}_{Bel}^{i}(t) + \kappa_{1}I_{Bel}^{i}(t) = \kappa_{1}\psi_{i}(t)$$ (3.28)

Based on Theorem 1, in the presence of attacks, $D_{KL}(\zeta^{a}_{\omega i}||\zeta_{\omega i}) \gg \Delta_{1}$, which makes the self-belief of the DER $\psi_{i}(t)$ close to zero and, consequently, the value of $I_{Bel}^{i}(t)$ becomes close to zero. On the other hand, based on Theorem 1, in the absence of attack $D_{KL}(\zeta^{a}_{\omega i}||\zeta_{\omega i})$ tends to zero, which makes $\psi_{i}(t)$ close to one and, consequently, $I_{Bel}^{i}(t)$ becomes close to one.

If a DER is under direct attack, its self-belief tends to zero according to (3.26). The DER transmits its self-belief value to the neighboring DERs. Using the received self-belief values, neighboring DERs ignore the information received from the attacked DER which prevents the attack propagation. Note that the discount factor in (3.26) evaluates the importance of current information with regards to past information. The discount factor ensures that if an attacker removes the
effect of attack in a while, or if a short-period adversarial effect exists rather than attack (such as packet dropout), then the belief of the DER will be recovered, as it mainly depends on the current information.

3.5.2 Belief of DERs About Their Neighbor’s Observed Frequency

To evaluate the level of confidence of a DER on its neighbor’s observed frequency, we introduce the notion of external-belief or trust. If the self-belief value of a DER is low, it forms beliefs on its neighboring DER’s information (either intact or compromised) and updates its external-belief which depends on the beliefs on each of its neighbors using only local information. Therefore, the DERs can identify the compromised neighbor and discard its information in their control protocol.

In the worst-case scenario, a compromised DER always transmits the self-belief value of 1 to its neighbors to deceive them. Based on the external-belief a DER can identify the corrupted neighbors and discards their information.

Using the KL divergence between exchanged information of the i-th DER and its neighbor, one can define the $\Upsilon_{ij}(t)$ as

$$\Upsilon_{ij}(t) = \kappa_2 \int_0^t e^{\kappa_2(t-\tau)} \chi_{ij}(\tau) d\tau$$  \hspace{1cm} (3.29)

where $0 \leq \Upsilon_{ij}(t) \leq 1$ and

$$\chi_{ij}(t) = \frac{\Delta_2}{\Delta_2 + D_{KL}(\omega_i||m_i)} \forall j \in N_i$$  \hspace{1cm} (3.30)

with $m_i = (1/|N_i|) \sum_{j \in N_i} \omega_j$; $\Delta_2 > 0$ represent the threshold to account for the channel fading and other uncertainties; $0 < \kappa_2 < 1$ denotes the discount factor. For the neighboring DER under direct attack, the KL divergence $D_{KL}(\omega_i||m_i)$ becomes high which makes $\chi_{ij}(t)$ close to zero. Consequently, this makes the value of $\Upsilon_{ij}(t)$ close to zero. On the other hand, if the incoming information from neighboring DER is intact, then $D_{KL}(\omega_i||m_i)$ becomes close to zero which makes $\chi_{ij}(t)$ close to one. Equation (3.29) can be implemented using the following differential equation

$$\dot{\Upsilon}_{ij}(t) + \kappa_2 \Upsilon_{ij}(t) = \kappa_2 \chi_{ij}(t)$$  \hspace{1cm} (3.31)

Now, we define the external-belief value of a DER on its neighbors as

$$E_{ij}^{Bel}(t) = \min(I_i^{Bel}(t), \Upsilon_{ij}(t))$$  \hspace{1cm} (3.32)
with \(0 \leq E_{ij}^{Bel}(t) \leq 1\). Note also that the discount factor in (3.26) and (3.29) determines how much we value the current experience with regards to past experiences. It also guarantees that if the attack is not persistent and disappears after a while, or if a short-period adversary rather than attack (such as disturbance or packet dropout) causes, the belief will be recovered, as it mainly depends on the current circumstances.

### 3.5.3 The Mitigation Mechanism Using Self and External-belief values

This subsection presents a resilient or cyber-secure auxiliary control protocol for secondary control of microgrid. We employ the entropy-based self and external-belief values in the mitigation algorithm (See Fig. 3.1). More specifically, both self and external-belief values in (3.26) and (3.32) are incorporated into the frequency based auxiliary control in (3.10) and the resilient form is presented as

\[
\zeta_{\omega i} = -c_\omega \left( \sum_{j \in N_i} \alpha_{ij}(t)(\omega_i - \omega_j) + g_i(\omega_i - \omega_{ref}) \right) + \sum_{j \in N_i} \alpha_{ij}(t)(m_PiP_i - m_PjP_j) + \eta_{\omega i}
\]  

(3.33)

where

\[
\alpha_{ij}(t) = a_{ij}I_{i}^{Bel}(t)E_{ij}^{Bel}(t)
\]  

(3.34)

incorporates the self and external-belief discussed in the previous subsection. The following theorem solves Problem 1 using proposed resilient auxiliary control protocol in (3.33) for intact DERs in the presence of attack.

**Assumption 2 (m-local connectivity).** If at most \(m\) neighbors of each intact DER is under attack, at least \(m + 1\) neighbors of each intact DER are intact [82].

**Remark 4.** Assumption 2 is a common assumption in the distributed control literature [16], [82]. This assumption provides a minimum requirement for any distributed system to ensure consensus in the presence of attack.

**Theorem 2.** Consider the resilient DSFC in (3.33). Let Assumptions 1 and 2 be satisfied. Then, the frequency of the intact DERs synchronizes to the desired nominal frequency in mean square sense, despite the \(m\) compromised DERs.
Proof. The resilient frequency based secondary control in (3.33) can be rewritten as

$$\zeta_{\omega_i} = -c_{\omega i} (\sum_{j \in N_i} \alpha_{ij}(t)(\omega_i - \omega_j) + g_i(\omega_i - \omega_{ref})$$

$$+ \sum_{j \in N_i} \alpha_{ij}(t)(m_i P_i - m_j P_j)) + \eta_{\omega_i}$$

(3.35)

where the weight $\alpha_{ij}(t)$ defined in (3.34) combines the self-belief of agent $i$ and its external belief on agent $j$. The global form of the (3.35) becomes

$$\zeta_{\omega} = -c_{\omega}((L(t) + G)(\omega - \bar{\omega}_{ref}) + L(t)\bar{P}) + \eta_{\omega}$$

(3.36)

where $\omega = [\omega_1, \ldots, \omega_N]^T$, $\omega_{ref} = 1_N \otimes \bar{\omega}_{ref}$, $\eta_{\omega} = [\eta_{\omega 1}, \ldots, \eta_{\omega N}]^T$, $\zeta_{\omega} = [\zeta_{\omega 1}, \ldots, \zeta_{\omega N}]^T$ and $\bar{P} = [m_1 P_1, \ldots, m NP_N]^T$. Moreover, $L(t) \in \mathbb{R}^{N \times N}$ and $G \in \mathbb{R}^{N \times N}$ denote the graph Laplacian matrix and the diagonal gain matrix, with diagonal entries equal to the pinning gains $g_i$, respectively.

According to Assumption 2, the total number of the compromised agents is less than half of the network connectivity, i.e., $2m + 1$. Therefore, even if $m$ neighbors of an intact DER are attacked and collude to transmit the same value to mislead the intact DER, there still exists $m + 1$ intact neighbors that transmit the actual values which differ from the compromised ones. Moreover, since $m + 1$ intact DER’s neighbors are intact, it can update its external belief and isolate the compromised neighbors. As shown in [20], the resulting graph after isolating the compromised DERs in the entire network remains connected to the intact DERs. Therefore, there exists a spanning tree in the graph associated with all intact DERs. On the other hand, it is shown in [83]-[84], that distributed agents reach mean square consensus in the presence of Gaussian noise if the graph contains spanning tree. Thus, resilient DSFC in (3.33) intact DERs synchronize to the nominal frequency or the leader’s state. This completes the proof.

Remark 5. Note that even in the presence of replay attacks where the attacker replicates all the statistical characteristics of previous control signals for the DER, intact DERs lose their trust on the compromised DER’s due to the divergence term in calculating the external-belief in (3.30) and reject the corrupted information in their control protocol.
Remark 6. Although not considered in this chapter, the proposed cyber-secure distributed secondary control can be effectively integrated into the event-triggered based distributed controls (e.g., [64]) to increase the resilience of control system with respect to both FDI and DoS attacks.
3.6 Case Studies

3.6.1 Case A: Simulation results for IEEE 34-bus feeder

The microgrid test system is illustrated in Fig. 3.2. The IEEE 34 bus test feeder is utilized as the backbone of microgrid with six DERs integrated to different locations. This microgrid system is simulated in MATLAB/Simulink. The specification of lines is provided in [85]. A balanced feeder model by averaging the line parameters is utilized in the test system. Tables I and II summarize the specifications of the loads and DERs, respectively. The nominal frequency and line-to-line voltage are set to 60 Hz and 24.9 kV, respectively. DERs are connected to the feeder through six Y-Y, 480 V/24.9 kV, 400 kVA transformers with the series impedance of $0.03 + j 0.12$ pu. The communication graph of distributed secondary control system is depicted in Fig. 3.3. Only DER 1 knows the frequency and voltage reference values with the pinning gain $g_1 = 1$. The control gains $c_\omega$ and $c_v$ in (3.6) and (3.7) are set to 40. We assume zero-mean Gaussian communication noise with following statistical properties $N(0, 0.01)$. Two different cases are considered to evaluate the presented results for attack detection and mitigation in the distributed secondary control of microgrids. Case A.1 analyzes the results for DSFC and Case A.2 presents the results for DSVC in the presence of attacks in the microgrid.

Case A.1.1 (effect of attack on the conventional DSFC): In this case, we consider the attack on DER 6 based on (3.12). At $t = 0$, the microgrid is islanded from the main grid. From $t = 0$ to $t = 0.6$ s only the primary control is applied. The primary control takes action to provide frequency stability in the islanded However, the primary control only maintains frequency in stable ranges and cannot maintain frequency at exactly 60 Hz. Then, the secondary distributed frequency control is applied at $t = 0.6$ s to restore the microgrid frequency to 60 Hz. However, the attacker hijacks the DSFC of DER 6 and replaces the actual frequency with 60.2 Hz. Fig. 3.4 show that the conventional DSFC protocol leads to the loss of desired consensus. The frequency of each DER deviates from the desired frequency of 60 Hz and shows oscillatory behavior. In the presence of attack, the behavior of the compromised DER 6 is directly affected by the attack signal and its corrupted frequency is observed by reachable intact DERs which are affected by it and they also show oscillatory behaviors as shown in Fig. 3.4. Fig. 3.5 clearly shows that the relative entropy of compromised and reachable
DERs diverge and go beyond the predefined design threshold which is assumed to be in the presence of attack. The relative entropy of compromised DER is relatively much higher than the intact DERs and designed detector can easily detect the effect of attack.

*Case A.1.2 (attack detection and mitigation):* Similar to Case A.1.1, at $t = 0$, the microgrid is
islanded from the main grid. From \( t = 0 \) to \( t = 0.6 \) s only primary control is applied. Then, the secondary distributed frequency control is applied at \( t = 0.6 \) s to restore the microgrid frequency to 60 Hz. However, the attacker hijacks the DSFC of DER 6 and replaces the actual frequency with 60.2 Hz at \( t = 0.6 \) s and then, the designed attack detection and mitigation mechanism is applied at \( t = 0.7 \) s. As shown in Fig. 3.6, frequency of intact DERs restores to 60 Hz after applying the attack mitigation mechanism at \( t = 0.7 \) s. Active power of all DERs are also retrieved back as in intact mode. After applying the resilient DSFC in (3.33), intact DERs discard the frequency value received from corrupted DER and the mean and variance of their local frequency neighborhood tracking error distribution remain close to the normal case. Therefore, based on (3.22), the relative entropy for intact DERs remains close to zero but it keeps growing for the compromised DER 6 due to deviation in mean and variance of the corrupted frequency signal from the nominal one as shown in Fig. 3.7. According to (3.26)-(3.27), self-belief of a DER depends on its relative entropy and one can see in Fig. 3.7 that self-belief for all DERs becomes one except for the compromised DER 6, which indicates that all the DERs are confident about their frequencies, except for the compromised one. The self-belief of a DER measures the level of trustworthiness about its observed frequency, which is updated in each iteration and recursively used in resilient DSFC in (3.33) for mitigation of the attack. Based on the presented resilient DSFC, intact DERs do not incorporate the corrupted frequency from DER 6 and achieve the desired synchronization as shown in Fig. 3.6.

Case A.1.3 (attack detection and mitigation for periodic adversaries): In this subsection, the effectiveness of the presented attack detection and mitigation algorithm is validated for periodic attacks. The secondary distributed frequency control is applied at \( t = 0 \) s which synchronizes the frequency of the microgrid to 60 Hz. Then, the attacker hijacks the DSFC of DER 6 and replaces the actual frequency with 60.2 Hz at \( t = 1.2 \) s and \( t = 2.2 \) s. In the following, the simulation results are provided for two different attack durations. First, it is assumed that when the attack is applied at \( t = 1.2 \) s and \( t = 2.2 \) s, it is only effective for 0.05 s. Fig. 3.8 shows the DER frequencies and active power ratios. As seen in Fig. 3.8, due to the short duration of attack, its impact is minimal; DER frequencies slightly deviate from 60 Hz.

Based on (22), the relative entropy for intact DERs remains close to zero, but it keeps growing for the compromised DER 6 when the attack is effective due to deviation in mean and variance of the
According to (3.26)-(3.27), the self-belief of a DER depends on its relative entropy; one can see in Fig. 3.9 that self-belief values during the attack period are one for all DERs except for the compromised DER 6, which indicates that all the DERs are confident about their exchanged frequencies, except for the compromised one. As expected, for the time interval that the attacker turns off its attack signal, DER frequencies and active power ratios are restored to their intact values before the attack is applied.

In the second simulation scenario, it is assumed that when the attack is applied at $t = 1$ s and $t = 3$ s, it is effective for 0.5 s. Fig. 3.10 shows the DER frequencies and active power ratios. As seen in Fig. 3.10, after the attack is applied, DER frequencies deviate from 60 Hz and active power ratios experience noticeable oscillations. The relative entropy for intact DERs remains close to zero, but it keeps growing for the compromised DER 6 during durations of attack as shown in Fig. 3.11 (a). As seen in Fig. 3.11(b), self-belief during attack periods becomes one for all DERs except for the compromised DER 6. The attack mitigation scheme restores DER frequencies to 60 Hz and active power ratios to a common value. For the time interval that the attacker turns off its attack
signal, DER frequencies and active power ratios are restored to their intact values before the attack is applied.

Case A.2.1 (effect of attack on the conventional DSVC): In this case, we consider the attack on DER 6 based on (3.12). From $t = 0$ to $t = 0.65$ s only primary control is applied and then the attacker hijacks the DSVC of DER 6 and replaces the actual voltage with 482 V at $t = 0.65$ s. In the presence of attack, the conventional DSVC leads to the loss of desired consensus as shown in Fig. 3.12(a) and Fig. 3.12(b). Voltage and reactive power ratio for each DER deviate from the desired consensus and show oscillatory response. The corrupted voltage magnitude of DER 6 is directly
observed by reachable intact DERs which are affected by it. The reachable neighboring DERs also show oscillatory behaviors in their operating voltage and reactive power as shown in Fig. 3.12(a). This makes the relative entropy of compromised and reachable DERs diverge and go beyond the predefined design threshold of $\gamma_i = 5 \ \forall i$ in the presence of attack as shown in Fig. 3.13.

**Case A.2.2 (attack detection and mitigation on DSVC):** In this case, we consider the attack on DER 6. From $t = 0$ to $t = 0.65$ s only primary control is applied and then the attacker hijacks the DSVC of DER 6 and replaces the actual voltage with 482 V at $t = 0.65$ s. As shown in Fig. 3.14(a) and Fig. 3.14(b), voltage of all DERs except the hijacked one synchronize to 480 V after applying the mitigation mechanism at $t = 0.7$ s. The reactive power of DERs are also shared based on their ratings. Fig. 3.15(a) shows that the relative entropy for intact DERs remains close to zero, but it keeps growing for the compromised DER 6 due to deviation of the corrupted voltage from the nominal one, and, consequently, as shown in Fig. 3.15(b), self-belief for all DERs becomes one except for the compromised DER 6.
3.6.2 Case B: Simulation results for an Islanded Microgrid with 20 DERs

Case B verifies the validity of proposed control techniques on a 60Hz and 480V microgrid test system with 20 DERs. The single-line diagram of this microgrid test system is illustrated in Fig. 3.16. This test system is simulated in MATLAB/Simulink. The specifications of DERs are listed in Table III. Lines and loads specifications are shown in Tables IV. The communication network graph is depicted in Fig. 3.17. The frequency reference value is shared with DER1 with the pinning gain $g_1 = 1$. $\omega_{ref}$ is set to $2\pi \times 60$ rad/s. The control gains $c_\omega$ is set to 40. We assume zero-mean Gaussian communication noise with following statistical properties $\mathcal{N}(0, 0.01)$. This system is used to validate the proposed attack detection and mitigation schemes considering the DSFC.

Case B.1 (effect of attack on the conventional DSFC): We consider the attack on DER 20. At $t = 0$, the microgrid is islanded from the main grid. From $t = 0$ to $t = 0.7s$ only primary control is applied. The primary control takes action to provide frequency stability in the islanded microgrid. However, primary control only maintains frequency in stable ranges and cannot maintain frequency at exactly 60 Hz. Then, the secondary distributed frequency control is applied at $t = 0.7s$ to restore microgrid frequency to 60Hz. However, the attacker hijacks the DSFC of DER 20 and replaces the actual frequency with 60.2 Hz. Fig. 3.18 shows that the conventional DSFC protocol leads to the loss of the desired consensus. The frequency of each DER deviates from the desired frequency of 60 Hz and shows oscillatory behavior. In the presence of attack, the behavior of the compromised DER 20 is directly affected by the attack signal and its corrupted frequency is shared with neighboring DERs. This causes an oscillatory behavior in the neighboring DERs as shown in Fig. 3.18(a). Fig. 3.19 shows that the relative entropy of compromised and neighboring DERs diverge due to deviation in their behavior from the nominal one and go beyond predefined design threshold which is assumed.
to be $\gamma_i = 5 \ \forall i$.

*Case B.2 (attack detection and mitigation):* Similar to Case B.1, at $t = 0$, the microgrid is islanded from the main grid. From $t = 0$ to $t = 0.7s$ only primary control is applied. Then, the secondary distributed frequency control is applied at $t = 0.7s$. The attacker hijacks the DSFC of DER 20 and
replaces the actual frequency with 60.2 Hz at $t = 0.7$ s and then, the designed attack detection and mitigation mechanism is applied at $t = 0.75$ s. As shown in Fig. 3.20, frequency of intact DERs restores to 60 Hz after applying the attack mitigation mechanism at $t = 0.75$ s. Active power of all DERs are also retrieved back as in intact mode. After applying resilient DSFC, intact DERs discard locally observed frequency of corrupted DER. Therefore, the relative entropy for intact DERs remains close to zero but it keeps growing for the compromised DER 20 as shown in Fig. 3.21(a). Fig. 3.21(b) shows that self-belief for all DERs becomes one except the compromised DER 20.
3.6.3 Case C: Experimental verification of proposed techniques using a hardware-in-the-loop testing setup

To experimentally validate the performance of proposed attack detection and mitigation techniques, a hardware-in-the-loop (HIL) laboratory testbed is developed using Opal-RT as a real-time digital simulator and Raspberry Pi modules. A microgrid testbed including four DERs is simulated in Opal-RT. The specifications of DERs, loads, and lines are summarized in Table V. It is assumed that DERs communicate to each other through the communication graph network in Fig. 3.22. The nominal operating voltage and frequency of the microgrid test system are 480 V and 60 Hz, respectively. The frequency reference value is shared with DER1 with the pinning gain $g_1 = 1$. $\omega_{ref}$ is set to $2\pi \times 60$ rad/s. The control gains $c_\omega$ is set to 40. We assume zero-mean Gaussian communication noise with following statistical properties $\mathcal{N}(0, 0.01)$.

As seen in Fig. 3.22, four Raspberry Pi modules are utilized in the HIL testing. Each Raspberry Pi module hosts the cyber-secure DSFC protocol for a DER. Raspberry Pi modules communicate to each other through a distributed communication network. The HIL setup, including Opal-RT, Raspberry Pi modules, Gigabit ethernet switch, and host computer, is shown in Fig. 3.23. The microgrid electric circuit, including DERs, loads, lines, and primary controllers, are modelled in RT-LAB. The DER local measurements including the voltage, frequency, and active/reactive power measurements are sent to the corresponding Raspberry Pi module through User Datagram Protocol (UDP). Each Raspberry Pi module runs three processes in parallel. These processes include receiving real-time DER measurements and sending secondary control references to DERs, communicating to the neighboring DER Raspberry Pi modules, and running the secondary control protocol and attack detection and mitigation techniques.

Case C.1 (effect of attack on the conventional DSFC): We consider the attack on DER 2. At $t = 0$, the microgrid is islanded from the main grid. From $t = 0$ to $t = 30$ s only primary control is applied. Then, the secondary distributed frequency control is applied at $t = 30$ s to restore microgrid frequency to 60 Hz. However, the attacker hijacks the DSFC of DER 2 and replaces the actual frequency with 66 Hz. Fig. 3.24(a) and Fig. 3.24(b) show that the conventional DSFC protocol leads to the loss of the desired consensus. The frequency of each DER deviates from the desired frequency of 60 Hz and shows oscillatory behavior. In the presence of attack, the behavior
Figure 3.22: Microgrid test system for HIL testing.

Figure 3.23: HIL Setup.

Figure 3.24: Case C: Effect of attack on DSFC: (a) frequency; (b) active power ratio.
of the compromised DER 2 is directly affected by the attack signal and its corrupted frequency is shared with neighboring DERs. This causes an oscillatory behavior in the neighboring DERs. Fig. 3.25 shows that the relative entropy of compromised and neighboring DERs diverge due to deviation in their behavior from the nominal one.

**Case C.2 (attack detection and mitigation):** At $t = 0$, the microgrid is islanded from the main grid. From $t = 0$ to $t = 30s$ only primary control is applied. Then, the secondary distributed frequency control is applied at $t = 30s$. The attacker hijacks the DSFC of DER 2 and replaces the actual frequency with 66 Hz at $t = 30s$ and then, the designed attack detection and mitigation mechanism is applied at the same time. As shown in Fig. 3.26(a) and Fig. 3.26(b), frequency of intact DERs restores to 60 Hz after applying the attack mitigation mechanism. Active power of all DERs are also retrieved back as in intact mode.

**3.6.4 Conclusion**

This chapter addresses the effects of data manipulation attacks on distributed secondary frequency and voltage control in AC microgrids. An information-theoretic approach is employed for design of
detection and mitigation mechanism. Each DER detects the misbehavior of its neighbors on the
distributed communication network and, consequently, calculates a belief related to the trustworthi-
ness of the received information. It is shown that using the proposed cyber-secure approach, a DER
can distinguish data manipulation attacks from legitimate events and only discards the information
received from a neighbor if it is compromised. The proposed approach is ensured to work under a
mild communication graph connectivity.
4.1 Introduction

In this chapter, we relax some connectivity assumptions in the network for the resilient control design provided in Chapters 2 and 3 and present a distributed adaptive resilient mechanism to mitigate attacks on sensors and actuators.

We first describe, supported with analysis, the adverse effects of cyber-physical attacks on the consensus of the DMAS. Specifically, we show how an attack on a compromised agent can propagate and even destabilize the entire network. Conditions under which the network becomes unstable are provided. We also show that the local neighborhood tracking error of agents becomes zero under specific types of attack while they are far from the synchronization. Therefore, existing robust control approaches such as $H_\infty$ that aim at minimizing the local neighborhood tracking error can no longer mitigate these types of attacks. Then, based on the results of attack analysis, to mitigate the effect of attacks on sensors and actuators, an observer-like anomaly detector is first designed which provides the expected normal behavior of the agents when there is no attack. An adaptive attack compensator is designed and augmented with the controller to mitigate attacks without discarding information from compromised or unattacked neighbors'. We show that the consensus error is uniformly bounded using the proposed controller in the presence of the attack, and the bound can be made arbitrarily small.

4.2 Preliminaries

4.2.1 Graph Theory

A directed graph $G$ consists of a pair $(\mathcal{V}, \mathcal{E})$ in which set of nodes and set of edges are represented by $\mathcal{V} = v_1, \ldots, v_N$ and $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$, respectively. The adjacency matrix is defined as $A = [a_{ij}]$, with $a_{ij} > 0$ if $(v_j, v_i) \in \mathcal{E}$. The set of nodes $v_i$ with edges incoming to node $v_j$ is called as neighbors

76
of node \( v_i \), namely \( \mathcal{N}_i = v_j : (v_j, v_i) \in \mathcal{E} \). The graph Laplacian matrix is defined as \( L = H - A \), where \( H = \text{diag}(h_i) \) is known as the in-degree matrix, with \( \sum_{j \in \mathcal{N}_i} a_{ij} \) as the weighted in-degree of node \( i \). A node is called as a root node if it can reach all other nodes of the graph \( \mathcal{G} \) through a directed path. A directed tree is an acyclic digraph with a root node, such that any other node of the digraph can be reached by one and only one directed path starting at the root node. A graph is said to have a spanning tree if a subset of the edges forms a directed tree.

Throughout the paper, \( \lambda(.) \) represents the eigenvalues of a matrix. \( (.)^{\text{adj}} \) refers to adjoint of a matrix. \( \ker(.) \) denotes the null space. Furthermore, \( \lambda_{\text{max}}(.) \) and \( \lambda_{\text{min}}(.) \) represent maximum and minimum eigenvalue of matrix, respectively. \( \text{diag}(.) \) denotes the diagonal matrix. If \( A \) is an \( m \times n \) matrix, with \( a_{ij} \) being its \( i - \text{th} \) row and \( j - \text{th} \) column entry, and \( B \) is a \( p \times q \) matrix, then the Kronecker product \( A \otimes B \) is the \( mp \times nq \) block matrix given by

\[
A \otimes B = \begin{bmatrix}
a_{11}B & \cdots & a_{1n}B \\
\vdots & \ddots & \vdots \\
a_{m1}B & \cdots & a_{mn}B
\end{bmatrix}.
\]

**Assumption 1.** The directed graph \( \mathcal{G} \) has a spanning tree.

This assumption is a minimum requirement for the graph to guarantee consensus even in the absence of the attack [91], [27].

### 4.2.2 Standard Distributed Consensus in MAS

Consider the DMAS with \( N \) agents having identical system dynamics represented by

\[
x_i(k+1) = Ax_i(k) + Bu_i(k), \quad \forall \ i = 1, \ldots, N,
\]

where \( x_i(k) \in \mathbb{R}^n \) and \( u_i(k) \in \mathbb{R}^m \) are the state and control input of agent \( i \), respectively. \( A \) and \( B \) are the system and input matrices, respectively. \( (A, B) \) is assumed to be stabilizable.

Define the local neighborhood tracking error for the agent \( i \) as

\[
\varepsilon_i(k) = (1 + h_i)^{-1} \sum_{j \in \mathcal{N}_i} a_{ij}(x_j(k) - x_i(k)).
\]

Consider the distributed control law for node \( i \) as [91]

\[
u_i(k) = cK\varepsilon_i(k), \quad \forall \ i = 1, \ldots, N,
\]
where \( c \) is a positive coupling gain and \( K \in \mathbb{R}^{m \times n} \) is a control gain, designed to guarantee that agents reach consensus, i.e., \( x_i(k) \to x_j(k) \ \forall i, j \). Define the global state vector as \( x(k) = [x^T_1(k), x^T_2(k), \ldots, x^T_N(k)]^T \in \mathbb{R}^{nN} \).

Then, by substituting the controller \( u_i(k) \) from (3.3) into the system dynamics (3.1), the global dynamics of DMAS can be expressed as [27]

\[
 x(k + 1) = [I_N \otimes A - c(I + H)^{-1}L \otimes BK]x(k). \tag{4.4}
\]

Then, the solution to (4.4) is given by

\[
 x(k) = [I_N \otimes A - c\hat{L} \otimes BK]^k x(0) \triangleq A^c_k x(0), \tag{4.5}
\]

where \( \hat{L} \) is the normalized graph Laplacian matrix defined as [91]

\[
 \hat{L} = (I + H)^{-1}L. \tag{4.6}
\]

Let the eigenvalues of \( \hat{L} \) be \( \lambda_i, \ \forall \ i = 1, \ldots, N \). Then, \( \lambda_i \) lies inside the unit circle centered at \( 1 + j0 \) for \( i = 2, \ldots, N \) and \( \lambda_1 = 0 \) [27].

**Lemma 1.** [27] Let \( \mathcal{R} \subset \mathcal{V} \) be the set of root nodes and \( r = [p_{11}, \ldots, p_{1N}]^T \) be the left eigenvector of \( \hat{L} \) for \( \lambda_1 = 0 \). Then, \( p_{1i} > 0 \) if \( i \in \mathcal{R} \) and \( p_{1i} = 0 \) if \( i \notin \mathcal{R} \).

**Theorem 1.** [91], [27] Let feedback gain \( K \) be designed such that \( A - c\lambda_i BK \) is Schur stable for \( i = 2, \ldots, N \). Then, DMAS reaches consensus and the final consensus value can be written as

\[
 x(k) \to (r^T \otimes A^k)((x_1(0))^T, \ldots, (x_N(0))^T)^T \ \forall \ i = 1, \ldots, N. \tag{4.7}
\]

### 4.3 Attack Analysis for Discrete-time DMAS

In this section, we model false-data injection attack on sensors and actuators, and analyze its adverse effects on DMASs.

Attacks on actuators of agent \( i \) can be modeled as

\[
 u^a_i(k) = u_i(k) + \gamma_i u^a_i(k), \tag{4.8}
\]

where \( u_i \) is the control law given in (3.3), \( u^a_i \) represents the attacker’s signal injected into the actuator of agent \( i \), \( u^c_i \) is the distorted control law applied to (3.1) and the scalar \( \gamma_i \) is 1 when there is an attack on actuators of agent \( i \) and 0, otherwise.
Attacks on sensors of agent \( i \) can be modeled as

\[
x^c_i(k) = x_i(k) + \delta_i x_i^a(k),
\]

where \( x_i \) represents the state of agent \( i \), \( x_i^a \) is the attacker’s signal injected into the sensor of agent \( i \), \( x^c_i \) is the distorted state and the scalar \( \delta_i \) is 1 when there is an attack on sensors of agent \( i \) and 0, otherwise.

Based on the distributed control law (4.3), and using (4.8) and (4.9) in (4.1), one can express the DMAS dynamics for an agent \( i \) as

\[
x_i(k + 1) = Ax_i(k) + Bu_i(k) + Bf_i(k), \quad \forall \ i = 1, \ldots, N,
\]

where \( f_i(k) \) represents the overall attack signal injected into the agent \( i \), which is given by

\[
f_i(k) = c(1 + h_i)^{-1} K \sum_{j=1}^{N} a_{ij} (\delta_j x_j^a(k) - \delta_i x_i^a(k)) + \gamma_i u_i^a(k).
\]

**Definition 1.** In a graph, agent \( i \) is reachable from agent \( j \) if there exists \([v_1, v_2, \ldots, v_l] \in \mathcal{V}\) such that \( a_{jv_1} a_{v_1 v_2} \ldots a_{v_l i} \neq 0 \) for some \( l \geq 0 \), i.e., there is a directed path of length \( l + 1 \) from node \( j \) to node \( i \).

**Definition 2 (Compromised and unattacked Agent).** We call an agent that is directly under attack as a compromised agent. An agent is called unattacked if it is not compromised. We denote the set of agents as \( \mathcal{N}_{Int} \), i.e., \( \mathcal{N}_{Int} = \mathcal{N} \setminus \mathcal{N}_{Comp} \) where \( \mathcal{N}_{Comp} \) denotes the set of compromised agents.

In the control systems, the internal model principle (IMP) states that the controller must incorporate a model of the dynamics that generate the signals which the control system is supposed to track. We show in Theorem 2 that the attacker can also leverage the IMP and incorporate some eigenvalues of the consensus dynamics in its attack design to destabilize the entire network.

**Definition 3. (IMP-based and non-IMP-based Attacks.)** Let the attacker designs its attack signal \( f_i(k) \) on sensors and/or actuators of the compromised agent \( i \) as

\[
f_i(k + 1) = W f_i(k),
\]

with \( W \in \mathbb{R}^{m \times m} \) as the attacker’s dynamic. Define

\[
\begin{align*}
\Lambda_W &= [\lambda_{W_1}, \ldots, \lambda_{W_m}] \\
\Lambda_A &= [\lambda_{A_1}, \ldots, \lambda_{A_n}],
\end{align*}
\]
as the set of eigenvalues of $W$ and the system dynamics matrix $A$, respectively. Then, if $\Lambda_W \subseteq \Lambda_A$, the attack signal is called the IMP-based attack. Otherwise, if $\Lambda_W \not\subseteq \Lambda_A$ or the attacker has no dynamics (e.g. a random signal), it is called a non-IMP based attack.

**Remark 1.** Note that, in this paper, attack signals are not restricted to IMP-based attacks that are designed based on the dynamics (4.12). Attacks are categorized into two classes in Definition 3 based on their effects on the network stability and it will be shown that IMP-based attacks can destabilize the network while non-IMP based attacks cannot. The non-IMP based attacks cover a broad range of attacks. The proposed resilient approach presented in section IV works for both IMP and non-IMP based attacks.

**Remark 2.** Note that if the attack is only on actuators of the agent $i$, i.e., $f_i(k) = u_{ai}^a(k)$, then the attacker can design $u_{ai}^a(k) = Wf_i(k - 1)$, as an IMP-based attack signal that follows dynamics in (12). If the attack is only on the sensor of the agent $i$, then $f_i(k) = -c(1 + h_i)^{-1}Kx_{ai}^a(k)$ and the attacker can design $x_{ai}^a(k) = Wf_i(k - 1)$, to follow the dynamics in (12). Note that the scalar coefficient cannot change the common mode of the dynamics. Hence, (12) can be implemented for attacks both on sensors and actuators.

We assume that the system matrix $A$ in (3.1) is marginally stable, with eigenvalues on the unit circle centered at the origin. This is a standard assumption in the literature for consensus and synchronization problems [29]. Define

$$S(k) = \sum_{j=1}^{N} p_{1j} f_j(k), \quad (4.14)$$

where $p_{1j}$ represents the element of left eigenvector corresponding to zero eigenvalue of $\hat{L}$. $S(k)$ in (4.14) is used in Theorem 2 to analyze the effects of attack on root nodes and non-root nodes. Note that $S(k) = \sum_{j=1}^{N} p_{1j} f_j(k)$ represents the product of the left eigenvector corresponding to the zero eigenvalue of the graph Laplacian matrix $\hat{L}$ and the attack vector $f(k)$. Based on Lemma 1, elements of the left eigenvector corresponding to non-root nodes are zero. Thus, for a non-root node, $S(k) = 0$ regardless of attack. On the other hand, the left eigenvector corresponding to root nodes are non-zero and thus $S(k) \neq 0$ if there is an attack on a root node.

**Theorem 2.** Consider the DMAS (4.10) under the attack $f_i(k)$ with the control protocol (4.3). Let $f_i(k)$ be designed as (4.12). Then,
1. An IMP-based attack destabilizes the complete network, if \( S(k) \neq 0 \), i.e., if attack is on a root node.

2. Any non-IMP based attack or IMP-based attack with \( S(k) = 0 \) deviates agents from the desired consensus behavior but does not cause instability, if agents are reachable from the compromised one.

**Proof.** The transfer function for the DMAS (4.1), from \( x_i(z) \) to \( u_i(z) \), in Z-domain, can be written as

\[
G(z) = \frac{x_i(z)}{u_i(z)} = (zI - A)^{-1}B. \tag{4.15}
\]

Using (4.3), the global control law under the influence of the attack can be expressed as

\[
u(z) = -(c\hat{L} \otimes K)x(z) + f(z), \tag{4.16}
\]

with \( u(z) = [u_1^T, \ldots, u_N^T]^T \), \( x(z) = [x_1^T, \ldots, x_N^T]^T \) and \( f(z) = [f_1^T, \ldots, f_N^T]^T \). Using (4.15) and (4.16), the system state in the global form can be written as

\[
x(z) = (I_N \otimes G(z))u(z) = (I_N \otimes G(z))(-c\hat{L} \otimes K)x(z) + f(z), \tag{4.17}
\]

where \( G(z) \in R^{nxm} \). Let \( M \) be a non-singular matrix such that \( \hat{L} = MAM^{-1} \), with \( \Lambda \) be the Jordan canonical form of \( \hat{L} \). The left and the right eigenvectors of \( \hat{L} \) corresponding to the zero eigenvalue are \( r \) and \( 1_N \), respectively [27]. Define \( M = [1 \ M_1] \), \( M^{-1} = [r^T \ M_2]^T \) where \( M_1 \in R^{N \times (N-1)} \) and \( M_2 \in R^{(N-1) \times N} \). Then, using (4.17) with \( \hat{L} = MAM^{-1} \), one has

\[
(M \otimes I_n)[I_{nN} + c\Lambda \otimes G(z)K](M^{-1} \otimes I_n)x(z) = (I_N \otimes G(z))f(z). \tag{4.18}
\]

Defining a state transformation as \( \tilde{x}(z) = (M^{-1} \otimes I_n)x(z) \) and premultiplying (4.18) with \( (M^{-1} \otimes I_n) \) gives

\[
\tilde{x}(z) = [I_{NN} + c\Lambda \otimes G(z)K]^{-1}(M^{-1} \otimes G(z))f(z). \tag{4.19}
\]

Let assume for simplicity that all the Jordan blocks are simple, i.e., \( M^{-1} = [p_{ij}] \) and \( M = [m_{ij}] \), where \( p_{ij} \) and \( m_{ij} \) represent the elements of matrices \( M^{-1} \) and \( M \). Then, for the agent \( i \), using (4.19) and the fact that the first eigenvalue of \( \hat{L} \) is zero and its corresponding right eigenvector is
\[ N \text{ i.e. } m_{i1} = 1, \text{ one has} \]

\[
x_i(z) = G(z) \sum_{j=1}^{N} p_{1j} f_j(z) + \sum_{h=2}^{N} m_{ih} [I_n + c \lambda_h G(z)K]^{-1} G(z) \sum_{j=1}^{N} p_{hj} f_j(z).
\]

(4.20)

We now show that \([I_n + cKG(z)\lambda_h]^{-1}\) is Schur and thus to check stability of agents under attacks, one needs to analyze the first term of (4.20).

Towards this end, since \((A - c \lambda_h BK), \forall h = 2, \ldots, N\) is Schur, therefore if we show that the roots of the characteristic polynomial \((A - c \lambda_h BK)\) are identical to the poles of \([I_n + cKG(z)\lambda_h]^{-1}\), then \([I_n + cKG(z)\lambda_h]^{-1}\) is also Schur. To this end, using (4.15), one has

\[
\Delta |(zI_n - (A - c \lambda_h BK))| = \Delta |(zI_n - A + c \lambda_h BK)|
\]

\[
= \Delta |zI_n - A|(I_n + c \lambda_h(zI_n - A)^{-1}BK)
\]

\[
= \Delta |zI_n - A|[\Delta |zI_n - A| + c \lambda_h(zI_n - A)^{adj} BK] \frac{\Delta |zI_n - A|}{\Delta |zI_n - A|}.
\]

(4.21)

Hence, this proves that the roots of the characteristic polynomial \((A - c \lambda_h BK)\) are identical to the poles of \([I_n + cKG(z)\lambda_h]^{-1}\), and thus \([I_n + cKG(z)\lambda_h]^{-1}\) is Schur.

To analyze the boundedness of the first term in (4.20), note that according to Lemma 1, \(\sum_{j=1}^{N} p_{1j} f_j(k)\) in (4.20), which is identical to \(S(k)\) in (4.14), is zero for an attack on non-root nodes and is nonzero if the attack is launched on root nodes. Consider now an IMP-based attack on a root node. Then, using the transfer function (4.15) and the attack signal defined in (4.12), one can write (4.20) as

\[
x_i(z) = \sum_{j=1}^{N} p_{1j} \frac{(zI_n - A)^{adj} B(zI_n - W)^{adj} f_i(0)}{(z^2 + \lambda_{A_i}^2)^2 \prod_{i=1, i \neq l}^{n} (z^2 + \lambda_{A_i}^2)(z^2 + \lambda_{W_i}^2)} + \sum_{h=2}^{N} m_{ih} [1 + cKG(z)\lambda_h]^{-1} G(z) \sum_{j=1}^{N} p_{hj} f_j(z),
\]

(4.22)

with \(\lambda_{A_i}\) as the marginal eigenvalue of the system dynamics \(A, \text{ i.e., } \lambda_{A_i}\) lies on the unit circle centered at the origin. Since the first term of (4.22) shows that the pole \(\lambda_{A_i}\) lies on the unit circle centered at the origin and has multiplicity greater than 1 due to IMP-based attacks, thus the system
states tend to infinity as $k \to \infty$. Therefore, attacks on root nodes destabilize the entire network in the sense that the state of all agents goes to infinity as time tends to infinity. This completes the proof of part 1.

To prove part 2, note that based on Lemma 1, if the attack is on a non-root node, then $S(k) = \sum_{j=1}^{N} p_{1j} f_j(k)$ is zero. Therefore, the first term in (4.20) becomes zero and it can be expressed as

$$x_i(z) = \sum_{h=2}^{N} m_{ih} [1 + cK(z) \lambda_h]^{-1} G(z) \sum_{j=1}^{N} p_{hj} f_j(z).$$

(4.23)

According to (4.21), $[I_n + cK(z) \lambda_h]^{-1}$ is Schur stable. Therefore, based on (3.23), and since agent $i$ is unattacked itself, which concludes $G(z)$ is also Schur, the system states are bounded, even in the presence of attacks. However, although agents that are reachable from a compromised agent show a stable behavior, their states deviate from the desired consensus value based on the result of Theorem 1 in [94]. This completes the proof.

Disturbance attenuation approaches focus on minimizing the effects of disturbance on the local neighborhood tracking error [92]. More specifically, the $H_{\infty}$ approach for DMAS (4.1) in presence of disturbance $w_i(k)$ designs a distributed control protocol as in (4.3), such that the desired consensus is achieved as in (4.7), if disturbance $w_i(k) = 0$ and the bounded $L_2$-gain condition is fulfilled for any disturbance $w_i(k) \in L_2(0, \infty)$

$$\sum_{k=0}^{\infty} \varepsilon^T(k) \bar{M} \varepsilon(k) \leq \gamma^2 \sum_{k=0}^{\infty} w^T(k) \bar{N} w(k),$$

(4.24)

where $\gamma > 0$ is attenuation constant, $\bar{M}$ and $\bar{N}$ are positive definite weight matrices.

**Lemma 2.** Consider the normalized graph Laplacian matrix $\hat{L}$ defined in (4.6). Then, $[\hat{L}^T \hat{L} - 2\hat{L}]$ is negative semidefinite.

**Proof.** Let $\lambda_k$ be the eigenvalue of the normalized graph Laplacian matrix $\hat{L}$. So, the eigenvalue of $[\hat{L}^T \hat{L} - 2\hat{L}]$ for a undirected graph can be written as

$$\lambda[\hat{L}^T \hat{L} - 2\hat{L}] = \lambda_k^2 - 2\lambda_k = (\lambda_k - 1)^2 - 1, \quad \forall k = 1, \ldots, N.$$  

(4.25)

Since all eigenvalues of matrix $\hat{L}$ lie inside unit circle centered at $1 + j0$, except $\lambda_1 = 0$ [27], therefore $(\lambda_k - 1)^2 - 1$ is less than or equal to zero for $k = 1, \ldots, N$. This shows that $[\hat{L}^T \hat{L} - 2\hat{L}]$ is negative semidefinite. □
In the following theorem, for the sake of simplicity, we consider the single integrator dynamics in global form given by

\[ x(k+1) = x(k) + u(k). \] (4.26)

Under the influence of attack, one can write the control input \( u(k) \) in (4.26) as

\[ u(k) = -\hat{L}x(k) + f(k). \] (4.27)

**Theorem 3.** Consider the DMAS with single integrator dynamics (4.26). Assume that the system is under a constant attack signal \( f(k) \). Then, \( \varepsilon_i(k) \to 0, \forall i \in \mathcal{N}_{Int} \) while agents do not reach the desired consensus.

**Proof.** Consider the Lyapunov function for the discrete-time DMAS as

\[ V(x(k), f(k)) = (-\hat{L}x(k) + f(k))^T (-\hat{L}x(k) + f(k)). \] (4.28)

For the considered system (4.26) under constant attack signal \( f(k+1) = f(k) \) with the control input (4.27), one has

\[
\begin{align*}
\Delta V(x(k), f(k)) & = (-\hat{L}[x(k) - \hat{L}x(k) + f(k)])^T (-\hat{L}[x(k) - \hat{L}x(k) + f(k)]) \\
& \quad -(-\hat{L}x(k))^T (-\hat{L}x(k)) - 2f(k)^T \hat{L}(-\hat{L}x(k) + f(k)).
\end{align*}
\] (4.29)

After simplifying (4.29) and based on Lemma 2, one has

\[
\begin{align*}
\Delta V(x(k), f(k)) & = (-\hat{L}x(k) + f(k))^T [\hat{L}^T \hat{L} \\
& \quad -2\hat{L}](-\hat{L}x(k) + f(k)) \leq 0.
\end{align*}
\] (4.30)

Then, using Lasalle’s invariance principle [93], the trajectories \((x(k), f(k))\) converge to a set that satisfies \( \Delta V(x(k), f(k)) = 0 \). Based on (4.30), this yields

\[
(-\hat{L}x(k) + f(k)) \in \ker(\hat{L}^T \hat{L} - 2\hat{L})
\] (4.31)

or

\[
(-\hat{L}x(k) + f(k)) = 0.
\] (4.32)
From (3.31), one has \((-\hat{L}x(k) + f(k)) = \bar{c}1_N\). According to this, the single integrator system dynamics becomes \(x_i(k + 1) = x_i(k) + \bar{c}\), which shows that it destabilizes the system. Therefore, \(x_i(k) \to \infty\) as \(k \to \infty\) \(\forall i = 1, \ldots, N\) while the local neighborhood tracking error goes to zero for all agent. Note that based on Theorem 2, (3.32) is the possible case when the attack is on a root node.

On the other hand, for an attack on a non-root node, from (4.32), one has \((-\hat{L}x(k) + f(k)) = 0\).

Since for the unattacked agent \(i\), \(f_i(k) = 0\), therefore, the local neighborhood tracking error for unattacked agents converge to zero, even in the presence of the attack.

We now show that unattacked agents do not reach the desired consensus, despite the fact the local neighborhood tracking error is zero. From (4.32), one has \(\hat{L}x(k) = f(k)\) which can be written for agent \(i\) as

\[
(1 + h_i)^{-1} \sum_{j=1}^{N} a_{ij}(x_j(k) - x_i(k)) = f_i(k).
\]

(4.33)

For a compromised agent \(i\), since \(f_i(k) \neq 0\), then, one has \(x_i(k) \neq x_j(k)\) for some \(i, j\).

Now assume that agent \(i\) is unattacked, i.e., \(f_i(k) = 0\). Then, based on (4.33), one has

\[
(1 + h_i)^{-1} \sum_{j=1}^{N} a_{ij}(x_j(k) - x_i(k)) = 0.
\]

(4.34)

Consider the unattacked agent \(i\) as an immediate neighbor of the compromised agent \(i_c\). Let assume by contradiction that only the compromised agent does not reach the desired consensus but all the unattacked agents reach the desired consensus. Using (4.34), one can write

\[
(1 + h_i)^{-1} \sum_{j \in N_i} a_{ij}(x_j - x_i) + a_{ii_c}(x_{ic} - x_i) = 0.
\]

(4.35)

Assuming that unattacked agents reach consensus, \(x_i(k) = x_j(k), \forall j \in N_i\). However, (4.35) cannot be satisfied if \(x_i(k) = x_j(k), \forall j \in N_i\) because \(x_{ic}(k) \neq x_i(k)\) and this contradicts the assumption. Therefore, this shows that the unattacked agent \(i\) is deviated from the desired consensus value. Similarly, one can use the same argument to show that all reachable agents from the compromised agent deviate from the desired consensus value. This completes the proof.

\[\square\]

**Corollary 1.** Let the attacker design its attack signal using the IMP-based approach described in Theorem 2. Then, it bypasses the \(H_\infty\) control protocol.
Proof. In the absence of the attack, minimizing the local neighborhood tracking error results in minimizing the consensus error. Therefore, the $H_{\infty}$ control in (4.24) is used to attenuate the effect of adversarial input on the local neighborhood tracking error. However, according to Theorem 3, in the presence of IMP attack, by making the local neighborhood tracking error go to zero, agents do not reach consensus. This completes the proof. \hfill \Box

4.4 Resilient Distributed Control Protocol for Attacks on Sensor and Actuator: An Adaptive Approach

In this section, the expected normal behavior of each agent is predicted using an observer-like predictor (called here expected state predictor), which employs the agent’s dynamics to predict its expected normal state at each time step. This expected state predictor does not use any actual state measurement, and, instead, calculates the expected normal state of the agent based on the evolution rule of its dynamics, and taking into account the local information it receives from its neighbors. Then, a distributed adaptive compensator is designed using predicted behavior of agents to compensate for any discrepancy between the actual state and its predicted normal one.

Consider the estimated state for agent $i$ as $\hat{x}_i(k)$. The distributed expected state predictor is designed as

$$\hat{x}_i(k+1) = A\hat{x}_i(k) + BK(1 + h_i)^{-1} \sum_{j=1}^{N} a_{ij} (\hat{x}_j - \hat{x}_i),$$

(4.36)

where the gain $K$ and the coupling coefficient $c$ are to be designed to ensure $A_c$ in (4.5) is Schur. The global expected state predictor state vector for (4.36) can be written as $\hat{x}(k) = [\hat{x}_1^T(k), \hat{x}_2^T(k), \ldots, \hat{x}_N^T(k)]^T \in \mathbb{R}^{nN}$.

Lemma 3. Consider the $N$ expected state predictors given in (4.36). Let the feedback gain $K$ and the coupling coefficient $c$ are designed to ensure $A_c$ in (4.5) is Schur. Then, the expected state predictor state $\hat{x}(k)$ converges to the desired consensus value.

Proof. The designed expected state predictor in (4.36) can be expressed as

$$\hat{x}_i(k+1) = A\hat{x}_i(k) + Bu_i(k),$$

(4.37)
where \( \hat{u}_i(k) = cK\hat{\epsilon}_i(k) \) with the local neighborhood tracking error \( \hat{\epsilon}(k) \) as

\[
\hat{\epsilon}_i(k) = (1 + h_i)^{-1} \sum_{j=1}^{N} a_{ij}(\hat{x}_j - \hat{x}_i)). \tag{4.38}
\]

One can write the global expected state predictor state dynamics as \( \hat{x}(k + 1) = A_c\hat{x}(k) \in \mathbb{R}^{nN} \) which yields \( \hat{x}(k) = A_c^k\hat{x}(0) \in \mathbb{R}^{nN} \). As \( A - c\lambda_iBK \) is Schur stable, with \( \lambda_i \) be the eigenvalues of the normalized graph Laplacian matrix \( \hat{L} \) for \( i = 2, \ldots, N \) and \( \lambda_1 = 0 \). Therefore, the expected state predictor states achieve the desired consensus value.

\[ \square \]

**Remark 3.** Note that a broad class of the DMAS includes the leader-follower or the containment control problem (i.e., DMAS with multiple-leader) for which even if the \( \hat{x}_i(0) \neq x_i(0) \), Lemma 3 is valid. This is because, the reference trajectory to be followed by agents is determined only by the leaders, which are assumed to be trusted by using more advanced sensors and investing more security. The system (4.36) acts as a reference model for the agents and if \( \hat{x}_i(0) \neq x_i(0) \), even for the unattacked DMAS, \( d_i \) in (4.42) will be nonzero until the difference between the initial conditions is gone. Agents converge to the desired behavior irrespective of initial values.

\[ \square \]

We now design a distributed resilient control protocol as

\[
u_{i,r}(k) = u_i(k) + u_{i,comp}(k), \tag{4.39}
\]

where \( u_i(k) \) represents the standard control protocol defined in (4.3) and \( u_{i,comp}(k) \) represents the distributed adaptive compensator protocol responsible for rejection of the adversarial input.

Consider the feedback gain \( K \) in the control protocol (4.3) given as

\[
K = (R_1 + B^T P_1 B)^{-1} B^T P_1 A = R_1^{-1} B^T P_1 A, \tag{4.40}
\]

where \( R_1 \) is a positive definite design matrix, and \( P_1 \) is the solution of

\[
A^T P_1 A - P_1 - A^T P_1 B(R_1 + B^T P_1 B)^{-1} B^T P_1 A = Q_1, \tag{4.41}
\]

with a positive definite matrix \( Q_1 \). The designed distributed control protocol is given by

\[
u_{i,r}(k) = cK\hat{\epsilon}_i(k) - d_i(k), \tag{4.42}
\]
where \( d_i(k) \) is the estimated response of the adaptive compensator and \( K \) is the gain given by (4.40) and (4.41). The local neighborhood tracking error \( \bar{\varepsilon}_i(k) \) in (4.42) is given by

\[
\bar{\varepsilon}_i(k) = (1 + h_i)^{-1} \sum_{j=1}^{N} a_{ij}(x_j^c(k) - x_i^c(k)).
\]

(4.43)

The update law for the distributed adaptive compensator is designed as

\[
d_i(k+1) = \theta c K(\hat{\varepsilon}_i(k) - \bar{\varepsilon}_i(k)) + \theta d_i(k),
\]

(4.44)

where \( \theta > 0 \) is a design parameter, and \( \bar{\varepsilon}_i(k) \) and \( \hat{\varepsilon}_i(k) \) are defined in (4.43) and (4.38).

According to Lemma 3, the expected state predictor converges to the desired consensus value. Therefore, consensus of DMAS can be achieved by showing the convergence of the agent state \( x_i(k) \) to the predicted state \( \hat{x}_i(k) \). Define the consensus error \( \tilde{x}(k) \) as

\[
\tilde{x}(k) = x(k) - \hat{x}(k).
\]

(4.45)

In the following theorem, we show that the consensus error remains bounded using the proposed resilient adaptive controller.

**Theorem 4.** Consider the DMAS (4.10) under attacks on sensors and actuators. Let the control protocol be developed as (4.42)-(4.44). Then, the agent’s consensus error in (4.45) is bounded, i.e., \( \|\tilde{x}(k)\| \leq b_0 \) for some bound \( b_0 \) and it can be made arbitrarily small, despite attack.

**Proof.** According to Lemma 3, the expected state predictor converges to the desired consensus value. Therefore, consensus of discrete-time DMAS can be achieved by showing the convergence of the agent state \( x_i(k) \) to the predicted state \( \hat{x}_i(k) \). Then, with (4.10) and (4.37), one can write \( \tilde{x}(k+1) \) as

\[
\tilde{x}(k+1) = (I_N \otimes A - c\hat{L} \otimes BK)\tilde{x}(k) - (I_N \otimes B)d(k),
\]

(4.46)

where

\[
d(k) = d(k) - f(k),
\]

(4.47)

denotes attack rejection error with \( d(k) = [d_1^T(k), d_2^T(k), \ldots, d_N^T(k)]^T \in \mathbb{R}^{nN} \) as the global adaptive compensator vector and the dynamics of the attack \( f(k) \) is defined in (4.12).

Using (4.44), the global dynamics of the adaptive compensator can be written as
\[ d(k + 1) = \theta c\hat{L} \otimes \hat{R}_1^{-1}B^TP_1A\hat{x}(k) + \theta \hat{d}(k) + \theta \hat{f}(k), \] (4.48)

where \( \hat{R}_1 = R_1 + B^TP_1B \) and \( \hat{f}(k) = 2f(k) - (\gamma \otimes I_N)u^a \). Note that \( \hat{f}(k) = f(k) \) only if the actuator of any agent is compromised. Define \( Q_2 = Q_2^T > 0 \) as \( Q_2 = cR_2(I + H)^{-1}L = cR_2\hat{L} \) with some positive definite matrix \( R_2 \). Let the real part of the minimum eigenvalue of the normalized graph Laplacian matrix \( \hat{L} \) be \( \lambda_m \).

Define the Lyapunov candidate function function as

\[ V(k) = \hat{x}^T(k)(Q_2 \otimes P_1)\hat{x}(k) + \theta^{-2}\hat{d}^T(k)(R_2 \otimes \hat{R}_1)\hat{d}(k). \] (4.49)

The difference equation of the Lyapunov candidate function can be written as

\[
\Delta V(k) = V(k + 1) - V(k) \\
= \underbrace{\hat{x}^T(k + 1)(Q_2 \otimes P_1)\hat{x}(k + 1) - \hat{x}^T(k)(Q_2 \otimes P_1)\hat{x}(k)}_{\text{part 1}} \]

\[ + \underbrace{\theta^{-1}\hat{d}^T(k + 1)(R_2 \otimes R_1)\hat{d}(k + 1) - \theta^{-1}\hat{d}^T(k)(R_2 \otimes R_1)\hat{d}(k)}_{\text{part 2}}. \] (4.50)

Using (4.46), part 1 of the difference equation of the Lyapunov candidate function (4.50) can be expressed as

\[
= \hat{x}^T(k)(Q_2 \otimes A^TP_1A - 2cQ_2\hat{L} \otimes A^TP_1BK) \\
+ c^2\hat{L}^TQ_2\hat{L} \otimes (BK)^TP_1BK - (Q_2 \otimes P_1))\hat{x}(k) \\
- 2\hat{x}^T(k)(Q_2 \otimes A^TP_1B) - c\hat{L}^TQ_2 \otimes (BK)^TP_1B)\hat{d}(k) \\
+ \hat{d}^T(k)(Q_2 \otimes B^TP_1B)\hat{d}(k). \] (4.51)

Using the Young’s inequality, one can further simplify and express (4.51) as

\[
\leq -\hat{x}^T(k)(Q_2 \otimes Q_1)\hat{x}(k) - \hat{x}^T(k)(-Q_2 + 2cQ_2\hat{L} \otimes A^TP_1BK)\hat{x}(k) \\
+ 2c^2\lambda_{\min}(c^2\hat{L}^T\hat{L} \lambda_{\min}(TQ_1^{-1}))\hat{x}^T(k)(Q_2 \otimes Q_1)\hat{x}(k) \\
- 2\hat{x}^T(k)(Q_2 \otimes A^TP_1B)\hat{d}(k) + 2\hat{d}^T(k)(Q_2 \otimes B^TP_1B)\hat{d}(k), \] (4.52)

where \( T = K^TB^TP_1BK \). We now consider the part 2 of the difference equation of the Lyapunov candidate function in (4.50) as

\[
\theta^{-2}\hat{d}^T(k + 1)(R_2 \otimes \hat{R}_1)\hat{d}(k + 1) - \theta^{-2}\hat{d}^T(k)(R_2 \otimes \hat{R}_1)\hat{d}(k), \] (4.53)
where $\tilde{R}_1 = (R_1 + B^TP_1B)$ is a positive definite matrix. Using (4.47), one can express (4.53) as

$$
\frac{1}{\theta^2}[d^T(k + 1)(R_2 \otimes \tilde{R}_1)d(k + 1) - 2d^T(k + 1)(R_2 \otimes \tilde{R}_1)f(k + 1)
+ f^T(k + 1)(R_2 \otimes \tilde{R}_1)(f(k + 1) - \tilde{d}^T(k)(R_2 \otimes \tilde{R}_1)\tilde{d}(k)].
$$

(4.54)

Using the dynamics of the distributed adaptive compensator in (4.48) with (4.54), one has

$$
\psi(k) \text{ is constant, i.e., } \psi(k) = \tilde{f}(k).
$$

Integrating (4.52) and (4.56) with further simplification, one has

$$
\Delta V \leq -\tilde{x}^T(k)(Q_2 \otimes Q_1)\tilde{x}(k)
-\tilde{x}^T(k)(-Q_2 + \frac{1}{2}cQ_2\hat{L}) \otimes A^TP_1BK)\tilde{x}(k)
+ 2e^2\lambda_{\text{min}}(L^T\hat{L})\lambda_{\text{min}}(TQ_1^{-1})\tilde{x}^T(k)(Q_2 \otimes Q_1)\tilde{x}(k)
- (\theta^{-2} - 2 + 2\lambda_{\text{min}}(c\hat{L}B^TP_1B\tilde{R}_1^{-1})\tilde{d}^T(k)(R_2 \otimes \tilde{R}_1)\tilde{d}(k)
+ 4[\tilde{f}(k) - \theta^{-1}\psi(k)f(k)]^T(R_2 \otimes \tilde{R}_1)[\tilde{f}(k) - \theta^{-1}\psi(k)f(k)],
$$

(4.57)

where $\psi(k)$ denotes how the value of attack signal changes at next time instant. If the attack signal is constant, i.e., $f(k + 1) = f(k)$, then $\psi(k) = 1$. Thus, one can infer that $\psi(k)$ is always bounded, i.e., $|\psi(k)| < \zeta \forall k$. Integrating (4.52) and (4.56) with further simplification, one has

$$
\Delta V \leq \frac{3}{2}\tilde{x}^T(k)(cQ_2 \otimes A^TP_1BK)\tilde{x}(k)
+ (2 - \theta^{-2})\tilde{d}^T(k)(R_2 \otimes \tilde{R}_1)\tilde{d}(k)
+ 4[\tilde{f}(k) - \theta^{-1}\psi(k)f(k)]^T(R_2 \otimes \tilde{R}_1)[\tilde{f}(k) - \theta^{-1}\psi(k)f(k)],
$$

(4.58)

One can show that $\Delta V \leq 0$, if the coupling coefficient satisfies $\frac{2}{\lambda_m} < c < \frac{1}{\lambda_m \sqrt{2\lambda_{\text{min}}(TQ_1^{-1})}}$ and

$$
\left\|\tilde{d}(k)\right\| > \frac{4\left\|f(k) - \theta^{-1}\psi(k)f(k)\right\|}{\theta^{-2} - 2 - 2\lambda_{\text{min}}(c\hat{L}B^TP_1B\tilde{R}_1^{-1})}.
$$
The design parameter $\theta$ can be chosen such that $\theta < \frac{1}{\sqrt{2+2\lambda_{\min}(cLB^TP_1BR_1^{-1})}}$ and then, one can ensure the bound in (4.58). This shows that the agent’s consensus error is bounded, i.e., $\|\tilde{x}(k)\| \leq b_0$ for some bound $b_0$. Therefore, the actual agent’s state $x(k)$ achieves the desired consensus behavior with a bounded error that can be made arbitrarily small by appropriate selection of the design parameter $\theta$. This completes the proof.

Remark 4. The coupling coefficient $c$ needs to be in a certain range which depends on the $\lambda_m$ and $\lambda_{\min}(TQ_1^{-1})$. This condition is standard in the literature of DMAS [91]. On the other hand, the condition for the bound on $\tilde{d}(k)$ in (4.58) depends on the design parameters $\theta$, and one can select this parameter to satisfy (4.58) which ensures $\Delta V \leq 0$. Thus, the bound on consensus error can be made arbitrarily small based on the selection of design parameter $\theta$. Moreover, this bound is conservative, and as shown in the simulation results, the consensus error almost goes to zero.

Remark 5. Compromised agents under the effect of the sensor attack might not be recovered completely and result a nonzero bound in the consensus error defined in (4.45). The proposed distributed adaptive law compensates the difference between the incoming neighboring sensor measurement ($x_c^i(k)$) and the desired state $\hat{x}_i(k)$ and $x_c^i(k) \neq x_i(k)$ in the case of sensor attack. Under the actuator attack $x_c^i(k) = x_i(k)$ and the bound on the consensus error can be made arbitrarily small.

4.5 Simulation Results

We consider a leader-follower network of autonomous underwater vehicle’s (AUV’s) for the evaluation of the presented results.

The communication network in Fig.4.1 considers the Sentry AUVs as agents which are manufactured by the Woods Hole Oceanographic Institution [95]. The linearized model of the Sentry is of 6 DOF, but it is generally decomposed into four non-interacting subsystems which are speed subsystem ($u$), the roll subsystem ($\phi$), the steering subsystem ($\nu, r, \psi$), and the diving subsystem ($\omega, q, z, \theta$). Here, we focus on the diving subsystem of Sentry AUV for the desired depth maneuvering in the leader-follower network. The diving subsystem dynamics of Sentry AUV follows the dynamics in (4.1) where
with \( x_i(k) = [\omega_i(k), q_i(k), z_i(k), \theta_i(k)]^T \), and \( u_i(k) = [\delta^b_i(k), \delta^b_i(k)]^T \), where \( \delta^b_i(k), \delta^b_i(k) \) and \( (\omega_i(k), q_i(k), z_i(k), \theta_i(k)) \) represent bone and stern plane deflections, and heave speed, pitch rate, depth and pitch, respectively.

Figure 4.1: Graph topology.

In the network communication graph, we assumed that the agent 0 represents a active non-autonomous leader which aim to follow a desired sinusoidal depth trajectory and agents 1 to 5 designate the followers. The leader has the control input \( u_0(k) = K_0 x_0(k) + r(k) \), where \( K_0 \) is state feedback gain, \( x_0 \) denotes the leader state and \( r(k) \) represents the desired sinusoidal trajectory, respectively. Since the leader input is non-zero, slightly different discrete-time control protocol from that the one proposed in the paper is used for which the leader exchanges its input signal \( u_0 \) with its neighbors and agents reach consensus by exchanging states and leader’s input. This, however, does not change our attack analysis and mitigation. The state feedback gain \( K_0 = [ -0.18 -2.25 0.13 -0.21; 1.56 5.39 0.49 1.59 ] \)
Now, the effect of multiple attacks on the network is analyzed. We consider attack on actuators of Agent 2 and Agent 3 with attack signals $u_{a2}(k) = [30\ 30]'$ and $u_{a3}(k) = [20\sin(k)\ 20\sin(k)]'$, respectively at $t = 40$ sec. Fig. 4.2 shows that agents which are reachable from compromised Agents 2 and 3 are deviated from the desired behavior. This verifies results of Theorem 2. Then, Fig. 4.3 illustrates the response of the system under the influence of multiple attacks using the proposed controller with $Q_1$ and $R_1$ be identity matrix in (4.40) and (4.41), respectively. The system states achieve the desired consensus behavior, even in the presence of the attack. This result demonstrates the effectiveness of the proposed resilient controller in Theorem 4 for multiple attacks. Note that result in Fig. 4.3 also shows that this approach is not limited to particular model of attack and attack signal can be constant or time-varying. Compared to the existing work such as [18], the
presented approach brings back the compromised agents to the network. However, approaches such as [18] work also for attacks on the communication network while the presented approach is limited to attacks on sensors and actuators.

4.6 Conclusion

This paper analyzes the effects of attacks for leaderless DMAS and designs an adaptive resilient distributed control protocol for attack mitigation. It is shown that how the IMP-based attack on a root node can destabilize the entire network. To overcome the effect of the attacks on sensors and actuators, a resilient controller is developed based on an expected state predictor. The presented controller shows that the attack on the sensor and actuator can be mitigated without compromising the connectivity of the network and achieves the desired consensus. Although not considered in this paper, attacks on the communication links can be removed by integrating our approach with existing resilient methods presented in [10]-[11]. The presented approach also works for the leader-follower problems in which leaders are assumed to be trusted.
CHAPTER 5
SECURE EVENT-TRIGGERED DISTRIBUTED KALMAN FILTERS FOR STATE ESTIMATION OVER WIRELESS SENSOR NETWORKS

5.1 Introduction

Motivated by results on resilient designs presented in the previous chapters, we consider the problem of secure state estimation for distributed sensor networks. This chapter analyzes adverse effect of attacks and designs a resilient event-triggered distributed state estimation approach that can perform accurate state estimation despite attacks. More specifically, first, we show that the attacker can cause emerging non-triggering misbehavior so that the compromised sensors do not broadcast any information to their neighbors. This can significantly harm the network connectivity and its collective observability, which is a necessary condition for solving the distributed state estimation problem. We then show that an attacker can achieve continuous-triggering misbehavior which drains the communication resources and affects performance. Then, to detect adversarial intrusions a Kullback-Leibler (KL) divergence based detector is presented and estimated via k-nearest neighbors approach to obviate the restrictive Gaussian assumption on the probability density function of the attack signal. Based on attack detection results, finally, to mitigate attacks on event-triggered DKF, a meta-Bayesian approach is employed that performs second-order inference to form confidence and trust about the truthfulness or legitimacy of the outcome of its own first-order inference (i.e., the posterior belief about the state estimate) and those of its neighbors, respectively. Each sensor communicates its confidence to its neighbors and also incorporates the trust about its neighbors into its posterior update law to put less weight on untrusted data and thus successfully discard corrupted information.

5.2 Preliminaries

The data communication among sensors in a WSN is captured by an undirected graph $G$, consists of a pair $(\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, 2, \ldots, N\}$ is the set of nodes or sensors and $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ is the set of edges. An edge from node $j$ to node $i$, represented by $(j, i)$, implies that node $j$ can broadcast
information to node $i$. Moreover, $N_i = \{j : (j,i) \in E\}$ is the set of neighbors of node $i$ on the graph $\mathcal{G}$. An induced subgraph $\mathcal{G}^w$ is obtained by removing a set of nodes $W \subset V$ from the original graph $\mathcal{G}$, which is represented by nodes set $V \setminus W$ and contains the edges of $E$ with both endpoints in $V \setminus W$.

Throughout this chapter, $\mathbb{R}$ and $\mathbb{N}$ represent the sets of real numbers and natural numbers, respectively. $A^T$ denotes transpose of a matrix $A$. $tr(A)$ and $\max(a_i)$ represent trace of a matrix $A$ and maximum value in the set, respectively. $|S|$ represents the cardinality of a set $S$. $\sigma_{\max}(A)$, $\lambda_{\max}(A)$, and $I_n$ represent maximum singular value, maximum eigenvalue of matrix $A$, and an identity matrix of dimension $n$, respectively. $\mathcal{U}(a,b)$ with $a < b$ denotes an uniform distribution between the interval $a$ and $b$. Consider $p_X(x)$ as the probability density of the random variable or vector $x$ with $X$ taking values in the finite set $\{0,\ldots,p\}$. When a random variable $X$ is distributed normally with mean $\nu$ and variance $\sigma^2$, we use the notation $X \sim \mathcal{N}(\nu,\sigma^2)$. $E[X]$ and $\Sigma_X = E[(X-E[X])(X-E[X])^T]$ denotes, respectively, the expectation and the covariance of $X$. Finally, $E[\cdot|\cdot]$ represents the conditional expectation.

### 5.2.1 Process Dynamics and Sensor Models

Consider a process that evolves according to the following dynamics

$$x(k+1) = Ax(k) + w(k),$$

where $A$ denotes the process dynamic matrix, and $x(k) \in \mathbb{R}^n$ and $w(k)$ are, respectively, the process state and process noise at the time $k$. The process noise $w(k)$ is assumed to be independent and identically distributed (i.i.d.) with Gaussian distribution, and $x_0 \in \mathcal{N}(\hat{x}_0,P_0)$ represents the initial process state with $\hat{x}_0$ as mean and $P_0$ as covariance, respectively.

The goal is to estimate the state $x(k)$ for the process (5.1) in a distributed fashion using $N$ sensor nodes that communicate through the graph $\mathcal{G}$, and their sensing models are given by

$$y_i(k) = C_i x_i(k) + v_i(k); \quad \forall i = 1, \cdots, N,$$

where $y_i(k) \in \mathbb{R}^p$ represents the measurement data with $v_i(k)$ as the i.i.d. Gaussian measurement noise and $C_i$ as the observation matrix of the sensor $i$, respectively.

**Assumption 1.** The process noise $w(k)$, the measurement noise $v_i(k)$, and the initial state $x_0$ are uncorrelated random vector sequences.
Assumption 2. The sequences \( w(k) \) and \( v_i(k) \) are zero-mean Gaussian noise with

\[
E[w(k)(w(h))^T] = \mu_{kh} Q
\]

and

\[
E[v_i(k)(v_i(h))^T] = \mu_{kh} R_i,
\]

with \( \mu_{kh} = 0 \) if \( k \neq h \), and \( \mu_{kh} = 1 \) otherwise. Moreover, \( Q \geq 0 \) and \( R_i > 0 \) denote the noise covariance matrices for process and measurement noise, respectively and both are finite.

Definition 1. (Collectively observable) [106]. We call the plant dynamics (5.1) and the measurement equation (5.2) collectively observable, if the pair \((A, C_S)\) is observable where \( C_S \) is the stack column vectors of \( C_j \), \( \forall j \in S \) with \( S \subseteq V \) and \( C(S) > N/2 \).

Assumption 3. The plant dynamics (5.1) and the measurement equation (5.2) are collectively observable, but not necessarily locally observable, i.e., \((A, C_i) \ \forall i \in V\) is not necessarily observable.

Assumptions 1 and 2 are standard assumptions in Kalman filters. Assumption 3 states that the state of the target in (5.1) cannot be observed by measurements of any single sensor, i.e., the pairs \((A, C_i)\) cannot be observable (see for instances [106] and [132]). It also provides the necessary assumption of collectively observable for the estimation problem to be solvable. Also note that under Assumption 2, i.e., the process and measurement covariance are finite, the stochastic observability rank condition coincides with the deterministic observability [Theorem 1, 43]. Therefore, deterministic observability rank condition holds true irrespective of the process and measurement noise.

5.2.2 Overview of Event-triggered Distributed Kalman Filter

This subsection presents the overview of the event-triggered DKF for estimating the process state \( x(k) \) in (5.1) from a collection of noisy measurements \( y_i(k) \) in (5.2).

Let the prior and posterior estimates of the target state \( x(k) \) for sensor node \( i \) at time \( k \) be denoted by \( x_i(k|k-1) \) and \( x_i(k|k) \), respectively. In the centralized Kalman filter, a recursive rule based on Bayesian inference is employed to compute the posterior estimate \( x_i(k|k) \) based on its prior estimate \( x_i(k|k-1) \) and the new measurement \( y_i(k) \). When the next measurement comes, the previous posterior estimate is used as a new prior and it proceeds with the same recursive estimation
rule. In the event-triggered DKF, the recursion rule for computing the posterior incorporates not only its own prior and observations, but also its neighbors’ predictive state estimate. Sensor $i$ communicates its prior state estimate to its neighbors and if the norm of the error between the actual output and the predictive output becomes greater than a threshold after a new observation arrives. That is, it employs the following event-triggered mechanism for exchange of data with its neighbors

$$\|y_i(k) - C_i \tilde{x}_i(k-1)\| < \alpha,$$

where $\alpha$ denotes a predefined threshold for event-triggering. Moreover, $\tilde{x}_i(k)$ denotes the predictive state estimate for sensor $i$ and follows the update law

$$\tilde{x}_i(k) = \zeta_i(k)x_i(k|k-1) + (1 - \zeta_i(k))A\tilde{x}_i(k-1), \ \forall i \in \mathcal{V},$$

with $\zeta_i(k) \in \{0,1\}$ as the transmit function. Note that the predictive state estimate update equation in (4) depends on the value of the transmit function $\zeta_i(k)$ which is either zero or one depending on the triggering condition in (3). When $\zeta_i(k) = 1$, then the prior and predictive state estimates are the same, i.e., $\tilde{x}_i(k) = x_i(k|k-1)$. When $\zeta_i(k) = 0$, however, the predictive state estimate depends on its own previous state estimate, i.e., $\tilde{x}_i(k) = A\tilde{x}_i(k-1)$.

Incorporating (5.4), the following recursion rule is used to update the posterior state estimate in the event-triggered DKF [112], [114] for sensor $i$ as

$$x_i(k|k) = x_i(k|k-1) + K_i(k)(y_i(k) - C_i x_i(k|k-1)) + \gamma_i \sum_{j \in \mathcal{N}_i} (\tilde{x}_j(k) - \tilde{x}_i(k)),$$

where

$$x_i(k|k-1) = Ax_i(k-1|k-1),$$

is the prior update. Moreover, the second and the third terms in (5.5) denote, respectively, the innovation part (i.e., the estimation error based on the sensor $i^{th}$ new observation and its prior prediction) and the consensus part (i.e., deviation of the sensor state estimates from its neighbor’s state estimates). We call this recursion rule as the Bayesian first-order inference on the posterior, which provides the belief over the value of the state.

Moreover, $K_i(k)$ and $\gamma_i$ in (5.5), respectively, denote the Kalman gain and the coupling coefficient. The Kalman gain $K_i(k)$ in (5.5) depends on the estimation error covariance matrices
associated with the prior \( x_i(k|k-1) \) and the posterior \( x_i(k|k) \) for the sensor \( i \). Let define the prior and posterior estimated error covariances as

\[
P_i(k|k-1) = E[(x(k) - x_i(k|k-1))(x(k) - x_i(k|k-1))^T],
\]

\[
P_i(k|k) = E[(x(k) - x_i(k|k))(x(k) - x_i(k|k))^T].
\]

which are simplified as [112], [114]

\[
P_i(k|k) = M_i(k)P_i(k|k-1)(M_i(k))^T + K_i(k)R_i(K_i(k))^T, \quad (5.8)
\]

and

\[
P_i(k|k-1) = AP_i(k-1|k-1)A^T + Q. \quad (5.9)
\]

with \( M_i(k) = I_n - K_i(k)C_i \). Then, the Kalman gain \( K_i(k) \) is designed to minimize the estimation covariance and is given by [112], [114]

\[
K_i(k) = P_i(k|k-1)(C_i)^T(R_i(k) + C_iP_i(k|k-1)(C_i)^T)^{-1}. \quad (5.10)
\]

Let the innovation sequence \( r_i(k) \) for the node \( i \) be defined as

\[
r_i(k) = y_i(k) - C_i\hat{x}_i(k|k-1), \quad (5.11)
\]

where \( r_i(k) \sim \mathcal{N}(0, \Omega_i(k)) \) with

\[
\Omega_i(k) = E[r_i(k)(r_i(k))^T] = C_iP_i(k|k-1)C_i^T + R_i(k). \quad (5.12)
\]

Note that for the notional simplicity, henceforth we denote the prior and posterior state estimations as \( x_i(k|k-1) \triangleq \hat{x}_i(k) \) and \( x_i(k|k) \triangleq \hat{x}_i(k) \), respectively. Also, the prior covariance and the posterior covariance are, respectively, denoted by \( P_i(k|k-1) \triangleq \hat{P}_i(k) \) and \( P_i(k|k) \triangleq \hat{P}_i(k) \).

The event-triggered DKF algorithm becomes

\[
\text{Time updates:}
\begin{cases}
\hat{x}_i(k+1) = A\hat{x}_i(k) \\
\hat{P}_i(k+1) = A\hat{P}_i(k)A^T + Q(k)
\end{cases} \quad (a)
\]

\[
\text{Measurement updates:}
\]

99
\[
\begin{align*}
\dot{x}_i(k) &= \ddot{x}_i(k) + K_i(k)y_i(k) - C_i\bar{x}_i(k) \\
+ \gamma_i \sum_{j \in N_i} (\ddot{x}_j(k) - \ddot{x}_i(k)), & \quad (a) \\
\ddot{x}_i(k) &= \zeta_i(k)\bar{x}_i(k) + (1 - \zeta_i(k))A\ddot{x}_i(k-1), & \quad (b) \\
K_i(k) &= \tilde{P}_i(k)C_i^T(R_i(k) + C_i\tilde{P}_i(k)C_i^T)^{-1}, & \quad (c) \\
\tilde{P}_i(k) &= M_i\tilde{P}_i(k) + K_i(k)R_i(k)(K_i(k))^T. & \quad (d)
\end{align*}
\]

**Remark 1.** Based on the result presented in [17, Th.1], the event triggered DKF (5.13)-(5.14) ensures that the estimation error \( \dot{x}_i(k) - x(k) \) is exponentially bounded in the mean square sense \( \forall i \in V \).

**Remark 2.** The consensus gain \( \gamma_i \) in (5) is designed such that the stability of the event-triggered DKF in (13)-(14) is guaranteed. Specifically, as shown in [Theorem 2, 19], if

\[
\gamma_i = \frac{2(I - K_iC_i)(\Gamma_i)^{-1}}{\lambda_{\text{max}}(\mathcal{L})\lambda_{\text{max}}((\Gamma)^{-1})}
\]

where \( \mathcal{L} \) denotes the Laplacian matrix associated with the graph \( \mathcal{G} \) and \( \Gamma = \text{diag}\{\Gamma_1, \ldots, \Gamma_N\} \) with \( \Gamma_i = (I - K_iC_i)^TA^T(\tilde{P}_i)^+A(I - K_iC_i), \forall i = \{1, \ldots, N\} \), then the stability of the event-triggered DKF in (13)-(14) is guaranteed. However, the design of event-triggered DKF itself is not the concern of this chapter and this chapter mainly analyzes the adverse effects of cyber-physical attacks on the event-triggered DKF and proposes an information-theoretic approach based attack detection and mitigation mechanism. Note that the presented attack analysis and mitigation can be extended to other event-triggered methods such as [113] and [115] as well.

### 5.2.3 Attack Modeling

In this subsection, we model the effects of attacks on the event-triggered DKF. An attacker can design a false data injection attack to affect the triggering mechanism presented in (5.3) and consequently compromise the system behavior.

**Definition 2.** *(Compromised and intact sensor node).* We call a sensor node that is directly under attack as a compromised sensor node. A sensor node is called intact if it is not compromised. Throughout the chapter, \( V^c \) and \( V \setminus V^c \) denote, respectively, the set of compromised and intact sensor nodes.
Consider the sensing model (5.2) for sensor node $i$ under the effect of the attack as

$$y_i^a(k) = y_i(k) + f_i(k) = C_i x_i(k) + v_i(k) + f_i(k),$$

(5.15)

where $y_i(k)$ and $y_i^a(k)$ are, respectively, the sensor $i$'s actual and corrupted measurements and $f_i(k) \in \mathbb{R}^p$ represents the adversarial input on sensor node $i$. For a compromised sensor node $i$, let $p' \subseteq p$ be the subset of measurements disrupted by the attacker.

Let the false data injection attack $\bar{x}_j(k)$ on the communication link be given by

$$\bar{x}_j^a(k) = \bar{x}_j(k) + \bar{f}_j(k), \quad \forall j \in N_i.$$  

(5.16)

Using (5.15)-(5.16), in the presence of an attack on sensor node $i$ and/or its neighbors, its state estimate equations in (5.14)-(5.13) becomes

$$\begin{cases}
\dot{x}_i^a(k) = \bar{x}_i^a(k) + K_i^a(k)(y_i(k) - C_i \bar{x}_i^a(k)) \\
+ \gamma_i \sum_{j \in N_i} (\bar{x}_j(k) - \bar{x}_i^a(k)) + f_i^a(k), \\
\bar{x}_i^a(k + 1) = A \bar{x}_i^a(k), \\
\bar{x}_i^a(k) = \zeta_i(k) \bar{x}_i^a(k) + (1 - \zeta_i(k)) A \bar{x}_i^a(k - 1),
\end{cases}$$

(5.17)

where

$$f_i^a(k) = K_i^a(k) f_i(k) + \gamma_i \sum_{j \in N_i} \bar{f}_j(k),$$

(5.18)

with

$$\bar{f}_j(k) = \zeta_j(k) \bar{f}_j(k) + (1 - \zeta_j(k)) \bar{f}_j(k - 1).$$

(5.19)

The Kalman gain $K_i^a(k)$ in presence of attack is given by

$$K_i^a(k) = P_i^a(k) C_i^T (R_i(k) + C_i P_i^a(k) C_i^T)^{-1}.$$  

(5.20)

The first part in (5.18) represents the direct attack on sensor node $i$ and the second part denotes the aggregative effect of adversarial input on neighboring sensors, i.e., $j \in N_i$. Moreover, $\bar{x}_i^a(k)$, $\bar{x}_i^a(k)$, and $\bar{x}_i^a(k)$ denote, respectively, the corrupted posterior, prior, and predictive state estimates. The Kalman gain $K_i^a(k)$ depends on the following corrupted prior state estimation error covariance

$$\hat{P}_i^a(k + 1) = A \hat{P}_i^a(k) A^T + Q.$$  

(5.21)

where the corrupted posterior state estimation error covariance $\hat{P}_i^a(k)$ evolution is shown in the following theorem.
Theorem 1. Consider the process dynamics (5.1) with compromised sensor model (5.15). Let the state estimation equation be given by (5.17) in the presence of attacks modeled by $f_i^a(k)$ in (5.18).

Then, the corrupted posterior state estimation error covariance $\hat{P}_i^a(k)$ is given by

$$
\hat{P}_i^a(k) = M_i^a(k)\hat{P}_i^a(k)(M_i^a(k))^T + K_i^a(k)[R_i(k) + \Sigma^f_i(k)](K_i^a(k))^T \\
+ 2\gamma_i \sum_{j \in N_i} (\hat{P}_{i,j}(k) - \hat{P}_i^a(k))(M_i^a(k))^T - 2K_i^a(k)\Xi_f(k) \\
+ \gamma_i^2 \left( \sum_{j \in N_i} (\hat{P}_{i,j}(k) - 2\hat{P}_{i,j}(k) + \hat{P}_i^a(k)) \right),
$$

(5.22)

where $\Sigma^f_i(k)$ and $\Xi_f(k)$ denote the attacker’s input dependent covariance matrices and $M_i^a = (I_n - K_i^a(k)C_i)$ with $K_i^a(k)$ as the Kalman gain and $\hat{P}_i^a(k)$ as the prior state estimation error covariance update in (5.20) and (5.21), respectively. Moreover, $\hat{P}_{i,j}(k)$ and $\hat{P}_{i,j}(k)$ are cross-correlated estimation error covariances updated according to (6)-(8).

Proof. See Appendix A.

Note that the corrupted state estimation error covariance recursion $\hat{P}_i^a(k)$ in (5.22) depends on the attacker’s input distribution. Since the state estimation depends on compromised estimation error covariance $\hat{P}_i^a(k)$, therefore, the attacker can design its attack signal to blow up the estimates of the desired process state and damage the system performance.

5.3 Effect of Attack on Triggering Mechanism

This section presents the effects of cyber-physical attacks on the event-triggered DKF. We show that although event-triggered approaches are energy efficient, they are prone to triggering misbehaviors, which can harm the network connectivity, observability and drain its limited resources.

5.3.1 Non-triggering Misbehavior

In this subsection, we show how an attacker can manipulate the sensor measurement to mislead the event-triggered mechanism and damage network connectivity and collective observability by causing non-triggering misbehavior as defined in the following Definition 3.

Definition 3 (Non-triggering Misbehavior). The attacker designs an attack strategy such that a compromised sensor node does not transmit any information to its neighbors by misleading the triggering mechanism in (5.3), even if the actual performance deviates from the desired one.
The following theorem shows how a false data injection attack, followed by an eavesdropping attack, can manipulate the sensor reading to avoid the event-triggered mechanism (5.3) from being violated while the actual performance could be far from the desired one. To this end, we first define the vertex cut of the graph as follows.

**Definition 4 (Vertex cut).** A set of nodes $C \subset \mathcal{V}$ is a vertex cut of a graph $\mathcal{G}$ if removing the nodes in the set $C$ results in disconnected graph clusters.

**Theorem 2.** Consider the process dynamics (5.1) with $N$ sensor nodes (5.2) communicating over the graph $\mathcal{G}$. Let sensor $i$ be under a false data injection attack given by

$$y^a_i(k) = y_i(k) + \theta^a_i(k)1_p, \quad \forall k \geq L + 1,$$

where $y_i(k)$ is the actual sensor measurement at time instant $k$ and $L$ denotes the last triggering time instant. Moreover, $\theta^a_i(k) \sim \mathcal{U}(a(k), b(k))$ is a scalar uniformly distributed random variable in the interval $(a(k), b(k))$ with

$$\begin{cases} a(k) = \varphi - \|C_i\tilde{x}_i(k-1)\| + \|y_i(k)\|, \\ b(k) = \varphi + \|C_i\tilde{x}_i(k-1)\| - \|y_i(k)\|, \end{cases}$$

where $\tilde{x}_i(k)$ and $\varphi < \alpha$ denote, respectively, the predictive state estimate and an arbitrary scalar value less than the triggering threshold $\alpha$. Then,

1. The triggering condition (5.3) will not be violated for the sensor node $i$ and it shows non-triggering misbehavior;

2. The original graph $\mathcal{G}$ is clustered into several subgraphs, if all sensors in a vertex cut are under attack (5.23).

**Proof.** Taking norms from both sides of (5.23), the corrupted sensor measurement $y^a_i(k)$ becomes

$$\|y^a_i(k)\| = \|y_i(k) + \theta^a_i(k)1_p\|.$$

Using the triangular inequality for (5.25) yields

$$\|y_i(k)\| - \|\theta^a_i(k)1_p\| \leq \|y^a_i(k)\| \leq \|y_i(k)\| + \|\theta^a_i(k)1_p\|.$$
Based on the bounds of $\theta^a_i(k)$, given by (5.24), (5.26) becomes

$$\|C_i \tilde{x}_i(k-1)\| - \varphi \leq \|y^a_i(k)\| \leq \|C_i \tilde{x}_i(k-1)\| + \varphi,$$

which yields

$$(\|y^a_i(k)\| - \|C_i \tilde{x}_i(k-1)\| - \varphi)(\|y^a_i(k)\| - \|C_i \tilde{x}_i(k-1)\| + \varphi) \leq 0. \quad (5.28)$$

This implies that the condition

$$\|y^a_i(k) - C_i \tilde{x}_i(k-1)\| \leq \varphi < \alpha,$$

always holds true. Therefore, under (5.23)-(5.24), the corrupted sensor node $i$ shows non-triggering misbehavior, which proves part 1.

We now prove part 2. Let $\mathcal{A}_n \subseteq \mathcal{V}^c$ be the set of sensor nodes showing non-triggering misbehavior. Then, based on the presented result in part 1, under the attack signal (5.23), sensor nodes in the set $\mathcal{A}_n$ are misled by the attacker and consequently do not transmit any information to their neighbors which make them to act as sink nodes. Since the set of sensor nodes $\mathcal{A}_n$ is assumed to be a vertex cut. Then, the non-triggering misbehavior of sensor nodes in $\mathcal{A}_n$ prevents information flow from one portion of the graph $\mathcal{G}$ to another portion of the graph $\mathcal{G}$ and thus clusters the original graph $\mathcal{G}$ into subgraphs. This completes the proof.

Remark 3. Note that to design the presented strategic false data injection attack signal given in (5.23) an attacker needs to eavesdrop the actual sensor measurement $y_i(k)$ and the last transmitted prior state estimate $\tilde{x}_i(L)$ through the communication channel. The attacker then determines the predictive state estimate $\tilde{x}_i(k)$ using the dynamics in (5.5) at each time instant $k \geq L+1$ to achieve non-triggering misbehavior for the sensor node $i$.

We provide Example 1 for further illustration of the results of Theorem 2.

Example 1. Consider a graph topology for a distributed sensor network given in fig. 5.1. Let the vertex cut $\mathcal{A}_n = \{5,6\}$ be under the presented false data injection attack in Theorem 2 and show non-triggering misbehavior. Then, the sensor nodes in $\mathcal{A}_n = \{5,6\}$ do not transmit any information to their neighbors under the designed false data injection attack. Moreover, the sensor nodes in $\mathcal{A}_n = \{5,6\}$ act as sink nodes and prevent information flow from subgraph $\mathcal{G}_1$ to subgraph $\mathcal{G}_2$ which clusters the graph $\mathcal{G}$ into two non-interacting subgraphs $\mathcal{G}_1$ and $\mathcal{G}_2$ as shown in Fig. 5.1. This
example shows that the attacker can compromise the vertex cut $A_n$ of the original graph $G$ such that it shows non-triggering misbehavior and harm the network connectivity or cluster the graph into various non-interacting subgraphs.

We now analyze the effect of non-triggering misbehavior on the collective observability of the sensor network. To do so the following definitions are needed.

**Definition 5 (Potential Set).** A set of nodes $P \subset V$ is said to be a potential set of the graph $G$ if the pair $(A, C_{V \setminus P})$ is not collectively observable.

**Definition 6 (Minimal Potential Set).** A set of nodes $P_m \subset V$ is said to be a minimal potential set if $P_m$ is a potential set and no subset of $P_m$ is a potential set.

**Remark 4.** Note that if the attacker knows the graph structure and the local pair $(A_i, C_i)$, $\forall i \in V$. Then, the attacker can identify the minimum potential set of sensor nodes $P_m$ in the graph $G$ and achieves non-triggering misbehavior for $P_m$. Thus, the set of sensor nodes $P_m$ does not exchange any information with its neighbors and becomes isolated in the graph $G$.
Corollary 1. Let the set of sensors that shows non-triggering misbehavior be the minimal potential set $\mathcal{S}_n$. Then, the network is no longer collectively observable and the process state reconstruction from the distributed sensor measurements is impossible.

Proof. According to the statement of the corollary, $\mathcal{S}_n$ represents a minimal potential set of the graph $\mathcal{G}$ and shows non-triggering misbehavior. Then, the sensor nodes in the set $\mathcal{S}_n$ do not transmit any information to their neighbors and they act as sink nodes, i.e., they only absorb information. Therefore, the exchange of information happen just between the remaining sensor nodes in the graph $\mathcal{G}\backslash\mathcal{S}_n$. Hence, after excluding the minimum potential nodes $\mathcal{S}_n$, the pair $(A, C_{\mathcal{G}\backslash\mathcal{S}_n})$ becomes unobservable based on the Definitions 5 and 6, and thus makes the state reconstruction impossible. This completes the proof.

5.3.2 Continuous-triggering Misbehavior

In this subsection, we discuss how an attacker can compromise the actual sensor measurement to mislead the event-triggered mechanism and achieves continuous-triggering misbehavior and thus results in a time-driven DKF that not only drains the communication resources but also continuously propagates the adverse effect of attack in the network.

Definition 7 (Continuous-triggering Misbehavior). Let the attacker design an attack strategy such that it deceives the triggering mechanism in (5.3) at each time instant. This turns the event-driven DKF into a time-driven DKF that continuously exchanges corrupted information among sensor nodes. We call this a continuous-triggering misbehavior.

We now show how a reply attack, followed by an eavesdropping attack, can manipulate the sensor reading to cause continuous violation of the event-triggered mechanism (5.3).

Theorem 3. Consider the process dynamics (5.1) with $N$ sensor nodes (5.2) communicating over the graph $\mathcal{G}$. Let the sensor node $i$ in (5.2) be under a replay attack given by

$$y^0_i(k) = C_i \bar{x}_i(k - 1) + v_i(k), \quad \forall k \geq l + 1,$$

where $\bar{x}_i(k - 1)$ represents the last transmitted prior state and $v_i(k)$ denotes a scalar disruption signal. $l$ denotes the last triggering time instant when intact prior state estimate was transmitted. Then, the sensor node $i$ shows continuous-triggering misbehavior if the attacker selects $\|v_i(k)\| > \alpha$. 


Proof. To mislead a sensor to cause a continuous-triggering misbehavior, the attacker needs to design the attack signal such that the event-triggered condition (5.3) is constantly being violated, i.e., \( \|y_i^a(k) - C_i \bar{x}_i(k-1)\| \geq \alpha \) all the time. The attacker can eavesdrop the last transmitted prior state estimate \( \bar{x}_i(k-1) \) and design the strategic attack signal given by (5.30). Then, one has

\[
y_i^a(k) - C_i \bar{x}_i(k-1) = C_i \bar{x}_i(k-1) + \delta_i(k) - C_i \tilde{x}_i(k-1) \\
= C_i \bar{x}_i(k-1) + v_i(k) - C_i [\zeta_i(k-1) \bar{x}_i(k-1)] \\
+ (1 - \zeta_i(k-1))A \bar{x}_i(k-2) \\
= (1 - \zeta_i(k-1))C_i [\bar{x}_i(k-1) - A \bar{x}_i(k-2)] + v_i(k),
\]

(5.31)

Taking the norm from both sides of (5.31) yields

\[
\|y_i^a(k) - C_i \bar{x}_i(k-1)\| \\
= \|(1 - \zeta_i(k-1))C_i [\bar{x}_i(k-1) - A \bar{x}_i(k-2)] + v_i(k)\|,
\]

(5.32)

Since for \( k = l + 1, \zeta_i(l) = 1 \)

\[
\|y_i^a(l+1) - C_i \bar{x}_i(l)\| = \|v_i(l+1)\|,
\]

(5.33)

If the attacker selects \( v_i(l+1) \) such that \( \|v_i(l+1)\| > \alpha \), then the attack signal (5.30) ensures triggering at time instant \( k = l + 1 \). Then, based on similar argument for (5.32), \( \forall k \geq l + 1 \)

\[
\|y_i^a(k) - C_i \bar{x}_i(k-1)\| = \|v_i(k)\| > \alpha,
\]

(5.34)

which ensures continuous triggering misbehavior. This completes the proof.

To achieve continuous-triggering misbehavior the attacker needs to eavesdrop prior state estimates \( \bar{x}_i(k-1) \) at each triggering instant and selects the \( v_i(k) \) large enough such that \( \|v_i(k)\| > \alpha \) always holds true.

Note that continuous-triggering misbehavior can completely ruin the advantage of event-triggered mechanisms and turn it into time-driven mechanisms. This significantly increases the communication burden. Since nodes in the WSNs are usually powered through batteries with limited energy, the attacker can drain sensors limited resources by designing the above-discussed attack signals to achieve continuous-triggering misbehavior, and, consequently can make them non-operating in the network along with the deteriorated performance of the network.
Note that although we classified attacks into non-triggering misbehavior and continuous-triggering misbehavior, to analyze how the attacker can leverage the event-triggered mechanism, the following analysis, detection and mitigation approaches are not restricted to any class of attacks.

5.4 Attack Detection

In this section, we present an entropy estimation-based attack detection approach for the event-triggered DKF.

The KL divergence is a non-negative measure of the relative entropy between two probability distributions which is defined as follows.

**Definition 8 (KL Divergence) [36].** Let $X$ and $Z$ be two random variables with probability density function $P_X$ and $P_Z$, respectively. The KL divergence measure between $P_X$ and $P_Z$ is defined as

$$D_{KL}(P_X||P_Z) = \int_{\theta \in \Theta} P_X(\theta) \log \left( \frac{P_X(\theta)}{P_Z(\theta)} \right),$$ (5.35)

with the following properties [41]

1. $D_{KL}(P_X||P_Z) \geq 0$;

2. $D_{KL}(P_X||P_Z) = 0$ if and only if, $P_X = P_Z$;

3. $D_{KL}(P_X||P_Z) \neq D_{KL}(P_Z||P_X)$.

In the existing resilient literature, the entropy-based anomaly detectors need to know the probability density function of sequences, i.e., $P_X$ and $P_Z$, to determine the relative entropy. In most of the cases, authors assume that the probability density function of corrupted innovation sequence remains Gaussian (see [36] and [135] for instance). Since, the attacker’s input signal is unknown, it is restrictive to assume that the probability density function of the corrupted sequence remains Gaussian. To relax this *restrictive assumption* on probability density function of the corrupted sequence, we estimate the relative entropy between two random sequences $X$ and $Z$ using $k$–*nearest neighbor* ($k$ – NN) based divergence estimator [40].

Let $\{X_1, \ldots, X_{n_1}\}$ and $\{Z_1, \ldots, Z_{n_2}\}$ be i.i.d. samples drawn independently from $P_X$ and $P_Z$, respectively with $X_j, Z_j \in \mathbb{R}^m$. Let $d_k^X(i)$ be the Euclidean distance between $X_i$ and its $k$ – NN in $\{X_l\}_{l \neq i}$. The $k$ – NN of a sample $s$ in $\{s_1, \ldots, s_n\}$ is $s_{i(k)}$ where $i(1), \ldots, i(n)$ such that...
\[ \|s - s_{i(1)}\| \leq \|s - s_{i(2)}\| \leq \ldots \leq \|s - s_{i(n)}\|. \] (5.36)

More specifically, the Euclidean distance \(d_k^X(i)\) is given by [136]

\[ d_k^X(i) = \min_{j=1,\ldots,n_1 \neq \{i,j_1,\ldots,j_{k-1}\}} \|X_i - X_j\| \]

The \(k - NN\) based relative entropy estimator is given by [40]

\[ \hat{D}_{KL}(P_X||P_Z) = \frac{m}{n_1} \sum_{i=1}^{n_1} \log \frac{d_k^Z(i)}{d_k^X(i)} + \log \frac{n_2}{n_1 - 1}. \] (5.37)

The innovation sequences represent the deviation of the actual output of the system from the estimated one. It is known that innovation sequences approach a steady state quickly and thus it is reasonable to design innovation-based anomaly detectors to capture the system abnormality [36]. Using the innovation sequence of each sensor and the innovation sequences that it estimates for its neighbors, we present innovation based divergence estimator and design detectors to capture the effect of the attacks on the event-triggered DKF.

Based on innovation expression (5.11), in the presence of attack, one can write the compromised innovation \(r^a_i(k)\) for sensor node \(i\) with disrupted measurement \(y^a_i(k)\) in (5.15) and state estimation \(\bar{x}_i^a\) based on (5.17) as

\[ r^a_i(k) = y^a_i(k) - C_i\bar{x}_i^a(k). \] (5.38)

Let \(\{r^a_i(l),\ldots,r^a_i(l - 1 + w)\}\) and \(\{r_i(l),\ldots,r_i(l - 1 + w)\}\) be i.i.d. \(p\)-dimensional samples of corrupted and nominal innovation sequences with probability density function \(P_{r_i^a}\) and \(P_{r_i}\), respectively. The nominal innovation sequence follows \(r_i(k)\) defined in (5.11). Using \(k - NN\) based relative entropy estimator (5.37), one has [40]

\[ \hat{D}_{KL}(P_{r_i^a}||P_{r_i}) = \frac{p}{w} \sum_{j=1}^{w} \log \frac{d_{k}^{r^a_i(j)}}{d_{k}^{r_i(j)}} + \log \frac{w}{w-1}, \forall i \in \mathcal{V}. \] (5.39)

Define the average of the estimated KL divergence over a time window of \(T\) as

\[ \Phi_i(k) = \frac{1}{T} \sum_{l=k-T+1}^{k} \hat{D}_{KL}(P_{r_i^a}||P_{r_i}), \forall i \in \mathcal{V}. \] (5.40)

Now, in the following theorem, it is shown that the effect of attacks on the sensors can be captured using (5.40).

**Theorem 4.** Consider the distributed sensor network (5.1)-(5.2) under attack on sensor. Then,
1. in the absence of attack, $\Phi_i(k) = \log(w/w - 1), \forall k$;

2. in the presence of attack, $\Phi_i(k) > \delta, \forall k > l_a$, where $\delta$ and $l_a$ denotes, respectively, a predefined threshold and the time instant at which the attack happen.

**Proof.** In the absence of attack, the samples of innovation sequences $\{r^a_i(l), \ldots, r^a_i(l - 1 + w)\}$ and $\{r_i(l), \ldots, r_i(l - 1 + w)\}$ are similar. Then, the Euclidean distance $d^a_k(j) = d^i_k(j), \forall j \in \{1, ..., w\}$ and one has

$$\hat{D}_{KL}(P^a_i || P^i) = \log \frac{w}{w - 1}, \forall i \in \mathcal{V}.$$ (5.41)

Based on (5.41), one has

$$\Phi_i(k) = \frac{1}{T} \sum_{l=k-T+1}^{k} \log \frac{w}{w - 1} = \log \frac{w}{w - 1} < \delta, \forall i \in \mathcal{V}. \quad (5.42)$$

where $\log(w/w - 1)$ in (42) depends on the sample size of innovation sequence and $\log(w/w - 1) \leq 0.1, \forall w \geq 10$. Therefore, the predefined threshold $\delta$ can be selected with some $\delta > 0.1$ such that the condition in (42) is always satisfied. This complete the proof of part 1.

In the presence of attack, the samples of innovation sequences $\{r^a_i(l), \ldots, r^a_i(l - 1 + w)\}$ and $\{r_i(l), \ldots, r_i(l - 1 + w)\}$ are different, i.e., $d^a_k(j) \neq d^i_k(j), \forall j \in \{1, ..., w\}$. More specifically, $d^i_k(j) > d^a_k(j), \forall j \in \{1, ..., w\}$ due to change in the corrupted innovation sequence. Therefore, based on (5.39) the estimated relative entropy between sequences becomes

$$\hat{D}_{KL}(P^a_i || P^i) = \frac{p}{w} \sum_{j=1}^{w} \log(1 + \frac{\Delta^a_k(j)}{d^a_k(j)}) + \log \frac{w}{w - 1}, \forall i \in \mathcal{V}, \quad (5.43)$$

with $\Delta^a_k(j)$ as the change in Euclidean distance due to corrupted innovation sequence. Based on (5.43), one has

$$\hat{D}_{KL}(P^a_i || P^i) = \frac{p}{w} \sum_{j=1}^{w} \log(1 + \frac{\Delta^a_k(j)}{d^a_k(j)}) + \log \frac{w}{w - 1} \gg \log \frac{w}{w - 1}. \quad (5.44)$$

Thus, one has

$$\Phi_i(k) = \frac{1}{T} \sum_{l=k-T+1}^{k} \hat{D}_{KL}(P^a_i || P^i) > \delta, \forall i \in \mathcal{V}, \quad (5.45)$$

where $T$ and $\delta$ denote the sliding window size and the predefined design threshold. This completes the proof.
Based on Theorem 4, one can use the following condition for attack detection.

\[
\begin{align*}
\Phi_i(k) &< \delta : H_0, \\
\Phi_i(k) &> \delta : H_1,
\end{align*}
\tag{5.46}
\]

where \( \delta \) denotes the designed threshold for detection, the null hypothesis \( H_0 \) represents the intact mode of sensor nodes and \( H_1 \) denotes the compromised mode of sensor nodes.

Remark 5. Note that in the absence of an attack, the innovation sequence has a known zero-mean Gaussian distribution due to the measurement noise. Based on the prior system knowledge, one can always consider that the nominal innovation sequence is zero-mean Gaussian distribution with predefined covariance. The bound on the predefined covariance can be determined during normal operation of the event-triggered DKF. This assumption for the knowledge of the nominal innovation sequence for attack detection is standard in the existing literature (see [135] for instance). The designed threshold \( \delta \) in (5.46) is a predefined parameter and chosen appropriately for the detection of the attack signal. Moreover, the selection of detection threshold based on expert knowledge is standard in the existing literature. For example, several results on adversary detection and stealthiness have considered similar thresholds [36], [124].

**Algorithm 1** Detecting attacks on sensors.

1: Initialize with a time window \( T \) and detection threshold \( \delta \).

2: **procedure** \( \forall i = 1, \ldots, N \)

3: Use samples of innovation sequences \( \{r_i^a(l), \ldots, r_i^a(l - 1 + w)\} \) and \( \{r_i(l), \ldots, r_i(l - 1 + w)\} \) based on (5.38) and (5.11), \( \forall l \in k \).

4: Estimate the \( \hat{D}_{KL}(P_{r_i^a} || P_{r_i}) \) using (5.44).

5: Compute \( \Phi_i(k) \) as (5.45) and use condition in (5.46) to detect attacks on sensors.

6: **end procedure**

Based on the results presented in Theorem 4 and Algorithm 1, one can capture attacks on both sensors and communication links, but it cannot identify the specific compromised communication link as modelled in (5.16). To detect the source of attacks, we present an estimated entropy-based detector to capture the effect of attacks on the specific communication channel. More specifically, the relative entropy between the estimated innovation sequences for the neighbors at particular
sensor node and the nominal innovation sequence of the considered sensor node is estimated using (5.37).

Define the estimated innovation sequences \( \zeta_{a}^{i,j}(k) \) for a neighbor \( j \) under attacks on communication channel from the sensor node \( i \) side as

\[
\zeta_{a}^{i,j}(k) = y_{i}(k) - C_{j} \hat{x}_{j}^{a}(k),
\]

(5.47)

where \( \hat{x}_{j}^{a}(k) \) is the corrupted communicated state estimation of neighbor \( j \) at sensor node \( i \) at the last triggering instant.

Let \( \{ \zeta_{a}^{i,j}(l), \ldots, \zeta_{a}^{i,j}(l-1+w) \} \) be i.i.d. \( p \)-dimensional samples of neighbor’s estimated innovation at the sensor node \( i \) with probability density function \( P_{\zeta_{a}^{i,j}} \). Using \( k-NN \) based relative entropy estimator (5.37), one has

\[
\hat{D}_{KL}(P_{\zeta_{a}^{i,j}} \parallel P_{r_i}) = \frac{p}{w} \sum_{j=1}^{w} \log \frac{d_{j}^{i}(j)}{d_{k}^{i,j}(j)} + \log \frac{w}{w-1}, \forall i \in \mathcal{V}, j \in N_{i}.
\]

(5.48)

Note that in the presence of attacks on the communication channels, the neighbor’s actual innovation differs the neighbor’s estimated innovation at sensor \( i \). In the absence of the attack, the mean value of all the sensor state estimates converge to the mean of the desired process state at steady state, and, therefore, the innovation sequences \( r_{i} \) and \( \zeta_{a}^{i,j} \) have the same zero mean Gaussian distributions. In the presence of attack, however, as shown in Theorem 5 and Algorithm 2, their distributions diverge.

Define the average of the KL divergence over a time window of \( T \) as

\[
\Psi_{i,j}(k) = \frac{1}{T} \sum_{l=k-T+1}^{k} \hat{D}_{KL}(P_{\zeta_{a}^{i,j}} \parallel P_{r_i}), \forall i \in \mathcal{V}, \ j \in N_{i}.
\]

(5.49)

**Theorem 5.** Consider the distributed sensor network (5.1)-(5.2) under attack on communication links (5.16). Then, in the presence of an attack, \( \Psi_{i,j}(k) > \delta, \forall k \) where \( \delta \) denotes a predefined threshold.

**Proof.** The result follows a similar argument as given in the proof of part 2 of Theorem 4.

\( \square \)

### 5.5 Secure Distributed Estimation Mechanism

This section presents a meta-Bayesian approach for secure event-triggered DKF, which incorporates the outcome of the attack detection mechanism to perform second-order inference and
Algorithm 2 Detecting attack on a specific communication link.

1: Initialize with a time window $T$ and detection threshold $\delta$.
2: \textbf{procedure} $\forall i = 1, \ldots, N$
3: For each sensor node $j \in N_i$, use samples of innovation sequences $\{\zeta_{i,j}^a(l), \ldots, \zeta_{i,j}^a(l - 1 + w)\}$ and $\{r_i(l), \ldots, r_i(l - 1 + w)\}$ based on (5.47) and (5.11), $\forall l \in k$.
4: Estimate the $\hat{D}_{KL}(P_{\zeta_{i,j}} || P_{r_i})$ using (5.48).
5: Compute $\Psi_{i,j}(k)$ as (5.49) and use same argument in (5.46) to detect attacks on specific communication link.
6: \textbf{end procedure}

consequently form beliefs over beliefs. That is, the second-order inference forms confidence and trust about the truthfulness or legitimacy of the sensors’ own state estimate (i.e., the posterior belief of the first-order Bayesian inference) and those of its neighbor’s state estimates, respectively. Each sensor communicates its confidence to its neighbors. Then sensors incorporate the confidence of their neighbors and their own trust about their neighbors into their posterior update laws to successfully discard the corrupted information.

5.5.1 Confidence of sensor nodes

The second-order inference forms a confidence value for each sensor node which determines the level of trustworthiness of the sensor about its own measurement and state estimate (i.e., the posterior belief of the first-order Bayesian inference). If a sensor node is compromised, then the presented attack detector detects the adversary and it then reduces its level of trustworthiness about its own understanding of the environment and communicates it with its neighbors to inform them the significance of its outgoing information and thus slow down the attack propagation.

To determine the confidence of the sensor node $i$, based on the divergence $\hat{D}_{KL}(P_{r_i} || P_{r_i})$ from Theorem 4, we first define

$$\chi_i(k) = \frac{\Upsilon_1}{\Upsilon_1 + \hat{D}_{KL}(P_{r_i} || P_{r_i})},$$

with $0 < \Upsilon_1 < 1$ represents a predefined threshold to account for the channel fading and other uncertainties. Then, in the following lemma, we formally present the results for the confidence of sensor node $i$. 113
Lemma 1. Let $\beta_i(k)$ be the confidence of the sensor node $i$ which is updated using

$$\beta_i(k) = \sum_{l=0}^{k-1} (\kappa_1)^{k-l+1} \chi_i(l),$$

where $\chi_i(k)$ is defined in (5.50), and $0 < \kappa_1 < 1$ is a discount factor. Then, $\beta_i(k) \in (0,1]$ and

1. $\beta_i(k) \to 0$, $\forall i \in V^c$;

2. $\beta_i(k) \to 1$, $\forall i \in V \setminus V^c$.

Proof. Based on the expression (5.50), since $\hat{D}_{KL}(P_{r_i^a}||P_{r_i}) \geq 0$, one has $\chi_i(k) \in (0,1]$. Then, using (5.51), one can infer that $\beta_i(k) \in (0,1]$.

Now according to Theorem 4, if the sensor node $i$ is under attack, then $\hat{D}_{KL}(P_{r_i^a}||P_{r_i}) >> \Upsilon_1$ in (5.50), which makes $\chi_i(k)$ close to zero. Then, based on expression (5.51) with the discount factor $0 < \kappa_1 < 1$, the confidence $\beta_i(k)$ in (5.51) approaches zero, and thus the $i^{th}$ sensor’s belief about the trustworthiness of its own information would be low. This completes the proof of part 1.

On the other hand, based on Theorem 4, in the absence of attacks, $\hat{D}_{KL}(P_{r_i^a}||P_{r_i}) \to 0$ as $w \to \infty$, which makes $\chi_i(k)$ close to one and, consequently, $\beta_i(k)$ becomes close to one. This indicates that the $i^{th}$ sensor node is confident about its own state estimate. This completes the proof of part 2.

Note that the expression for the confidence of sensor node $i$ in (5.51) can be implemented using the following difference equation

$$\beta_i(k+1) = \beta_i(k) + \kappa_1 \chi_i(k).$$

(5.52)

Note also that the discount factor in (5.51) determines how much we value the current experience with regards to past experiences. It also guarantees that if the attack is not persistent and disappears after a while, or if a short-period adversary rather than attack (such as packet dropout) causes, the belief will be recovered, as it mainly depends on the current circumstances.

5.5.2 Trust of sensor nodes about their incoming information

Similar to the previous subsection, the second-order inference forms trust of sensor nodes to represent their level of trust on their neighboring sensor’s state estimates. Trust decides the usefulness of the neighboring information in the state estimation of sensor node $i$. 

114
The trust of the sensor node $i$ on its neighboring sensor $j$ can be determined based on the divergence $\hat{D}_{KL}(P_{\xi_{i,j}} || P_{r_i})$ in (5.47) from Theorem 5, from which we define

$$\theta_{i,j}(k) = \frac{\Lambda_1}{\Lambda_1 + \hat{D}_{KL}(P_{\xi_{i,j}} || P_{r_i})},$$

(5.53)

where $0 < \Lambda_1 < 1$ represents a predefined threshold to account for the channel fading and other uncertainties. Then, in the following lemma, we formally present the results for the trust of the sensor node $i$ on its neighboring sensor $j$.

**Lemma 2.** Let $\sigma_{i,j}(k)$ be the trust of the sensor node $i$ on its neighboring sensor $j$ which is updated using

$$\sigma_{i,j}(k) = \sum_{l=0}^{k-1} (\kappa_2)^{k-l+1} \theta_{i,j}(l),$$

(5.54)

where $\theta_{i,j}(k)$ is defined in (5.53), and $0 < \kappa_2 < 1$ is a discount factor. Then, $\sigma_{i,j}(k) \in (0, 1]$ and

1. $\sigma_{i,j}(k) \to 0$, $\forall j \in \mathcal{V}^c \cap \mathcal{N}_i$;
2. $\sigma_{i,j}(k) \to 1$, $\forall j \in \mathcal{V}\setminus\mathcal{V}^c \cap \mathcal{N}_i$.

**Proof.** The result follows a similar argument as given in the proof of Lemma 1. \qed

Note that the trust of sensor node $i$ in (5.54) can be implemented using the following difference equation

$$\sigma_{i,j}(k+1) = \sigma_{i,j}(k) + \kappa_2 \theta_{i,j}(k).$$

(5.55)

Using the presented idea of trust, one can identify the attacks on the communication channel and discard the contribution of compromised information for the state estimation.

### 5.5.3 Attack mitigation mechanism using confidence and trust of sensors

This subsection incorporates the confidence and trust of sensors to design a resilient event-triggered DKF. To this end, using the presented confidence $\beta_i(k)$ in (5.51) and trust $\sigma_{i,j}(k)$ in (5.54), we design the resilient form of the event-triggered DKF as

$$\hat{x}_i(k) = \bar{x}_i(k) + K_i(k)(\beta_i(k)y_i(k) + (1 - \beta_i(k))C_im_i(k) - C_i\bar{x}_i(k))$$

$$+ \gamma_i \sum_{j \in \mathcal{N}_i} \sigma_{i,j}(k)(\beta_j(k)(\bar{x}_j(k) - \bar{x}_i(k))),$$

(5.56)

where the weighted neighbor’s state estimate $m_i(k)$ is defined as
\[ m_i(k) = \frac{1}{|N_i|} \sum_{j \in N_i} \sigma_{i,j}(k) \beta_j(k) \tilde{x}_j(k) \approx x(k) + \varepsilon_i(k), \quad \forall k \ \| \varepsilon_i(k) \| < \tau, \]  

(5.57)

where \( \varepsilon_i(k) \) denotes the deviation between the weighted neighbor’s state estimate \( m_i(k) \) and the actual process state \( x(k) \). Note that in (5.57) the weighted state estimate depends on the trust values \( \sigma_{i,j}(k) \) and the confidence values \( \beta_j(k) \), \( \forall j \in N_i \). Since the weighted state estimate depends only on the information from intact neighbors, then one has \( \| \varepsilon_i(k) \| < \tau \) for some \( \tau > 0 \), \( \forall k \). For the sake of mathematical representation, we approximate the weighted state estimate \( m_i(k) \) in terms of the actual process state \( x(k) \), i.e., \( m_i(k) \approx x(k) + \varepsilon_i(k) \). We call this a meta-Bayesian inference that integrates the first-order inference (state estimates) with second-order estimates or belief (trust and confidence on the trustworthiness of state estimate beliefs).

Define the prior and predictive state estimation errors as

\[ \tilde{\eta}_i(k) = x(k) - \tilde{x}_i(k) \]
\[ \bar{\eta}_i(k) = x(k) - \bar{x}_i(k), \]

(5.58)

Using the threshold in triggering mechanism (5.3), one has

\[ \| \tilde{\eta}_i(k) \| - \| x(k + 1) - x(k) + v_i(k + 1) \| \leq \frac{\alpha}{\| C_i \|}, \]
\[ \| \bar{\eta}_i(k) \| \leq \frac{\alpha}{\| C_i \|} + B, \]

(5.59)

where \( B \) denotes the bound on \( \| x(k + 1) - x(k) + v_i(k + 1) \| \).

Other notations used in the following theorem are given by

\[ \bar{\eta}(k) = [\bar{\eta}_1(k), \ldots, \bar{\eta}_N(k)], \quad M(k) = diag[M_1(k), \ldots, M_N(k)] \]
\[ \Upsilon = diag[\gamma_1, \ldots, \gamma_N], \quad \Upsilon_m = \| \max \{ \gamma_i \} \|, \quad \forall i \in V, \]
\[ \bar{\beta} = (I_N - diag(\beta_i)), \quad E(k) = [\varepsilon_1(k), \ldots, \varepsilon_N(k)], \]
\[ \tilde{\eta}(k) = [\tilde{\eta}_1(k), \ldots, \tilde{\eta}_N(k)]. \]

(5.60)

**Assumption 4.** At least \( (\mathcal{C}(N_i)/2) + 1 \) neighbors of the sensor node \( i \) are intact.

Assumption 4 is similar to the assumption found in the secure estimation and control literature [7], [125]. Necessary and sufficient condition for any centralized or distributed estimator to resiliently estimate actual state is that the number of attacked sensors is less than half of all sensors.

**Theorem 6.** Consider the resilient event triggered DKF (5.56) with the triggering mechanism (5.3). Let the time-varying graph be \( \mathcal{G}(k) \) such that at each time instant \( k \), Assumptions 3 and 4 are satisfied. Then,
1. The following uniform bound holds on state estimation error in (5.58), despite attacks
\[ \|\bar{\eta}(k)\| \leq (A_o)^k \|\bar{\eta}(0)\| + \sum_{m=0}^{k-1} (A_o)^{k-m-1} B_o, \] (5.61)
where
\[ A_o = \sigma_{\max}((I_N \otimes A)M(k)), \]
\[ B_o = \sigma_{\max}(A)\sigma_{\max}(\mathcal{L}(k))Y_m \sqrt{N}(\alpha/\|C_i\| + B) \]
\[ + (\sigma_{\max}(A) + \sigma_{\max}(A_o)) \|\beta\| \sqrt{N}\tau, \] (5.62)
with \( \mathcal{L}(k) \) denotes the confidence and trust dependent time-varying graph Laplacian matrix, and bound \( \tau \) defined in (5.57);

2. The uniform bound on the state estimation error (5.61) becomes
\[ \lim_{k \to \infty} \|\bar{\eta}(k)\| \leq \frac{A_o B_o}{1 - A_o}. \] (5.63)
Moreover, other notations used in (5.62) are defined in (5.60).

Proof. Using the presented resilient estimator (5.56), one has
\[ \bar{x}_i(k+1) = A\bar{x}_i(k) \]
\[ = A(\bar{x}_i(k) + K_i(k)\beta_i(k)y_i(k) + (1 - \beta_i(k))C_i m_i(k) \]
\[ - C_i \bar{x}_i(k)) + \gamma_i \sum_{j \in N_i} \sigma_{i,j}(k)\beta_j(k)(\bar{x}_j(k) - \bar{x}_i(k)), \] (5.64)
Substituting (5.57) into (5.64) and using (5.58), the state estimation error dynamics becomes
\[ \bar{\eta}_i(k+1) = AM_i(k)\bar{\eta}_i(k) + A\gamma_i \sum_{j \in N_i} a_{ij}(k)(\bar{\eta}_j(k) - \bar{\eta}_i(k)) \]
\[ - AK_i(k)(1 - \beta_i(k))C_i \varepsilon_i(k), \] (5.65)
where \( a_{ij}(k) = \sigma_{i,j}(k)\beta_j(k) \) and \( M_i(k) = I - K_i(k)C_i \).
Using (5.65) and notations defined in (5.60), the global form of error dynamics becomes
\[ \bar{\eta}(k+1) = (I_N \otimes A)M(k)\bar{\eta}(k) - (\Upsilon \otimes A)\mathcal{L}(k)\bar{\eta}(k) \]
\[ - (\beta \otimes A)(I_nN - M(k))E(k)). \] (5.66)
Note that Assumption 4 implies that the total number of the compromised sensors is less than half of the total number of sensors in the network. That is, if \( q \) neighbors of an intact sensor node are attacked and collude to send the same value to mislead it, there still exists \( q + 1 \) intact neighbors that communicate values different from the compromised ones. Moreover, since at least
half of the intact sensor’s neighbors are intact, it can update its beliefs to discard the compromised neighbor’s state estimates. Furthermore, since the time-varying graph $G(k)$ resulting from isolating the compromised sensors, based on Assumptions 3 and 4, the entire network is still collectively observable. Using the trust and confidence of neighboring sensors, the incoming information from the compromised communication channels is discarded.

Now taking norm of equation (5.66) from both sides and then using the triangular inequality, one has

$$\|ar{\eta}(k+1)\| \leq \|(I_N \otimes A)M(k)\bar{\eta}(k)\| + \|((\Upsilon \otimes A)L(k)\bar{\eta}(k))
+ \|(\bar{\beta} \otimes A)(I_{nN} - M(k))E(k)\|.$$  

Using (5.67), (5.67) can be rewritten as

$$\|ar{\eta}(k+1)\| \leq A_0 \|ar{\eta}(k)\| + \sigma_{\text{max}}(L(k)) \|(\Upsilon \otimes A)\bar{\eta}(k)\|
+ \|(\bar{\beta} \otimes A) - (\bar{\beta} \otimes I_n)(I_N \otimes A)M(k))E(k)\|.$$  

After some manipulations, one has

$$\|ar{\eta}(k+1)\| \leq A_0 \|ar{\eta}(k)\| + \sigma_{\text{max}}(A) \sigma_{\text{max}}(L(k)) \Upsilon_m \|ar{\eta}(k)\|
+ (\sigma_{\text{max}}(A) + \sigma_{\text{max}}(A_o)) \|\bar{\beta}\| \sqrt{N\tau},$$  

with $\Upsilon_m$ defined in (5.60). Then, using (5.59), one can write (5.69) as

$$\|ar{\eta}(k+1)\| \leq A_0 \|ar{\eta}(k)\| + (\sigma_{\text{max}}(A) + \sigma_{\text{max}}(A_o)) \|\bar{\beta}\| \sqrt{N\tau}
+ \sigma_{\text{max}}(A) \sigma_{\text{max}}(L(k)) \Upsilon_m \sqrt{N\alpha / \|C_i\| + B}),$$  

After solving (5.70), one has

$$\|ar{\eta}(k)\| \leq (A_0)^k \|ar{\eta}(0)\| + \sum_{m=0}^{k-1} (A_0)^{k-m-1}B_0,$$

where $A_0$ and $B_0$ are given in (5.62). This completes the proof of part 1. Based on Assumption 3, the distributed sensor network is always collectively observable. Thus, based on result provided in [137], one can conclude that $A_0$ is always Schur and then the upper bound on state estimation error becomes (5.63). This completes the proof.

Based on the attack detection approach presented in Algorithms 1 and 2, one can detect the attacker’s misbehavior and estimate the actual state using the result presented in Theorem 6 and Algorithm 3.
Algorithm 3 Secure Distributed Estimation Mechanism (SDEM).

1: Start with initial innovation sequences and design parameters $\Upsilon_1$ and $\Lambda_1$.

2: procedure $\forall i = 1, \ldots, N$

3: Use samples of innovation sequences $\{r^a_i(l), \ldots, r^a_i(l - 1 + w)\}$ and $\{r_i(l), \ldots, r_i(l - 1 + w)\}$ based on (5.38) and (5.11), $\forall l \in k$.

4: Estimate the $\hat{D}_{KL}(P_{ra_i} || P_{r_i})$ using (5.44).

5: Based on (5.50)-(5.51), compute confidence $\beta_i(k)$ as

$$\beta_i(k) = \Upsilon_1 \sum_{l=0}^{k-1} \frac{(\kappa_1)^{k-l+1}}{\Upsilon_1 + \hat{D}_{KL}(P_{ra_i} || P_{r_i})}. \quad (5.72)$$

6: For each sensor node $j \in N_i$, use samples of innovation sequences $\{\zeta^a_{i,j}(l), \ldots, \zeta^a_{i,j}(l - 1 + w)\}$ and $\{r_i(l), \ldots, r_i(l - 1 + w)\}$ based on (5.47) and (5.11), $\forall l \in k$.

7: Estimate the $\hat{D}_{KL}(P_{ca_{i,j}} || P_{r_i})$ using (5.48).

8: Using (5.53)-(5.54), compute trust $\sigma_{i,j}(k)$ as

$$\sigma_{i,j}(k) = \Lambda_1 \sum_{l=0}^{k-1} \frac{(\kappa_2)^{k-l+1}}{\Lambda_1 + \hat{D}_{KL}(P_{ca_{i,j}} || P_{r_i})} \theta_{i,j}(l). \quad (5.73)$$

9: Using the sensor measurement $y_i(k)$ with the confidence $\beta_i(k)$, the trust on neighbor’s $\sigma_{i,j}(k)$ and neighbor’s state estimates $\hat{x}_j(k)$, $\forall j \in N_i$, update the resilient state estimator in (5.56).

10: end procedure

5.6 Simulation Results

In this section, we discuss simulation results to demonstrate the efficacy of presented attack detection and mitigation mechanism. The sensor network assumed to have following undirected graph topology as given in Fig. 5.2 with objective to follow the desired process dynamics.

Consider the process dynamics in (5.1) for generating the target trajectory as

$$x(k + 1) = \begin{bmatrix} \cos(\pi/200) & -\sin(\pi/200) \\ \sin(\pi/200) & \cos(\pi/200) \end{bmatrix} x(k) + w(k), \quad (5.74)$$

with the observation matrix $C_i$ in (5.2), noise covariances and initial state as

$$C_i = [5 \ 0; 0 \ 2], \ Q = I_2, \ R_i = I_2, \ x_0 = (0.5, 0). \quad (5.75)$$
Figure 5.3: Sensor network without any attack. (a) State estimation errors (b) Transmit function for sensor 2

For intact sensor network, the state estimates of sensors converge to the desired process state in the mean square sense and their state estimation error goes to zero for each sensor node as shown in Fig. 5.3 (a). The event generation based on the event-triggering mechanism in (5.3) with the triggering threshold $\alpha = 1.8$ is shown in Fig. 5.3 (b). Then, we consider the sensor 2 of the network is compromised with the adversarial input $\delta_2(k) = 2 + 10\sin(100k)$ after 20 seconds. Fig. 5.4
Figure 5.4: Sensor node 2 under continuous-triggering misbehavior. (a) State estimation errors (b) Transmit function for sensor 2

(a) shows the attacker’s effect on sensor 2 and one can notice that the compromised sensors and other sensors in the network deviates from desired target state and results in non-zero estimation error based on attacker’s input. Furthermore, the event generation based on the event-triggering mechanism in (5.3) in the presence of attack is shown in Fig. 5.4 (b) and it can be seen that after injection of the attack on sensor 2, the event-triggered system becomes time-triggered and shows continuous-triggering misbehavior. This result follows the analysis presented for the continuous-triggering misbehavior. In Fig. 5.5, we show the results for non-triggering misbehavior for sensor node 2 which follow the presented analysis.

Now, we detect the effect of the attack on the sensor using presented attack detection mechanism. Fig. 5.6 (a) shows the result for estimated KL divergence based attack detection mechanism and it illustrates that the after the injection of attack signal the estimated KL divergence starts increasing for compromised sensor node as well as for sensor nodes which has a path from the compromised sensor. One can always design a threshold and detect the effect of the attack in the sensor network.
Figure 5.5: Sensor node 2 under non-triggering misbehavior. (a) State estimation errors (b) Transmit function for sensor 2

and, then isolate the corrupted sensor in WSNs to avoid propagation of attack in the WSNs.

The estimated divergence for the compromised sensor, i.e., sensor 2 grows after attack injection at $k = 20$ which follows the result presented in the Theorem 4. The confidence of the sensor is evaluated based on the Lemma 1 with the discount factor $\kappa_1 = 0.5$ and the uncertainty threshold $\Upsilon_1 = 0.5$. Fig. 5.6 (b) shows the confidence of sensors in the presence of the considered attack which is close to one for healthy sensors and tends to zero for the compromised one. Then, the belief based proposed resilient estimator is implemented and Fig. 5.7 shows the result for the state estimation using the resilient estimator (5.56). After the injection of attack, within a few seconds, the sensors reach consensus on the state estimates, i.e., the state estimates of sensors converge to the actual position of the target. The result in Fig. 5.7 follows Theorem 6.
Figure 5.6: Sensor node 2 under attack. (a) Estimated KL divergence (b) Confidence of sensors

Figure 5.7: State estimation errors under attack on sensor 2 using proposed resilient state estimator.

5.7 Conclusion

In this chapter, first, we analyze the adverse effects of cyber-physical attacks on the event-triggered distributed Kalman filter (DKF). We show that attacker can adversely affect the perfor-
mance of the DKF. We also show that the event-triggered mechanism in the DKF can be leveraged by the attacker to result in a non-triggering misbehavior that significantly harms the network connectivity and its collective observability. Then, to detect adversarial intrusions in the DKF, we relax restrictive Gaussian assumption on probability density functions of attack signals and estimate the Kullback-Leibler (KL) divergence via k-nearest neighbors approach. Finally, to mitigate attacks, a meta-Bayesian approach is presented that incorporates the outcome of the attack detection mechanism to perform second-order inference and consequently form beliefs over beliefs, i.e., confidence and trust of a sensor. Each sensor communicates its confidence to its neighbors. Sensors then incorporate the confidence of their neighbors and their own trust about their neighbors into their posterior update laws to successfully discard corrupted sensor information. Then, the simulation result illustrates the performance of the presented resilient event-triggered DKF.
CHAPTER 6

ASSURED LEARNING-ENABLED AUTONOMY: A METACOGNITIVE REINFORCEMENT LEARNING FRAMEWORK

6.1 Introduction

This chapter presents the safe reinforcement learning (RL) framework for autonomous control systems under constraints. Reinforcement learning agents with pre-specified reward functions cannot provide guaranteed safety across variety of circumstances that an uncertain system might encounter. To guarantee performance while assuring the satisfaction of safety constraints across variety of circumstances, an assured autonomous control framework is designed by empowering RL algorithms with meta-cognitive learning capabilities. We first discuss that RL agents with pre-specified reward functions cannot guarantee satisfaction of the desired specified constraints and performances across all circumstances that an uncertain system might encounter. That is, the system either violates safety specifications or achieves no optimality and liveness specifications. To overcome this issue, a metacognitive decision-making layer is augmented with the RL agent to learn what reward functions to choose to satisfy desired specifications and to achieve a good enough performance across variety of circumstances. More specifically, a fitness function is defined in the metacognitive layer that indicates how safe the system would react in the future for a given reward function and in case of a drop in the fitness function, a Bayesian RL algorithm will proactively adapt the reward function parameters to maximize system’s assuredness (i.e., satisfaction of the desired STL safety and liveness specifications) and guarantee performance. Off-policy RL algorithms are proposed to find optimal policies corresponding to each hyperparameter by reusing the data collected from the system. The proposed approach separates learning the reward function that satisfies specifications from learning the control policy that maximizes the reward and thus allows us to evaluate as many hyperparameters as required using reused data collected from the system dynamics.
6.2 Preliminaries

6.2.1 Notations

Throughout the chapter, \(\mathbb{R}\) and \(\mathbb{N}\) represent the sets of real numbers and natural numbers, respectively. \(\mathbb{R}^n\) denotes \(n\)-dimensional Euclidean space. The superscript \((\cdot)^T\) denotes transposition. \(I\) denotes the identity matrix of proper dimension. \([K]_{ij}\) denotes the \(i,j\)-th element of the vector \(K\). \([K]_{ij}\) denotes the \([i,j]\)-th entry of the matrix \(K\). \(\text{diag}(A)\) denotes a diagonal matrix in which all off-diagonal entries are zero, i.e., \([A]_{ij} = 0, \forall i \neq j\). \(\text{Tr}(A)\) stands for trace of the matrix \(A\). When a random variable \(\varepsilon_i\) is distributed normally with mean \(m\) and variance \(w^2\), we use the notation \(\varepsilon_i \sim \mathcal{N}(m, w^2)\). \(\otimes\) denotes Kronecker product and \(\text{vec}(A)\) denotes the \(mn\)-vector constructed by stacking the columns of matrix \(A \in \mathbb{R}^{n \times m}\) on top of one another.

**Definition 1.** The weighted Kullback–Leibler (KL) divergence between distributions \(P\) and \(Q\) is defined as

\[
D_{KL}^h(X||Z) = \int P_X(\theta) \log \left( \frac{P_X(\theta)}{P_Z(\theta)} \right)^{h(x)} d\theta
\]

where \(h(x)\) is a non-negative real-valued weighting function.

Note that the weighting function is defined to weight more heavily promising regions of the state space. Note also that \(D_{KL}(P_X||P_Z) \geq 0\) and \(D_{KL}(P_X||P_Z) = 0\) if and only if \(P_X = P_Z\).

6.2.2 Signal Temporal Logic

Temporal logics can be used to specify rich time-dependent constraints for control systems in a wide variety of applications. Signal temporal logic (STL) is a category of temporal logics, which allows the specification of temporal properties of real-valued signals. STL is a predicate logic defined over continuous-time signals [165]-[168]. Let \(x(t)\) be a continuous-time signal. The predicates \(\sigma\) are evaluated as True \((\top)\) or False \((\bot)\) according to a corresponding predicate function \(z^\sigma(x) : \mathbb{R}^n \to \mathbb{R}\) as

\[
\sigma = \begin{cases} 
\top & z^\sigma(x) > 0 \\
\bot & z^\sigma(x) \leq 0 
\end{cases}
\]
The predicate function $z^\sigma(x)$ is a linear or nonlinear combination of the elements of the signal $x$ and the predicate $\sigma$ belongs to a set of predicates $P_\sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_N\}$ with $N \in \mathbb{N}$ denoting the number of predicates. Predicates can be recursively combined using Boolean logic negation ($\neg$), disjunction ($\lor$) and conjunction ($\land$) as well as temporal operators eventually ($\diamond$), globally or always ($\Box$) and until ($U$) to form increasingly complex formulas $\varphi$ (also referred to as task specifications) as

$$\varphi := T \mid \neg \sigma \mid \varphi_1 \land \varphi_2 \lor \varphi_1 \lor \varphi_2 \mid \diamond_{[a,b]} \varphi \mid \Box_{[a,b]} \varphi \mid \varphi_1 U_{[a,b]} \varphi_2$$

For each predicate $\sigma_i, i = 1 \ldots, N$, a predicate function $z^\sigma(x(t))$ is defined as (2). The time bounds of the until operator $\varphi U_{[a,b]} \mu$ are given as $a, b \in [0, \infty)$ with $a < b$. The commonly used temporal operators eventually and always follow from $\diamond_{[a,b]} \varphi = T U_{[a,b]} \varphi$, respectively. For example, the temporal formula $\diamond_{[3,6]} \varphi$ is satisfied when the STL formula $\varphi$ becomes True within the time interval of 3 to 6 seconds. A signal $x(t)$ is said to satisfy an STL expression at a time $t$ by the following qualitative semantics [164]-[167].

\[
\begin{align*}
(x, t) \models \sigma & \iff z^\sigma(x(t)) > 0 \\
(x, t) \models \neg \sigma & \iff \neg ((x, t) \models \sigma) \\
(x, t) \models \varphi \land \mu & \iff (x, t) \models \varphi \land (x, t) \models \mu \\
(x, t) \models \varphi \lor \mu & \iff (x, t) \models \varphi \lor (x, t) \models \mu \\
(x, t) \models \varphi U_{[a,b]} \mu & \iff \exists t_1 \in [t + a, t + b] \text{ s.t. } (x, t_1) \models \mu \\
& \quad \land \forall t_2 \in [t, t_1], (x, t_2) \models \varphi \\
(x, t) \models \diamond_{[a,b]} \varphi & \iff \exists t_1 \in [t + a, t + b] \text{ s.t. } (x, t_1) \models \varphi \\
(x, t) \models \Box_{[a,b]} \varphi & \iff \forall t_1 \in [t + a, t + b] \text{ s.t. } (x, t_1) \models \varphi
\end{align*}
\]

The symbol $\models$ denotes satisfaction of an STL formula. The time interval $[a, b]$ differentiates STL from general temporal logic and defines the quantitative timing to achieve the continuing temporal formula.

Apart from syntaxes and qualitative semantics, STL is also equipped with various robustness measures that quantify the extent to which a temporal constraint is satisfied. Given STL formulas $\varphi$ and $\mu$, the spatial robustness is defined as [167]

127
\[ \rho_{\sigma}(x,t) = z_{\sigma}(x(t)) \]
\[ \rho_{-\sigma}(x,t) = -\rho_{\sigma}(x,t) \]
\[ \rho_{\varphi \wedge \mu}(x,t) = \min(\rho_{\varphi}(x,t), \rho_{\mu}(x,t)) \]
\[ \rho_{\varphi \vee \mu}(x,t) = \max(\rho_{\varphi}(x,t), \rho_{\mu}(x,t)) \]
\[ \rho_{\varphi \bigcup [a,b]}(x,t) = \max_{t_1 \in [t+a,t+b]} \left( \min(\rho_{\mu}(x,t_1), \min_{t_2 \in [t,t_1]} (\rho_{\varphi}(x,t_2)) \right) \]
\[ \rho_{[a,b] \varphi}(x,t) = \max_{t_1 \in [t+a,t+b]} \rho_{\varphi}(x,t_1) \]
\[ \rho_{[a,b] \bigcap \varphi}(x,t) = \min_{t_1 \in [t+a,t+b]} \rho_{\varphi}(x,t_1) \]

This robustness measure determines how well a given signal \( x(t) \) satisfies a specification. The space robustness defines such a real-valued function \( \rho_{\sigma}(x,t) \) which is positive if and only if \( (x,t) \vDash \sigma \). That is, \( \rho_{\sigma}(x,t) > 0 \iff (x,t) \vDash \sigma \). Let a trajectory \( \tau[0,T] \) be defined by the signals \( x(t) \) throughout its evolution from time 0 to \( T \). A trajectory then satisfies the specification if and only if \( \rho_{\sigma}(x,t) > 0 \forall t \in [0,t_f] \), where \( t_f \) is the end time of the STL horizon. The robustness degree is the bound on the perturbation that the signal can tolerate without changing the truth value of specification.

### 6.2.3 Gaussian process

A Gaussian process (GP) can be viewed as a distribution over functions, in the sense that a draw from a GP is a function. GP has been widely used as a nonparametric regression method, where the goal is to find an approximation of a nonlinear map \( f : X \to \mathbb{R} \) from a state \( x \) to the function value \( f(x) \). The function values \( f(x) \) to be random variables, so that any finite number of them have a joint Gaussian distribution. When a process \( f \) follows a Gaussian process model, then

\[ f(.) \sim GP(m_0(.),k_0(.,.)) \]

where \( m_0(.) \) is the mean function and \( k_0(.,.) \) is the real-valued positive definite covariance kernel function [169]. In GP inference, the posterior mean and covariance of a function value \( f(x) \) at an arbitrary state \( x \) can be obtained by conditioning the GP distribution of \( f \) on a set of past measurements. Let \( X_n = [x_1,...,x_n] \) be a set of discrete state measurements, providing the set of
inducing inputs. For each measurement \( x_i \), there is an observed output \( y_i = f(x_i) = m(x_i) + \varepsilon_i \) where \( \varepsilon_i \sim N(0, w^2) \). The stack outputs give \( y = [y_1, ..., y_n] \). The posterior distribution at a query point \( x \) is also a Gaussian distribution and is given by [169]

\[
\begin{align*}
    m_n(x) &= m_0(x) + K(x, X_n)T(K_n + I w^2)^{-1} \\
    k_n(x, x') &= k_0(x, x') - K(x, X_n)T(K_n + I w^2)^{-1}K(x, X_I)
\end{align*}
\]

(6.6)

where the vector \( K(x, X_n) = [k_0(x, x_1), ..., k_0(x, x_n)] \) contains the covariance between the new data, \( x \), and the states in \( X_n \), and \( [K_n]_{ij} = k_0(x_i, x_j), \forall i, j \in \{1, ..., n\} \).

6.3 Problem Statement and Motivation

In this section, the problem of optimal control of systems subject to the desired specifications is formulated. We then discuss that optimizing a single reward or performance function cannot work for all circumstances and it is essential to adapt the reward function to the context to provide a good enough performance and assure safety and liveness of the system.

Consider the non-linear continuous-time system given by

\[
\dot{x}(t) = f(x(t)) + g(x(t))u(t)
\]

(6.7)

where \( x \in X \) and \( u \in U \) denote the admissible set of states and inputs, respectively. We assume that \( f(0) = 0, f(x(t)) \) and \( g(x(t)) \) are locally Lipschitz functions on a set \( \Omega \subseteq \mathbb{R}^n \) that contains the origin, and that the system is stabilizable on \( \Omega \).

The control objective is to design the control signal \( u \) for the system (6.7) to 1) make the system achieve desired behaviors (e.g., track the desired trajectory \( x_d(t) \) with good transient response) while 2) guaranteeing safety specifications specified by STL, i.e., guaranteeing \( (x, t) \models \sigma \) where \( \sigma \) belongs to a set of predicates \( P_\sigma = [\sigma_1, \sigma_2, \ldots, \sigma_N] \) with \( N \) as the number of the constraints. To achieve these goals, one can use an objective function for which its minimization subject to \( (x, t) \models \sigma \) provide an optimal and safe control solution aligned with the intention of the designer. That is, the following optimal control formulation can be used to achieve an optimal performance (encoded in the reward function \( r \)) while guaranteeing STL specifications.

**Problem 1 (Safety-Certified Optimal Control).** Given the system (6.7), find a control policy \( u \) that solves the following safe optimal control problem.
\[
\min J(x(t), u(t), x_d(t)) = \int_t^\infty e^{-\gamma(t-\tau)} r(x(\tau), u(\tau), x_d(\tau)) d\tau
\]
\[
s.t. (x, \tau) \models \sigma, \ \forall \tau \geq t
\]

where \(\gamma\) is a positive discount factor,

\[
r(x(t), u(t), x_d(t)) = \sum_j q_j r_j(x(t), u(t), x_d(t))
\]

is the overall reward function with \(r_j(x(t), u(t), x_d(t))\) as the cost for the \(i\)-th sub-goal of the system with \(q_j\) as its weight function, and \(x_d(t)\) is an external signal (e.g., reference trajectory).

The optimization framework in Problem 1, if efficiently solved, works well for systems operating in structured environments in which the system is supposed to perform a single task and the priorities across sub-goals (i.e., \(q_j\) in (6.9)) do not change over time and reference trajectories need not be adjusted. However, first of all, Problem 1 is hard to solve as it considers both optimality and safety in one framework. Second of all, even if efficiently solved, for complex systems such as self-driving cars for which the system might encounter numerous circumstances, a fixed reward function cannot capture the complexity of the semantics of complex tasks across all circumstances. As the circumstance changes, the previously rewarding maneuvers might not be achievable safely and thus the feasibility of the solution to Problem 1 might be jeopardized.

**Definition 2. (Feasible Control Policy).** Consider the system (6.7) with specifications \(\sigma\). A control policy \(u(t) = \mu(x)\) is said to be a feasible solution to Problem 1 if

1. \(\mu(x)\) stabilizes the system (6.7) on \(\Omega\).

2. There exists a safe set \(S \in X\) such that for every \(x_0 \in S, x_t(x_0, \mu) \models \sigma \ \forall t\), where \(x_t(x_0, \mu)\) is the state trajectory \(x(t)\) and time \(t \geq 0\) generated by (6.7) with the initial condition \(x_0\) and the policy \(u = \mu(x)\).

3. \(J(x, u, x_d) \leq \infty\) for all \(x \in \Omega\).

The feasibility of Problem 1 can be jeopardized as the context changes unless the reward weights and/or the reference trajectory are adapted to the context. For example, consider the case where a vehicle is performing some maneuvers with a desired velocity safely under a normal road condition. If, however, the friction of the road changes and the vehicle does not adapt its aspiration towards its
desired reference trajectory, when solving Problem 1, it must either violate its safety specifications or the performance function will become unbounded as the system’s state cannot follow the desired speed without violating safety. Since the performance will be unbound for any safe policy, the vehicle might only wander around and not reach any goal, providing a very poor performance. This highlights the importance of proposing a metacognitive framework that adapts to the context.

**Remark 1.** One might argue that the original weights or desired reference trajectory in Problem 1 can be appropriately designed in a context-dependent fashion to ensure satisfaction of the desired specifications across variety of circumstances. However, during the design stage, it is generally not possible to foresee the circumstances that will cause violation of the desired specifications and come up with a context-dependent reward function. This is generally due to modeling errors, unknown changes in the environment, and operator intervention.

Solving Problem 1 for systems with uncertain dynamics is hard. While RL algorithms can solve optimal control problems for systems with uncertain dynamics, they typically do so without taking into account safety constraints. To deal with this challenge, in this chapter, we use two layers of control to solve Problem 1 and guarantee its feasibility. In the lower layer, an RL algorithm is used to find an optimal controller that minimizes the performance (6.8) without considering the safety constraints. The metacognitive layer then monitors safety constraints and their level of satisfaction to proactively make meta-decisions about what reward function to optimize to guarantee feasibility of Problem 1 as the context changes. To guarantee satisfaction of the desired specifications with maximum assuredness, the metacognitive layer must be added on top of the lower-layer optimal control design to decide about priorities over sub-goals as well as the adaptation of the desired reference trajectory. The metacognitive layer monitors system-level operation and provides corrective action by optimizing a fitness function that guarantees systems’ liveness and safety, and thus ensures the maximum assuredness across different circumstances.

### 6.4 Metacognitive Control Architecture

To find an optimal solution while always guaranteeing satisfaction of the desired specifications with maximum assuredness, as shown in Fig. 6.1, a metacognitive RL algorithm is presented and it consists of:
• A low-level RL based controller $K$ for the system $S$ minimizes the performance (6.8) without considering the safety constraints.

• A high-level metacognitive controller $C$ adapts the reward function for the low-level controller $K$ to guarantee feasibility of Problem 1 and to maximize assuredness.

We aim at synthesizing the controller $C$ for the system (6.1) such that the closed-loop system achieves the desired objective defined in terms of minimization of a low-level cost function $J(x(t), u(t), x_d(t))$ in (6.8) while guaranteeing systems’ liveness and safety, i.e., $(x, t) \models \sigma$ in the metacognitive layer. Separation of RL control design to optimize the performance and metacognitive design to maximize assuredness by optimizing a fitness function significantly simplifies solving Problem 1 and allows to present data-based techniques for solving it. Let $\theta_1 \in \mathbb{R}^{d_1}$, $\theta_2 \in \mathbb{R}^{d_2}$, and $\theta_3 \in \mathbb{R}^{d_3}$ be vectors of parameters in matrices $Q(\theta_1)$, $R(\theta_2)$, and $x_d(\theta_3)$. Let $\lambda$ be defined as the set of all admissible hyperparameters $\theta$ where $\theta := [\theta_1^T, \theta_2^T, \theta_3^T]^T$. Note that we assume that the set of all admissible parameters $\theta \in \lambda$ is predefined by the designer based on some prior knowledge. With a slight abuse of notation, we write $r_\theta$, $Q_\theta$, and $R_\theta$ instead of $x_d(\theta_3)$, $Q(\theta_1)$, and $R(\theta_2)$ in what follows.

6.4.1 Metacognitive layer Monitoring and Control

The focus of this subsection is on designing a metacognitive layer that guarantees feasibility of Problem 1, regardless of the type of the RL-based controller (e.g., policy gradient, policy iteration, etc) used in the lower layer. While the proposed approach is not limited to any specific type of performance, in the sequel, we consider an optimal set-point tracking problem with STL safety specifications. That is, we consider the following parameterized reward function (6.9) in terms of the hyperparameter vector $\theta$.

$$r(x(\tau), u(\tau), r(\tau), \theta) = (x(\tau) - r_\theta)^T Q_\theta (x(\tau) - r_\theta) + u^T(\tau) R_\theta u(\tau)$$

(6.10)

where $Q_\theta$ and $R_\theta$ are parametrized design weight matrices, which are assumed diagonal, and $r_\theta$ is the desired set-point. The hyperparameter can be then defined as the stack of all parameters of the design weight matrices and the desired set-point. The performance function in Problem 1 then becomes
Figure 6.1: Proposed metacognitive control scheme. \( S \): The system to be controlled; \( K \): low-level RL controller; \( C \): high-level metacognitive layer scheme.

\[
J(x, u, r) = \int_0^\infty e^{-\gamma(t-\tau)}[(x(t) - r_\theta)^T Q_\theta(x(t) - r_\theta) + u^T(\tau) R_\theta u(\tau)]d\tau \\
(6.11)
\]

**Assumption 1.** Let \( u^*_\theta(x) \) be the optimal solution to Problem 1 for a set of hyperparameters \( \theta \) in the performance function. At each circumstance, there exists a \( \theta \) such that its corresponding optimal control policy \( u^*_\theta(x) \) is a feasible solution to Problem 1.

**Remark 2.** Note that the lower layer receives the hyperparameter vector \( \theta \) from the metacognitive layer and any RL algorithm for an unknown system can be used to minimize the accumulated reward function for the specified \( \theta \) and determine the optimal control policy in the lower layer. While the presented framework considers a model-free RL-based control approach in the lower layer, it can be applied to any performance driven-control architecture such as [170]-[171].

The metacognitive layer monitors the functionality of the system-level operations and performs
corrective actions for the lower-level controller when required. That is, hyperparameters (i.e., design weights and set-point) are adjusted in the metacognitive layer to guarantee feasibility of Problem 1 with maximum assuredness as the context changes. To monitor the functionality of the lower-layer, i.e., to check if it performs as intended, accumulated robustness degree of temporal logic specifications is used to define a fitness function as a metric for measuring the system’s safety (constraint satisfying) and liveness (goal-reaching and reference tracking). If the fitness function drops, which is an indication that the feasibility of the solution to Problem 1 is about to be violated, the metacognitive layer adapts the hyperparameters of the reward function to guarantee feasibility. Since feasibility of Problem 1 is only guaranteed if the safety STL specifications are satisfied and the performance remains bounded, we use degree of robustness of the safety STL specifications as well as degree of robustness of goal or reference reaching STL specifications (liveness specifications to assure the performance boundness) and define a fitness as a metric to proactively monitor the feasibility of Problem 1 and reacts before STL specifications are violated.

6.4.1.1 Metacognitive Monitoring

While \( N \) safety specifications are considered as constraints in Problem 1, we define the \( N + 1 \)-th specification as the liveness (set-point tracking) of the system as the following STL formula.

\[
\sigma_{N+1} = 0_{[0,t_s]}\neg(\|x(t) - r_\theta\| > \varepsilon) \tag{6.12}
\]

with \( \varepsilon \) as the envelope on the tracking error, and \( t_s \) as the expected settling time.

**Lemma 1.** Let \( u^*_\theta(x) \) be the optimal control policy found by solving Problem 1 with the performance (6.11). If \( u^*_\theta(x) \) is feasible, then \( x_t(x_0, u^*_\theta) \models \sigma_{N+1} \forall t \).

**Proof.** For a given \( \theta \), under the assumption that there exists a stabilizing control policy, it is shown in [170] that the performance is bounded for the optimal controller \( u^*_\theta(x) \). On the other hand, based on Barbalat lemma [172], a uniformly continuous real function, whose integral up to infinity exists and is bounded, vanishes at infinity. Therefore, the performance is bounded for \( u^*_\theta(x) \) if it makes \( \|x(t) - r(t)\| \) become very small after the settling time \( t_s \). This completes the proof. \( \square \)
Remark 3. Incorporating goal-reaching specifications can help adapt the reward function to avoid performing unintended functionalities in many applications. A classic example for which this could have helped resolve the problem is OpenAI’s demo (https://openai.com/blog/faulty-reward-functions/) for which an RL agent in a boat racing game kept going in circles while repeatedly hitting the same reward targets to gain a high score without having to finish the course.

The STL specification \((x, t) \vDash \sigma_{N+1}\) essentially states that the trajectory tracking error should eventually become less than \(\varepsilon\) after the expected settling time. Otherwise, the set-point \(r\) is aimed too high to be achieved safely and must be adjusted. The settling time could be obtained from the knowledge that we have from the control expectation from the lower layer and can be conservative.

We now extend the set of predicates from \(P = [\sigma_1, ..., \sigma_N]\) with \(N\) as the number of the constraints to \(P = [\sigma_1, ..., \sigma_N, \sigma_{N+1}]\) to include the liveness STL predicate. The monitor then predicts if \((x, t) \vDash \sigma\) will be satisfied all the time to make proactive meta decisions accordingly. Let the stack of predicate functions for safety and liveness specification STL \(\sigma_i \in P_\sigma\) be

\[
\begin{bmatrix}
z^{\sigma_1}(x(t)) \\
z^{\sigma_2}(x(t)) \\
\vdots \\
z^{\sigma_{N+1}}(x(t))
\end{bmatrix}
\]

Using \(z^{\sigma}(x)\), a fitness function is now designed to monitor and estimate the accumulated level of satisfaction of the desired STL specifications (i.e., the safety-value function) in the metacognitive layer. If the fitness function drops, which is an indication that either the safety constraints are about to be violated in the future or liveness of the system will not guaranteed, i.e., the feasibility of the solution to Problem 1 is about to be violated, then metacognitive layer proactively adapts the hyperparameters of the reward function to guarantee feasibility.

Definition 3. Consider a specific hyperparameter vector \(\theta\) and let \(u = \mu(x)\) be a feasible solution to Problem 1 with the performance (6.11). The set \(S \in X\) is called a viable set (safe and live) of
\[\mu(x) \text{ if for every } x_0 \in S, \ x_t(x_0, \mu) \models \sigma \ \forall t \text{ and is defined as} \]

\[S_{\mu, \theta}(x) = \{x_0 : x_t(x_0, \mu) \models \sigma, \ \forall t \geq 0\} \tag{6.15}\]

where \(\sigma\) belongs to the set of predicates \(P = [\sigma_1, ..., \sigma_N, \sigma_{N+1}]\) with the predicate functions defined in (6.13).

Note that the dependence of \(S_{\mu, \theta}(x)\) on \(\theta\) is because the specification (6.14) depends on \(\theta\).

**Lemma 2.** Consider a specific hyperparameter vector \(\theta\). If the set \(S_{\mu, \theta}(x)\) is empty for all control policies \(\mu \in U\), then there exists no feasible solution to Problem 1 for the hyperparameter vector \(\theta\).

**Proof.** The set of predicates \(P\) includes all safety constraints with predicates \(\sigma_1, ..., \sigma_N\) as well as the liveness condition with predicate \(\sigma_{N+1}\) defined in (6.12). If the set \(S_{\mu, \theta}(x)\) is empty, then, there is no control policy \(\mu(x)\) to simultaneously satisfy all safety constraints and make the performance bounded (based on Lemma 1). Therefore, based on Definition 1, there is no feasible solution to Problem 1. This completes the proof. \(\square\)

To monitor the feasibility of Problem 1 for the current hyperparameter vector \(\theta\), we now define a fitness function based on the quantitative semantics of STL specifications under an optimal control policy found by minimizing (6.11) for the given \(\theta\). Let \(u^*_{\theta}(x)\) be the optimal control policy found by minimizing the performance function (6.11) for a given \(\theta\) and is applied to the system. I will be shown in the next section how to use off-policy learning to find optimal solutions for many hyperparameters while only a behavior policy is applied to the system to collect data. Based on (6.4), once \(u^*_{\theta}(x)\) is applied to the system, to monitor the degree of robustness of specifications under \(u^*_{\theta}(x)\), one can write the conjunction over predicate functions \(P = [\sigma_1, ..., \sigma_N, \sigma_{N+1}]\) as

\[\xi_{\theta}(x, t, \sigma) = \wedge_{i=1}^{N+1} \rho^{\sigma_i}(x, t) = \min_{i \in [1, ..., N+1]} (\rho^{\sigma_1(x, t), ..., \rho^{\sigma_{N+1}(x, t)}) \tag{6.16}\]

where \(\rho^{\sigma_i}(x, t) = z^{\sigma_i}(x(t))\) and \(x(t) = x_t(x_0, u^*_{\theta}(x))\) is the state trajectory \(x(t)\) and time \(t \geq 0\) generated by (6.7) with the initial condition \(x_0\) and the policy \(u^*_{\theta}(x)\).

In order to avoid the non-smooth analysis, a smooth under approximation for \(\xi_{\theta}(x, t, \sigma)\) is provided in the following lemma.
Lemma 3. Consider a conjunction of $N + 1$ predicate functions given in (6.13) and their overall robustness $\xi(x, t, \sigma)$ defined in (6.16). Then,

$$
\xi^a(x, t, \sigma) \triangleq - \sum_{i=1}^{N+1} \ln \left[ e^{-\rho^i(x, t)} \right] \leq \xi(x, t, \sigma) \tag{6.17}
$$

Proof. See [168].

Lemma 4. The sign of the function $\xi^a(x, t, \sigma)$ is the same as the sign of the function $\xi(x, t, \sigma)$.

Proof. It is immediate from (6.17) that $\xi(x, t, \sigma) < 0$ results in $\xi^a(x, t, \sigma) < 0$. On the other hand, if $\xi(x, t, \sigma) > 0$, then based on (6.16), $\rho^i(x, t) > 0$ for all $i \in [1, ..., N + 1]$ and thus $-\ln(e^{-\rho^i(x, t)}) > 0$, $i = [1, ..., N + 1]$, which completes the proof.

Now, a fitness function for a hyperparameter vector $\theta$ as a metric for measuring the system’s safety and liveness in terms of overall robustness is defined as

$$
f_{\theta}(x(t)) = \int_t^{\infty} e^{-a(t-t)}[(1 - l) \log(\xi^a(\tau, \sigma)) + (1 + l) \log(1 + \xi^a_{\theta}(x, \tau, \sigma))]d\tau \tag{6.18}
$$

where $l = (\xi^a(x, t, \sigma))$. The first term is a barrier function to make the fitness infinity if the degree of robustness becomes negative (STL specifications are violated). On the other hand, the lower the fitness, the better the robustness of safety and liveness specifications. This is because the inverse of the degree of robustness is used in the fitness function.

Note that for the nonempty set $S_{\mu, \theta}(x)$ in (6.15), the fitness function in (6.18) becomes

$$
f_{\theta}(x(t)) = \int_t^{\infty} e^{-a(t-t)}[2 \log(1 + \xi^a_{\theta}(x, \tau, \sigma))]d\tau \tag{6.19}
$$

Theorem 1. There exists a control policy $\mu(x)$ for which $S_{\mu, \theta}(x)$ is nonempty, if and only if the fitness function in (6.18) is bounded over some set.

Proof. If the set $S_{\mu, \theta}(x)$ is empty for all $\mu(x) \in U$, then, for any initial condition $x_0$ and any $\mu(x) \in U$, one has $x_t(x_0, \mu) \neq \sigma$ and consequently $\xi(x, t, \sigma) < 0 \Rightarrow \xi^a(x, t, \sigma) < 0$ for some time $t$. This makes the fitness function (6.18) unbounded because of the first term. On the other hand, if the set $S_{\mu, \theta}(x)$ is nonempty, then for some control policy $\mu(x) \in U$ and any initial condition in $x_0 \in S_{\mu, \theta}(x)$,
Thus, \( x_t(x_0, \mu) \models \sigma, \forall t \geq 0 \). Based on Lemma 4, \( \xi^\mu_\theta(x, t, \sigma) > 0 \forall t \geq 0 \). Let \( \varepsilon_0 = \min_{t \geq 0} \{ \xi^\mu_\theta(x, t, \sigma) \} < \infty \). Then,

\[
f_\theta(x_0) \leq 2 \ln(1 + \frac{1}{\varepsilon_0}) \int_t^\infty e^{-a(t-\tau)} d\tau \leq \frac{2}{a} \ln(1 + \frac{1}{\varepsilon_0}) < \infty, \ \forall x_0 \in S_{\mu, \theta}(x)
\]

This completes the proof.

We now present an online data-based approach to learn the fitness function as a function of the state. Define the meta reward as

\[
r_m(x, t) = (1 - l) \log(\xi^\mu_\theta(x, t, \sigma)) + (1 + l) \log(1 + \xi^\mu_\theta - 1(x, t, \sigma))
\]

The fitness function corresponding to \( u_\theta^*(x) \) at one specific state can now be interpreted as the accumulated meta rewards the system receives starting from that state when \( u_\theta^*(x) \) is applied to the system, and thus it can be interpreted as a safety-related value for that state. To calculate the fitness function for all states in a set of interest, one can run the system from all those states to collect a trajectory and then calculate the fitness function. This, however, is not practical and not data efficient. To obviate this issue, the fitness function in (6.18) is written as

\[
f_\theta(x(t)) = \int_t^{t+T} e^{-a(t-\tau)} r_m(x, \tau) \ d\tau + f_\theta(x(t + T))
\]

where \( f_\theta(x(t + T)) \) is the fitness value sampled at the time \( t + T \). This equation resembles the Bellman equation in RL and its importance is that it allows us to express the fitness values of states as fitness values of other sampled states. This opens the door for value-function approximation-like approaches with function approximation for calculating the fitness value for each state. That is, since the fitness of consecutive samples are related using (6.22), a parametrized form of the fitness function or a nonparametric form of it can be used to learn the fitness function for all states of interest using only a single trajectory of the system. This will allow fast proactive decision making in the upper layer and will prevent the system from reaching an irreversible crisis, for which no action can keep the system in its safety envelope in the future. To this end, a data-based approach will assess the fitness online in real-time without using a model of the system. Once the fitness is learned, a monitor will detect changes in the fitness and consequently the situation.
We first consider learning the fitness function. Since the form of the fitness function is not known, Gaussian processes (GP) is employed to estimate the function $f_\theta(x(t))$. In analogy to GP regression for RL [173], a GP prior is first imposed over the fitness function, i.e.,

$$f_\theta(x) \sim \mathcal{N}(m_0(x), k_0(x, x'))$$

with a mean $m_0(x)$ and covariance $k_0(x, x')$. The covariance form can be chosen to reflect the prior knowledge concerning the similarity of the state’s fitness in the domain of interest. To employ GP regression, based on (6.22), the temporal difference (TD) error for fitness function is written as

$$f_\theta(x(t)) - f_\theta(x(t + T)) = \int_t^{t+T} e^{-a(\tau-t)}r_m(x, \tau)d\tau$$

(6.23)

To learn the fitness function using GP, the sequence of samples of the fitness function corresponding to the trajectory $x_1, ..., x_L$ is used to present the following generative model.

$$R(x_{t+T}) = f_\theta(x(t)) - f_\theta(x(t + T)) + \delta(t)$$

(6.24)

where

$$R(x_i) = \int_{t_{i-1}}^{t_i} e^{-a(\tau-t_{i-1})}r_m(x, \tau) \, d\tau$$

and $\delta(t) \sim \mathcal{N}(0, w^2)$ denotes zero-mean Gaussian noise indicating uncertainty on the fitness function. Note that (6.24) can be considered as a latent variable model in which the fitness function plays the role of the latent or hidden variable while the meta reward plays the role of the observable output variable. As a Bayesian method, GP computes a predictive posterior over the latent values by conditioning on observed meta rewards. Let $X^{\theta}_L = [x_1, ..., x_L]$ with $x_t = x_t(x_0, u^*_\theta(x))$ be the trajectory collected after $u^*_\theta(x)$ (the optimal control input found by the lower layer RL for the hyperparameter $\theta$) applied to the system. An algorithm for the derivation of $u^*_\theta(t)$, for the new hyperparameter $\theta$ based on the recorded history data will be given and discussed in detail later in Section V. Let define the following vectors for this finite-state trajectory of length $L$ as

$$R^\theta_L = [R(x_1), \ldots, R(x_L)]^T$$

$$f^\theta_L = [f_\theta(x_1), \ldots, f_\theta(x_L)]^T$$

$$\delta^\theta_L = [\delta(1), \ldots, \delta(L)]^T$$

(6.25)

and covariance vector and matrices are given by

139
Based on (6.25)-(6.26), one has

\[
\begin{pmatrix}
\bar{\delta}_L \\
f_{L}^\theta
\end{pmatrix}
\sim \mathcal{N}\left\{ \begin{pmatrix} 0 \\ m_0(x) \end{pmatrix}, \begin{pmatrix} \sum_L & 0 \\ 0 & K_L^\theta \end{pmatrix} \right\}
\]

(6.27)

Using (6.24), for a finite state trajectory of length \(L\) one has

\[
R_{L-1}^\theta = H_L f_{L}^\theta + \bar{\delta}_{L-1}
\]

(6.28)

where

\[
H_L = \begin{bmatrix}
1 & -1 & 0 & \ldots & 0 \\
0 & 1 & -1 & \ldots & 0 \\
\vdots & \vdots & \ddots & \ddots & \ddots \\
0 & 0 & \ldots & 1 & -1
\end{bmatrix}_{L-1 \times L}
\]

(6.29)

Based on standard results on jointly Gaussian random variable, one has

\[
\begin{pmatrix}
R_{L-1}^\theta \\
f_\theta(x)
\end{pmatrix}
\sim \mathcal{N}\left\{ \begin{pmatrix} 0 \\ m_0(x) \end{pmatrix}, \begin{pmatrix} H_L K_L^\theta H_L^T + \sum_{L-1} & H_L K(x, X_L^\theta) \\ K(x, X_L^\theta)^T H_L^T & k_0(x, x') \end{pmatrix} \right\}
\]

(6.30)

Using (6.6), the posterior distribution of the fitness function \(f_\theta(x(t))\) at state \(x(t)\), conditioned on observed integral meta reward values \(R_{t-1}^\theta\) is given by

\[
(f_\theta(x(t)) | R_{t-1}^\theta) \sim \mathcal{N}(\nu_t^\theta(x), p_t^\theta(x))
\]

(6.31)

where

\[
\nu_t^\theta(x) = m_0(x) + K(x, X_t^\theta)^T \alpha_t^\theta
\]

\[
p_t^\theta(x) = k_0(x, x) - K(x, X_t^\theta)^T C_t^\theta K(x, X_t^\theta)
\]

(6.32)

with

\[
\alpha_t^\theta = R_{t-1}^\theta T (H_t K_t^\theta H_t^T + \sum_{t-1})^{-1} H_t
\]

\[
C_t^\theta = H_t^T (H_t K_t^\theta H_t^T + \sum_{t-1})^{-1} H_t
\]

(6.33)

Based on (6.22), the following difference error is used as a surprise signal to detect changes in the fitness.

\[
SP(t) = \nu_t(x) - \nu_{t+T}(x) - R(x_t)
\]

(6.34)
where \( \nu_t(x) \) is the mean of the GP and \( SP(t) \) is the surprise signal at the time \( t \). Note that after the GP learns the fitness, the surprise signal will be small, as the GP is learned to assure that (6.22) is satisfied. However, once the robustness degree changes due to a change in the situation, the surprise signal will increase, and if the average of the surprise signal over a horizon is bigger than a threshold, a new fitness function will be learned and metacognitive decisions will be made (as explained in the next section) to improve the fitness if it is below some desired state-dependent threshold. That is, the monitor will perform an evaluation of the surprise signal in moving horizon fashion and will indicate a change if

\[
\int_{t}^{t+\Delta} SP(\tau) \, d\tau \geq \beta \tag{6.35}
\]

for some threshold \( \beta \). The metacognitive layer does not adapt the hyperparameters all the time and only adapts them when two conditions are satisfied: 1) an event indicating a change is triggered, i.e., if (6.35) is satisfied, 2) the new learned fitness is below a threshold, i.e., it does not indicate future safety and liveness.

To monitor the second requirement for detecting a threat that requires adapting the hyperparameters, inspired by [173], a KL divergence metric is defined to measure similarity the GP learned for the current fitness and a base GP. The base GP can be obtained based on the knowledge of the constraints and STL specifications to assure the minimum safety of the system. Note that constructing the base GP only requires the knowledge of the STL specifications and is independent of the system dynamics, the situation, or the control objectives. A library of safe GPs can also be constructed as based GP and the previous learned GPs for other circumstances can be added. If the fitness remains close to any of the GPs in this library, this indicates that the system safety is still not in danger. If not, it is highly likely that the system’s safety is in danger of being violated in the near future.

Since the covariance function \( K \) corresponds to a (possibly infinite-dimensional) feature space, the Gaussian process can be viewed as a Gaussian distribution in the feature space. To show this, let \( \phi(x) \) be the feature space representation of the covariance function so that \( K(x, x) = \phi(x)^T \phi(x) \). Then, we use the notation

\[
f_{\theta}(x) = GP_K(\alpha_\theta, C_\theta) \tag{6.36}
\]
to denote the GP with the corresponding covariance function $K$ and parameters $\alpha^\theta_i$ and $C^\theta_i$. We define the GP for the base fitness as

$$f_b(x) = GP_K(\alpha^b_i, C^b_i)$$  \hfill (6.37)

**Lemma 5** [174]. Let the base GP and the GP for the hyperparameters $\theta$ share the same inducing inputs, i.e., $X^b_L = X^\theta_L = [x_1, ..., x_L]$, and the same covariance function $K$. Let $K(X^\theta_L, X^b_L) = Q^{-1}$.

Then, the KL divergence between two dynamic GPs $f^\theta(x)$ and $f_b(x)$ is given by

$$D_{KL}(f^\theta(x) || f_b(x)) = D_{KL}(GP_K(\alpha^\theta_i, C^\theta_i) || GP_K(\alpha^b_i, C^b_i))$$

$$= (\alpha^\theta_i - \alpha^b_i)^T V (\alpha^\theta_i - \alpha^b_i) + W$$  \hfill (6.38)

where $V = (Q + C^\theta_i)^{-1}$ and $W = \text{Tr}[(Q + C^b_i)V] - \log((Q + C^\theta_i)V)$

**Remark 4.** Since the base fitness function is learned offline based on the minimum acceptable degree of robustness, one can select many inducing points for the base fitness function and retain only a subset of them in the expressions of the posterior mean and kernel functions that increase the similarity of the inducing inputs for both GPs. To use the KL divergence metric (38), one can use the fact that $K(x, x) = \phi(x)^T \phi(x)$, and so $K(x, x_1) = K(x, x_2)K(x_1, x_2)$, to shift the inducing points of the base fitness function to those of the learned GP.

Let $f_b(x) = [f^b_1(x), ..., f^b_M(x)]$ be the stack of base GPs. After a change, if the condition $\min_i D_{KL}(f^\theta(x) || f^b_i(x)) \leq \varpi$, with $f_b(x)$ as the fitness before change, then this indicates that the system is still safe despite the change. Therefore, the monitor now triggers an event that requires adaptation of the hyperparameters if the following STL condition is violated

$$\varphi = \Box ((\int_{t}^{t+\Delta} SP(\tau) d\tau > \beta) \land \min_i (D_{KL}(f^\theta(x) || f^b_i(x)) > \varpi))$$  \hfill (6.39)

### 6.4.1.2 Metacognitive Control

In this section, after the STL specification (6.39) is violated, a metacognitive controller is presented to find a new set of hyperparameters to guarantee that the fitness is improved, i.e., $\max_i D_{KL}(f^\theta(x) || f^b_i(x)) \leq \varpi$ with the minimum sacrifice on the performance. That is, it is desired to assure safety while achieving as much performance as possible close to the $\theta^*$ be the optimal hyperparameter found to
optimize the performance prior to the change. The metacognitive layer then performs the following optimization.

\[
\begin{align*}
\min_{\theta} & \quad \|\theta - \theta^*\| \\
\text{s.t.} & \quad \min_i D_{KL}(f_{\theta}(x)||f_{b_i}(x)) \leq \varpi
\end{align*}
\tag{6.40}
\]

To solve this optimization, one can define the survival score function in the metacognitive layer as

\[
\mathcal{H}(\theta) = \|\theta - \theta^*\| + \log(1 + \frac{1}{\varpi - \min_i D_{KL}(f_{\theta}(x)||f_{b_i}(x))})
\tag{6.41}
\]

In this chapter, safe Bayesian optimization (SBO) \cite{SBO} is used to find the optimal hyperparameters that optimize the survival score function in (6.41). SBO algorithm provided in Algorithm 1 guarantees safety by only evaluating hyperparameters that achieve a safe score threshold with high probability. This threshold is chosen as a value below which we do not want the system fitness to fall, which can be an indication of great risk of violation of specifications. SBO is a sample efficient optimization algorithm that requires only a few evaluations of the hyper performance or survival score function to find the optimal hyperparameters. While the safe set of hyperparameters is not known in the beginning, it is estimated after each function evaluation of SBO. In fact, at each iteration, SBO tries to not only find the global maximum within the currently known safe set (exploitation), but also to increase the set of hyperparameters that are known to be safe (exploration) as described in Algorithm 1. More specifically, the SBO builds a surrogate model \(\mathcal{P}\) that maps the hyperparameters to the survival score \(\mathcal{H}(\theta)\) in the metacognitive layer and express as

\[
\mathcal{P} : \mathcal{D}' \rightarrow \mathbb{R}
\tag{6.42}
\]

where \(\mathcal{D}'\) denotes the bounded domain of hyperparameters.

Note that the latent function \(\mathcal{P}\) in (6.42) is unknown, as the dynamics are not known. \(\mathcal{P}\) can be sampled by running the system and evaluating the survival score function \(\mathcal{H}(\theta)\) from the recorded data. These samples of \(\mathcal{P}\) are generally uncertain because of noisy data and the score function is typically nonconvex, and no gradients are easily available.

Now, we used GP to perform a non-parametric regression for the latent function \(\mathcal{P}\) \cite{GP} and then SBO is used to optimize the survival score and determine the optimum values of hyperparameters. For the non-parametric regression, we define a prior mean function \(\mu_0(\theta)\), which encodes prior knowledge about the survival score function \(\mathcal{P}(\theta)\), and a covariance function \(k(\theta, \theta')\) which
defines the covariance of any two function values, \( P(θ) \) and \( P(θ') \) are used to model the uncertainty about the mean estimates. One can predict the survival score of the system corresponding to the hyperparameter \( θ \) by calculating its mean \( μ_k(θ) \) and covariance \( σ_k^2(θ) \) over a set of \( k \) observations \( \{θ_1:k, P_1:k\} \) and given by \( P(θ) \sim N(μ_k(θ), σ_k^2(θ)) \).

Based on the predicted score function, i.e., \( P(θ) \sim N(μ_k(θ), σ_k^2(θ)) \), the lower and upper bound of the confidence interval at iteration \( k \) is given as

\[
\begin{align*}
\bar{m}_k(θ) &= μ_k(θ) - β_kσ_k(θ) \\
M_k(θ) &= μ_k(θ) + β_kσ_k(θ)
\end{align*}
\]  

(6.43)

where \( β_k > 0 \) denotes a scalar factor that defines the desired confidence interval. Based on (6.43), the safe set of all the hyperparameters \( θ \) that lead to survival score values above the threshold \( P_{min} \) is given by

\[
S_k \leftarrow \{θ \in D'| \bar{m}_k ≥ P_{min}\}
\]  

(6.44)

Then, a set of potential maximizers is defined as [175]

\[
T_k \leftarrow \{θ \in S_k| M_k(θ) ≥ \max_θ \bar{m}_k(θ)\}
\]  

(6.45)

which contains all the safe set of hyperparameters for which the upper confidence interval \( M_k(θ) \) is above the best safe lower bound. In order to define, a set of potential expanders which quantifies whether a new set of hyperparameters can be classified as safe for a new observation, an optimistic characteristic function for expanders is given as

\[
g_k(θ) = \{θ' \in D'| S_k| \bar{m}_{k,(θ,M_k(θ))}(θ') ≥ P_{min}\}
\]  

(6.46)

where \( \bar{m}_{k,(θ,M_k(θ))} \) is the lower bound of the GP, based on prior data and a data point \( (θ, M_k(θ)) \) with a noiseless measurement of the upper confidence bound. Based on (6.46), one can determine how many previously unsafe points can be classified as safe according to (6.44) assuming that we measure \( M_k(θ) \) while evaluating \( P(θ) \). The characteristic function is positive if the new data point has a non-negligible chance to expand the safe set. Therefore, the set of possible expanders is given as

\[
G_k \leftarrow \{θ \in S_k| g_k(θ) > 0\}
\]  

(6.47)
Then, a new set of hyperparameters is selected to evaluate the performance on the real system by selecting the set of hyperparameters about which we are the most uncertain from the union of the sets $G_k$ and $M_k$, i.e., at iteration $k$ the score function is evaluated at $\theta_k$

$$\theta_k \leftarrow \arg\max_{\theta \in \{G_k \cup T_k\}} (M_k(\theta) - \bar{m}_k(\theta))$$

The evaluation approach in (6.48) works well for expanding the safe set [175], with a trade-off between exploration and exploitation. For the exploration, the most uncertain parameter locations are usually on the boundary of the safe set, which results in efficient exploration. An estimate of the best currently known set of hyperparameters is obtained from $\arg\max \bar{m}_k(\theta), \forall \theta \in S_k$ which corresponds to the point that achieves the best lower bound on the survival score.

Algorithm 4 SBO for Metacognitive Control.

1: procedure
2: Initialize GP with $(\theta_0, P(\theta_0))$
3: for $k = 1, \ldots$ do
4: $S_k \leftarrow \{\theta \in D | \bar{m}_k \geq P_{\text{min}}\}$ where $\bar{m}_k(\theta) = \mu_k - 1(\theta) - \beta_k \sigma_k - 1(\theta)$
5: $T_k \leftarrow \{\theta \in S_k | M_k(\theta) \geq \max_{\theta'} \bar{m}_k(\theta')\}$ where $M_k(\theta) = \mu_k - 1(\theta) + \beta_k \sigma_k - 1(\theta)$
6: $G_k \leftarrow \{\theta \in S_k | g_k(\theta) > 0\}$ where $g_k(\theta) = \{\theta' \in D \setminus S_k | m_k(\theta, M_k(\theta))(\theta') \geq P_{\text{min}}\}$
7: $\theta_k \leftarrow \arg\max_{\theta \in \{G_k \cup T_k\}} (M_k(\theta) - \bar{m}_k(\theta))$
8: Obtain measurement $P(\theta_k)$
9: Update GP with $(\theta_k, P(\theta_k))$
10: end for
11: end procedure

Remark 5. It is shown in [176]-[177] that, given a persistently exciting input, a single rich measured trajectory can be used to characterize the entire system trajectories. That is, having a single rich trajectory of an unknown system, the trajectory for a given sequence of inputs and an initial condition can be constructed without even applying it to the system. This can be leveraged to learn the fitness function for a given control policy selected by the Bayesian optimization algorithm without actually applying it to the system. More specifically, after a change, to evaluate the fitness of hyperparameters, a rich trajectory of the system is first collected and then used to reconstruct
the trajectory of the system with enough length for any hyperparameter under evaluation. This trajectory data can then be used to learn the GP for the hyperparameters without even applying its corresponding controller. This will allow us to even evaluate unsafe policies without even applying them to the system. Hence, for each set of hyperparameters, one can compute the fitness function from measured data without knowledge of the closed-loop system’s dynamics, and consequently, find the optimal hyperparameters that optimize the survival score function in (40). This resembles off-policy learning in RL.

Remark 6. Note that SBO is a derivative-free optimization algorithm, which is useful since a closed-form expression of \( P \) as a function of the hyperparameters \( \theta \) is not available. Also, it allows us to tune the hyperparameter with as few evaluations of \( P \) as possible which is crucial since each evaluation can be costly and time-consuming, as it requires a closed-loop experiment.

6.5 Low-Level RL-Based Control Architecture

In this section, a computational data-driven algorithm is first developed to find the new optimal control policy \( u^*_\theta(t) \), for the new hyperparameter \( \theta \) based on recorded history. It is then shown that the proposed algorithm converges to the optimal control policy \( u^*_\theta(t) \) for all admissible hyperparameters \( \theta \).

Following the same reasoning from [171], we are ready to give the following computational data-driven algorithm for finding the new optimal control policy, i.e., \( u^*_\theta(t) \), for a new hyperparameter \( \theta \) based on the recorded history data.

Remark 7. Note that Algorithm 2 does not rely on the dynamics of the system. Note also that, inspired by the off-policy algorithm in [171], Algorithm 2 has two separate phases. In the first phase, i.e., Step 2, a fixed initial exploratory control policy \( u \) is applied, and the system information is recorded over the time interval \([t_0, t_l]\). In the second phase, i.e., Steps 3-7, then, without requiring any knowledge of the system dynamics, the information collected in the first phase is repeatedly used to find a sequence of updated policies converging to \( u^* \).

The following theorem shows that Algorithm 2 converges to the optimal control policy \( u^*_\theta(t) \) for all admissible hyperparameters \( \theta \).
Algorithm 5 Low-level Off-policy RL-based Control.

1: procedure
2: Perform an experiment on the time interval $[t_0, t_f]$ by applying fixed stabilizing control policy $u(t) + e$ to the system, where $e$ is the exploration noise, and then record \{$X_\theta(t)$\} and corresponding \{$u(t)$\} at $N \geq l_1 + m \times l_2$ different sampling in the time interval $[t_0, t_f]$, where $X_\theta(t) = [e_d(t)^T, r_\theta^T]^T e_d(t) := x(t) - r_\theta$.
3: For a new parameters of $\theta$ where $\theta \in \lambda$, construct $\phi_1(X_\theta) \in \mathbb{R}^{l_1}$ and $\Phi(X_\theta) \in \mathbb{R}^{l_2}$ as suitable basis function vectors.
4: Compute $\Xi_k(\theta)$ and $\Theta_k(\theta)$ for a new set of parameters $\theta$ based on the recorded history data as follows:
\[
\Xi_k(\theta) = -I_{xx}(\theta) \begin{bmatrix} vec(\bar{Q}_\theta) \\ vec(R_\theta) \end{bmatrix} \tag{6.49}
\]
and
\[
\Theta_k(\theta) = [\delta_{xx}, -2\hat{I}_{xx}(\theta)(I_n \otimes \hat{W}_k^T R_\theta) - 2I_{xu}(\theta)(I_n \otimes R_\theta)] \tag{6.50}
\]
where $\bar{Q}_\theta := \text{diag}(Q_\theta, 0)$ and
\[
\delta_{xx} = [e^{-\gamma T}(\phi_1(X_\theta(t_1)) - \phi_1(X_\theta(t_0))), e^{-\gamma T}(\phi_1(X_\theta(t_2)) - \phi_1(X_\theta(t_1))), \ldots, e^{-\gamma T}(\phi_1(X_\theta(t_l)) - \phi_1(X_\theta(t_{l-1})))]^T \tag{6.51}
\]
\[
\hat{I}_{xx}(\theta) = \left[ \int_{t_0}^{t_1} e^{-\gamma(\tau-t)}(\Phi(X_\theta) \otimes \Phi(X_\theta))d\tau, \int_{t_1}^{t_2} e^{-\gamma(\tau-t)}(\Phi(X_\theta) \otimes \Phi(X_\theta))d\tau, \ldots, \int_{t_{l-1}}^{t_l} e^{-\gamma(\tau-t)}(\Phi(X_\theta) \otimes \Phi(X_\theta))d\tau \right]^T \tag{6.52}
\]
\[
I_{xx}(\theta) = \begin{bmatrix}
[\int_{t_0}^{t_1} e^{-\gamma(\tau-t)}X_\theta \otimes X_\theta d\tau, \\
\int_{t_1}^{t_2} e^{-\gamma(\tau-t)}X_\theta \otimes X_\theta d\tau, \\
\cdots, \int_{t_{l-1}}^{t_l} e^{-\gamma(\tau-t)}X_\theta \otimes X_\theta d\tau] \\
[\int_{t_0}^{t_1} e^{-\gamma(\tau-t)}(\hat{W}_k^T \Phi(X_\theta) \otimes \hat{W}_k^T \Phi(X_\theta))d\tau, \\
\int_{t_1}^{t_2} e^{-\gamma(\tau-t)}(\hat{W}_k^T \Phi(X_\theta) \otimes \hat{W}_k^T \Phi(X_\theta))d\tau, \\
\cdots, \int_{t_{l-1}}^{t_l} e^{-\gamma(\tau-t)}(\hat{W}_k^T \Phi(X_\theta) \otimes \hat{W}_k^T \Phi(X_\theta))d\tau]
\end{bmatrix}^T \tag{6.53}
\]
\[
I_{xu}(\theta) = \left[ \int_{t_0}^{t_1} e^{-\gamma(\tau-t)}(\Phi(X_\theta) \otimes u)d\tau, \int_{t_1}^{t_2} e^{-\gamma(\tau-t)}(\Phi(X_\theta) \otimes u)d\tau, \ldots, \int_{t_{l-1}}^{t_l} e^{-\gamma(\tau-t)}(\Phi(X_\theta) \otimes u)d\tau \right]^T \tag{6.54}
\]
5: Solve $\hat{W}_k^V \in \mathbb{R}^{l_1}$ and $\hat{W}_{k+1} \in \mathbb{R}^{l_2 \times m}$ from (6.55)
\[
\begin{bmatrix}
\hat{W}_k^V \\
\text{vec}(\hat{W}_{k+1}^T)
\end{bmatrix} = (\Theta_k^T(\theta)\Theta_k(\theta))^{-1}\Theta_k^T(\theta)\Xi_k(\theta)
\]  
(6.55)

and update the value function and control policy as follows:

\[
\hat{V}_k^\theta(X_\theta) := (\hat{W}_k^V)^T\phi_1(X_\theta) 
\]  
(6.56)

and

\[
u_k^{k+1}(t) := \hat{W}_{k+1}^T\Phi(X_\theta)
\]  
(6.57)

6: Let \(k \leftarrow k + 1\), and go to Step 5 until \(\|\hat{W}_k^V - \hat{W}_{k-1}^V\| \leq \varepsilon\) for \(k \geq 1\), where the constant \(\varepsilon > 0\) is a predefined small threshold.

7: Use \(u_\theta^*(t) := u_k^{k+1}(t)\) and \(V_\theta^*(X_\theta) := \hat{V}_k^\theta(X_\theta)\) as the approximated optimal control policy and its approximated optimal value function corresponding to a new set of \(\theta\), respectively.

8: \textbf{end procedure}

\textbf{Theorem 2. Convergence of Algorithm 2.} Let the new hyperparameter \(\theta\) be admissible. Using the fixed stabilizing control policy \(u(t)\), when \(N \geq l_1 + m \times l_2\), \(u_k^{k+1}(t)\) obtained from solving (6.55) in the off-policy Algorithm 2, converges to the optimal control policy \(u_\theta^*(t)\), \(\forall \theta \in \lambda\).

\textit{Proof.} \(\forall \theta \in \lambda\), set \(Q_\theta\), \(R_\theta\), and \(r_\theta\). Using (6.49)-(6.54), it follows from (6.55) that \(\hat{W}_k^V \in \mathbb{R}^{l_1}\) and \(\hat{W}_{k+1} \in \mathbb{R}^{l_2 \times m}\) satisfy the following Bellman equation.

\[
e^{-\gamma T} \hat{W}_k^T(\phi_1(X_\theta(t + \delta t)) - \phi_1(X_\theta(t))) \\
= -\int_t^{t+\delta t} e^{-\gamma (\tau-t)}((X_\theta)^TQ_\theta(X_\theta)) + u_k^T R_\theta u_k) d\tau \\
+ \int_t^{t+\delta t} e^{-\gamma (\tau-t)}(-2(\hat{W}_{k+1}^T\Phi(X_\theta))R_\theta(u - u_k)) x d\tau
\]  
(6.58)

Now, let \(W^V \in \mathbb{R}^{l_1}\) and \(\hat{W} \in \mathbb{R}^{l_2 \times m}\) such that \(u_k = \hat{W}^T\Phi(X_\theta)\) and

\[
\begin{bmatrix}
W^V \\
\text{vec}(\hat{W}^T)
\end{bmatrix} = (\Theta_k^T(\theta)\Theta_k(\theta))^{-1}\Theta_k^T(\theta)\Xi_k(\theta)
\]  
(6.59)

Then, one immediately has \(W^V = \hat{W}_k^V\) and \(\text{vec}(\hat{W}) = \text{vec}(\hat{W}_{k+1}^T)\). If the condition given in Step 2 is satisfied, i.e., the information is collected at \(N \geq l_1 + m \times l_2\) points, then \([\hat{W}_k^V, \text{vec}(\hat{W}_{k+1}^T)]^T\) has \(N\) independent elements, and therefore the solution of least squares (LS) in (6.55) and (6.59) are equal and unique. That is, \(\hat{W}_k^V = W^V\) and \(\hat{W}_{k+1} = \hat{W}\). This completes the proof. \(\square\)
6.6 Simulation Results

The presented algorithm is validated for the steering control of an autonomous vehicle in the lane changing scenario. Consider the following vehicle dynamics for steering control as [178]

\[
\dot{x}(t) = Ax(t) + Bu(t) \tag{6.60}
\]

with

\[
A = \begin{bmatrix}
0 & v_T & v_T & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & -\frac{k_f + k_r}{mTv_T} & -\frac{mTv_T + \frac{ak_f - bk_r}{v_T}}{v_T} \\
0 & 0 & -\frac{ak_f - bk_r}{I_T} & -\frac{a^2k_f + b^2k_r}{ITv_T}
\end{bmatrix}, \quad B = \begin{bmatrix}
0 \\
0 \\
\frac{mTv_T}{k_f} \\
\frac{ak_f}{I_T}
\end{bmatrix} \tag{6.61}
\]

where \(x(t) = [x_1 \ x_2 \ x_3 \ x_4]^T = [y \ \psi \ \alpha \ \dot{\psi}]^T\). The state variables are defined as lateral position of vehicle \(y\), yaw angle \(\psi\), slip angle \(\alpha\) and rate of change of yaw angle \(\dot{\psi}\). Moreover, \(\delta\) represents steering angle and act as the control input. \(v_T\) denotes the longitudinal speed of the vehicle. \(m_T\) is the total mass of the vehicle and \(I_T\) is its moment of inertia with respect to the center of the mass. \(k_f\) and \(k_r\) denote the stiffness parameters of the tire. Finally, \(a\) and \(b\) show the distance of front and rear tires to the center of the mass. The value of vehicle parameters used in the simulation are provided in Table I.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(m_T)</td>
<td>1300 kg</td>
<td>(k_f)</td>
<td>91000 N/rad</td>
</tr>
<tr>
<td>(I_T)</td>
<td>10000 m².kg</td>
<td>(k_r)</td>
<td>91000 N/rad</td>
</tr>
<tr>
<td>(v_T)</td>
<td>16 m/s</td>
<td>(\beta)</td>
<td>2</td>
</tr>
<tr>
<td>(a)</td>
<td>1.6154 m</td>
<td>(b)</td>
<td>1.8846 m</td>
</tr>
</tbody>
</table>

For the validation of the presented algorithm, lane changing scenario for two lane highway is consider as shown in Fig. 6.2. In this simulation following STL constraint is considered, the vehicle state is subject to desired specification on the offset from center line, i.e., \(\varphi = \square (|x_1 - r| < 1)\)
Fig. 6.2: Lane changing scenario for steering control of autonomous vehicle.

Fig. 6.3: Lane changing with fixed value of hyperparameter without any change in dynamics.

with \( r \) as the center lane trajectory (act as set point value). In the simulation, set point value is selected as \( r = 1 \) for \( t < 4s \), otherwise \( r = 3 \). Fig. 6.3 shows the result for lane changing with fixed values of hyperparameters and without any change in system dynamics. The control policy or steering angle in (60) is evaluated based on off-policy RL algorithm in [171] with hyperparameter values as \( Q = diag(10, 10, 10, 10) \) and \( R = 2 \). Then, in Fig. 6.4, we consider the change in the system dynamics after \( t=4s \) and it becomes

\[
\dot{x}(t) = (A + \Delta A)x(t) + (B + \Delta B)u(t)
\] (6.62)
with

\[
\Delta A = \begin{bmatrix}
0 & \Delta v_T & \Delta v_T & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & -\frac{k_f+k_r}{m_T\Delta v_T} & \frac{m_T\Delta v_T^+}{\Delta v_T} \\
0 & 0 & 0 & \frac{m_T\Delta v_T^+}{a^2k_f+b^2k_r} \\
0 & 0 & 0 & -\frac{m_T\Delta v_T^+}{I_T}\Delta v_T \\
\end{bmatrix},
\]

\[
\Delta B = \begin{bmatrix}
0 \\
0 \\
k_f \\
0 \\
\end{bmatrix}
\]

(6.63)

The vehicle parameter values for (6.63) are provided in Table I. Control input is evaluated based on off-policy RL algorithm in [171] with fixed hyperparameter values as \( Q = diag(10, 10, 10, 10) \) and \( R = 2 \). The result in Fig. 6.4 shows that the vehicle starts wavering and goes out of the lane. That is, vehicle violates the desired specification, i.e., \( \varphi = \square (|x_1 - r| < 1) \) after change in the dynamics.

Now, in order to implement the presented algorithm, first the fitness function in (6.22) is learned as GP based on the temporal difference in (6.23). Note the fitness function is learned offline and it is implemented for online monitoring and control in meta-cognitive layer. Figs. 6.5 and 6.6 show the vehicle trajectories in Figs. 6.3 and 6.4. Based on the result in predicted fitness function based on the learned GP for the vehicle trajectories in Figs. 6.3 and 6.4. Based on the result in vehicle trajectories in Figs. 6.3 and 6.4. Based on the result in Figs. 6.6 and 6.7, one can see how the fitness value grow due to the operation of the system close or beyond the desired STL specification. The fitness value is used for meta-cognitive monitoring and intermittent evaluation of the meta-cognitive control layer.

\[\text{Figure 6.4: Constraint violation during lane changing with fixed value of hyperparameter and change in dynamics.}\]
Based on the meta-cognitive monitor in (39), Algorithm 1 is evaluated using the survival score function in (41) and determine the optimum hyperparameter to ensure the desired STL specification, i.e., $\varphi = \square (|x_1 - r| < 1)$. Fig. 6.8 shows the vehicle trajectory with hyperparameter adaptation.
based on Algorithm 1 with change in dynamics. The new optimum hyperparameter values are found to be $Q = diag(96.11, 1.2, 1.15)$ and $R = 1$. Also, Fig. 6.9 shows how the predicted fitness value converges close to zero after the hyperparameter adaptation and overall fitness value becomes constant as shown in Fig. 6.10.

![Vehicle trajectory with hyperparameter adaptation based on Algorithm 1 for lane changing scenario.](image)

Figure 6.8: Vehicle trajectory with hyperparameter adaptation based on Algorithm 1 for lane changing scenario.

![Predicted fitness corresponding to vehicle trajectory with hyperparameter adaptation based on Algorithm 1.](image)

Figure 6.9: Predicted fitness corresponding to vehicle trajectory with hyperparameter adaptation based on Algorithm 1.

The presented algorithm is employed to learn control solutions with good enough performances while satisfying desired specifications and properties expressed in terms of STL. As shown in Figs. 6.8 and 6.9, based on meta-cognitive layer hyperparameters are adapted and lane changing problem for autonomous vehicle is solved without violating any constraint.
Figure 6.10: Overall fitness value under desired STL constraint for adapted vehicle trajectory.

6.7 Conclusion

In this chapter, an assured metacognitive RL-based autonomous control framework is presented to learn control solutions with good enough performances while satisfying desired specifications and properties expressed in terms of STL. We discussed that the pre-specified reward functions cannot guarantee the satisfaction of the desired specified constraints and properties across all circumstances that an uncertain system might encounter. That is, the system either violates safety specifications or achieves no optimality and liveness specifications. In order to overcome this issue, learning what reward functions to choose to satisfy desired specifications and to achieve a good enough performance across variety of circumstances, a metacognitive decision-making layer is presented in augmentation with the performance-driven layer. More specifically, an adaptive reward function is presented in terms of its gains and adaptive reference trajectory (hyperparameters), and these hyperparameters are determined based on metacognitive monitor and control to assure the satisfaction of the desired STL safety and liveness specifications. The proposed approach separates learning the reward function that satisfies specifications from learning the control policy that maximizes the reward and thus allows us to evaluate as many hyperparameters as required using reused data collected from the system dynamics.
CHAPTER 7
CONCLUSION AND FUTURE WORK

This dissertation analyzed the adverse effects of attacks and designed resilient distributed control mechanisms for multi-agent cyber-physical systems with guaranteed performance and consensus under mild assumptions. The effectiveness of the developed approach is certified by applying it to distributed frequency and voltage synchronization of AC microgrids under data manipulation attacks. Then, the adverse effects of cyber-physical attacks on distributed sensor networks are analyzed and attack mitigation mechanism for the event-triggered distributed Kalman filter is presented. It is shown that although event-triggered mechanisms are highly desirable, the attacker can leverage the event-triggered mechanism to cause triggering misbehaviors which significantly harms the network connectivity and performance. Then, an entropy estimation-based attack detection and mitigation mechanisms are designed. Finally, the safe reinforcement learning framework for autonomous control systems under constraints is developed. Reinforcement learning agents with pre-specified reward functions cannot provide guaranteed safety across variety of circumstances that an uncertain system might encounter. To guarantee performance while assuring the satisfaction of safety constraints across a variety of circumstances, an assured autonomous control framework is designed by empowering RL algorithms with meta-cognitive learning capabilities.

The following are some of the directions for the future work

• A possible direction for future work is to extend the results of resilient control designs to synchronization of DMASs with heterogeneous nonlinear dynamics. Since nonlinear systems can exhibit finite-time escape behavior, a problem of interest is to find the conditions under which the attacker can make the trajectories of agents unbounded in finite time and to obtain detection and mitigation mechanisms to counteract such attacks fast and thus avoid instability.

• Another possible direction is to extend the presented results to the containment control problem for which there exists more than one leader or exo-system with different dynamics under network uncertainties.
• Extension to meta-cognitive resilient design, i.e., a combination of high-level rules with autonomy can be the next level of resiliency which allows the system to learn and adapt from adversarial situations and acquire a level of resiliency against the unforeseen using some prior knowledge.

• Also possible direction for future work is to extend the results of safe reinforcement learning to a safe learning-based control framework with conflicting constraints.
A.1 Proof of Theorem 1 in Chapter 5

Note that for the notional simplicity, in the following proof, we keep the sensor index $i$ but ignore the time-indexing $k$. Without the time index, we represent the prior at time $k + 1$ as $ar{x}_i^a(k + 1) = (\bar{x}_i^a)^+$ and follow the same for other variables.

Using the process dynamics in (5.1) and the corrupted prior state estimate in (5.17), one has

$$
(\tilde{\eta}_i^a)^+ = x^+ - (\tilde{x}_i^a)^+ = A(x - \hat{x}_i^a) + w, 
$$

where the compromised posterior state estimate $\hat{x}_i^a(k)$ follows the dynamics (5.17). Similarly, using (5.17), the corrupted posterior state estimation error becomes

$$
\eta_i^a = x - \hat{x}_i^a = x - \bar{x}_i^a - K_i^a(y_i - C\bar{x}_i^a) - \gamma_i \sum_{j \in N_i} (\tilde{x}_j^a - \tilde{x}_i^a) - K_i^a f_i. 
$$

Then, one can write (1)-(2) as

$$
\begin{cases}
(\tilde{\eta}_i^a)^+ = A\eta_i^a + w, \\
\eta_i^a = (I_n - K_i^a C_i)\tilde{\eta}_i^a - K_i^a v_i + u_i^a,
\end{cases}
$$

where

$$
u_i^a = \gamma_i \sum_{j \in N_i} (\tilde{\eta}_j^a - \tilde{\eta}_i^a) - K_i^a f_i.
$$

Based on (5.4), we define the predictive state estimation error, respectively, under attack as

$$
(\tilde{\eta}_i^a)^+ = x^+ - (\tilde{x}_i^a)^+ = \zeta_i^+ (\tilde{\eta}_i^a)^+ + (1 - \zeta_i^+) A\tilde{\eta}_i^a.
$$

Using (3), the corrupted covariance of the prior estimation error becomes

$$
(\tilde{P}_i^a)^+ = E\left[(\tilde{\eta}_i^a)^+((\tilde{\eta}_i^a)^+)^T\right],
$$

$$
= E\left[(A\eta_i^a + w)(A\eta_i^a + w)^T\right] = A\tilde{P}_i^a A^T + Q.
$$

Using the corrupted predictive state estimate error $(\tilde{\eta}_i^a)^+$ in (5) with $(\tilde{P}_{i,j}^a)^+ = A\tilde{P}_{i,j}^a A^T + Q$, one can write the cross-correlated predictive state estimation error covariance $(\tilde{P}_{i,j}^a)^+$ as

$$
(\tilde{P}_{i,j}^a)^+ = E\left[(\tilde{\eta}_i^a)^+((\tilde{\eta}_j^a)^+)^T\right] = \zeta_i^+ (1 - \zeta_j^+) A(\tilde{P}_{i,j}^a)^+ + (1 - \zeta_i^+ \zeta_j^+ (\tilde{P}_{i,j}^a)^+ A^T + Q,
$$

158
where $\tilde{P}_{i,j}$ and $\tilde{P}_{i,j}^a$ be the cross-correlated estimation error covariances and their updates are given in (8)-(9).

The cross-correlated estimation error covariance $(\tilde{P}_{i,j}^a)^+$ in (7) is given by

$$
(\tilde{P}_{i,j}^a)^+ = E \left[ (\eta_i^a)^+(\eta_j^a)^+ \right] = \zeta_i^+ (\tilde{P}_{i,j}^a)^+ + (1 - \zeta_i^+) A \sum_{r \in N_i} (\tilde{P}_{i,r}^a - \tilde{P}_{i,j}^a)(\gamma_i A)^T + (1 - \zeta_i^+) \left[ A \tilde{P}_{i,j} M_i^a A^T + Q \right],
$$

(8)

where $\tilde{P}_{i,j}^a$ and $\tilde{P}_{i,j}^a$ denote the cross-correlated estimation error covariances evolve according to (7) and (9). Similarly, $(\tilde{P}_{i,j}^a)^+$ is updated based on the expression given by

$$
(\tilde{P}_{i,j}^a)^+ = E \left[ (\eta_i^a)^+(\eta_j^a)^+ \right] = \zeta_j^+ (\tilde{P}_{i,j}^a)^+ + (1 - \zeta_j^+) [A(M_i^a)^T \tilde{P}_{i,j}^a A^T + Q] + (1 - \zeta_j^+) A \gamma_i \sum_{s \in N_i} (\tilde{P}_{s,j}^a - \tilde{P}_{i,j}^a) A^T.
$$

(9)

Now using (2)-(5), one can write the covariance of posterior estimation error $\tilde{P}_i^a$ as

$$
\tilde{P}_i^a = E[M_i \tilde{\eta}_i^a(M_i \tilde{\eta}_i^a)^T] + E[K_i^a v_i(K_i^a v_i)^T] - 2E[(M_i \tilde{\eta}_i^a)(K_i^a v_i)^T] - 2E[K_i^a v_i(\gamma_i u_i^a)^T] + E[(\gamma_i u_i^a)^T] + 2E[(M_i \tilde{\eta}_i^a)(\gamma_i u_i^a)^T],
$$

(10)

Using (6) and measurement noise covariance, the first two terms of (10) become

$$
E[M_i \tilde{\eta}_i^a(M_i \tilde{\eta}_i^a)^T] = M_i \tilde{P}_i^a M_i^T, \quad E[K_i^a v_i(K_i^a v_i)^T] = K_i^a R_i(K_i^a)^T.
$$

(11)

According to Assumption 1, the measurement noise $v_i$ is i.i.d. and uncorrelated with state estimation errors, therefore, the third and fourth terms in (10) become zero. Now $u_i^a$ in (4) and Assumption 1, the last two terms in (10) can be simplified as

$$
E[(u_i^a)(u_i^a)^T] = \gamma_i^2 E \left[ \left| \sum_{j \in N_i} (\tilde{\eta}_j^a - \tilde{\eta}_i^a) \right| \left| \sum_{j \in N_i} (\tilde{\eta}_j^a - \tilde{\eta}_i^a) \right| \right] + E[K_i^a f_i(K_i^a f_i)^T] - 2K_i^a E[f_i \sum_{j \in N_i} (\tilde{\eta}_j^a - \tilde{\eta}_i^a)^T],
$$

(12)

and

$$
2E[(u_i^a)(M_i \tilde{\eta}_i^a)^T] = 2E[(\gamma_i \sum_{j \in N_i} (\tilde{\eta}_j^a - \tilde{\eta}_i^a) - K_i^a f_i)(M_i \tilde{\eta}_i^a)^T],
$$

(13)

$$
= 2\gamma_i \sum_{j \in N_i} (\tilde{P}_{i,j}^a - \tilde{P}_i^a) M_i^T - 2K_i^a E[f_i(\tilde{\eta}_i^a)^T] M_i^T,
$$

159
where the cross-correlated term $\tilde{P}^a_{i,j}$ is updated according to (8). Using (10)-(13), the posterior state estimation error $P^a_i$ under attacks is given by

$$
\hat{P}^a_i = M^a_i \tilde{P}^a_i (M^a_i)^T + K^a_i[R_i + \Sigma^f_i](K^a_i)^T - 2K^a_i \Xi_f \\
+ 2\gamma_i \sum_{j \in N_i}(\tilde{P}^a_{i,j} - \tilde{P}^a_i) (M^a_i)^T + \gamma_i^2(\sum_{j \in N_i}(\tilde{P}^a_j - 2\tilde{P}^a_{i,j} + \tilde{P}^a_i),
$$

with $\Xi_f = [E[f_i \sum_{j \in N_i}(\tilde{\eta}^a_j - \tilde{\eta}^a_i)^T]) + E[f_i(\tilde{\eta}^a_i)^T](M^a_i)^T].$ This completes the proof.
BIBLIOGRAPHY


inverter-based microgrids with consensus based distributed voltage control,” IEEE Trans. Con-


[50] A. Bidram, A. Davoudi, F. L. Lewis, and J. M. Guerrero, “Distributed cooperative control of
microgrids using feedback linearization,” IEEE Trans. Power Syst., vol. 28, no. 3, pp. 3462-

[51] A. Bidram, F. L. Lewis, and A. Davoudi, “Distributed control systems for small-scale power
34, no. 6, pp. 56-77, Nov. 2014.

14, no. 5, pp. 2052-2061, May 2018.


distributed cooperative control of multi-agent systems,” IET Generation, Transmission, Dis-

secondary voltage and frequency control for islanded microgrids,” IEEE Trans. Smart Grid,
vol. 10, no. 4, pp. 3780-3790, July 2019.

inverter-interfaced microgrids based on consensus algorithm with improved transient perfor-

cyber resilient and secure microgrid using software-defined networking,” IEEE Trans. Smart

[59] X. Liu and Z. Li, "False data attacks against AC state estimation with incomplete network

[60] O. Kosut, L. Jia, R. J. Thomas, and L. Tong, "Limiting false data attacks on power system


