

Robust and Adaptive Anomaly Detection for Distributed Slice Mobility Attacks in 5G Network Slicing

Ricardo Misael Ayala Molina

A Thesis

in

The Department

of

Concordia Institute for Information Systems Engineering (CIISE)

Presented in Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy (Information and Systems Engineering) at

Concordia University

Montréal, Québec, Canada

March 2026

© Ricardo Misael Ayala Molina, 2026

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Mr. Ricardo Misael Ayala Molina**

Entitled: **Robust and Adaptive Anomaly Detection for Distributed Slice Mobility
Attacks in 5G Network Slicing**

and submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy (Information and Systems Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

Dr. Mohsen Ghafouri Chair

Dr. Richard Pazzi External Examiner

Dr. Jun Yan Examiner

Dr. Carol Fung Examiner

Dr. Chadi Assi Supervisor

Approved by

Chun Wang, Chair
Department of Concordia Institute for Information Systems Engineering (CIISE)

2026

Mourad Debbabi, Dean
Gina Cody School of Engineering and Computer Science

Abstract

Robust and Adaptive Anomaly Detection for Distributed Slice Mobility Attacks in 5G Network Slicing

Ricardo Misael Ayala Molina, Ph.D.

Concordia University, 2026

The evolution of Fifth Generation (5G) technology has introduced network slicing as a pivotal innovation, enabling the creation of virtualized and logically isolated networks that operate over a shared physical infrastructure. Each Network Slice (NS) is tailored to specific application requirements, ensuring consistent Quality of Service (QoS) across diverse services. This architectural flexibility allows User Equipment (UE) to simultaneously connect to or switch between multiple slices through the Inter-Slice Switching (ISS) procedure to access varied services efficiently. However, this capability also exposes a critical vulnerability that can be exploited to launch a Distributed Slice Mobility (DSM) attack, which is a specialized form of Distributed Denial of Service (DDoS) that overwhelm the 5G control plane with excessive signaling traffic.

This thesis investigates advanced anomaly detection approaches to enhance the resilience of 5G NSs against DSM attacks under both static and evolving conditions. It first establishes a data-driven detection framework based on Long Short-Term Memory (LSTM)-Autoencoders, capable of capturing temporal dependencies in Third Generation Partnership Project (3GPP) Key Performance Indicators and Performance Measurement counters that characterize the control plane's behavior during normal and attack scenarios. To reflect the realities of 5G network environments where training data is often contaminated with mixed benign and malicious samples, the study integrates Positive-Unlabeled Learning (PUL) into the detection process, enabling accurate differentiation between normal and attack traffic without requiring exhaustive labeling.

As 5G networks evolve and adversaries adapt their strategies, the research further introduces

an adaptive detection framework designed to address incremental concept drift in DSM attack patterns. Leveraging a Conditional Wasserstein Generative Adversarial Network with Gradient Penalty (CWGAN-GP) to simulate drifted DSM traffic distributions, the framework integrates continual learning with latent feature extraction and clustering to ensure coordinated adaptation and sustained detection performance under evolving traffic distributions. Drift detection is achieved using the wasserstein distance, which quantifies distributional shifts and triggers retraining when significant deviations are detected. The model incorporates advanced regularization techniques such as Maximum Mean Discrepancy (MMD), Elastic Weight Consolidation (EWC), Self-Information Regularization (SIR), and Mean Squared Error (MSE) to preserve previously acquired knowledge while adapting to evolving attack behaviors.

Extensive evaluations on a Free5GC-based 5G testbed demonstrate the robustness and adaptability of the proposed methodologies. The models consistently achieve high detection accuracy and maintain stable performance in dynamic environments characterized by contaminated and drifted data. By integrating PUL and continual adaptation into a unified anomaly detection pipeline, this work contributes to the development of resilient, intelligent, and data-efficient defense mechanisms for securing NSs in next-generation mobile networks.

Acknowledgments

First and foremost, I would like to express my deepest gratitude to God for granting me the strength, perseverance, and courage to continue this journey and never give up, even in moments of doubt and challenge.

I owe my heartfelt thanks to my wife and daughters for their unconditional love, patience, and constant support throughout this demanding endeavor. Their encouragement and understanding have been my greatest source of motivation and inspiration.

I would also like to extend my sincere appreciation to my mother, brother, and sister for their continuous support and confidence in me. Their faith in my abilities has been a foundation upon which I have built my perseverance and determination.

I am especially grateful to Ing. Raúl Bermúdez, who believed in me and opened the first door that ultimately allowed me to pursue and complete these studies, perhaps without even realizing the impact of his trust and support. His confidence in me marked the beginning of this journey.

Special appreciation goes to my Ph.D. supervisor, Dr. Chadi Assi, and to Dr. Hyame Assem Alameddine from the Ericsson Research team, whose guidance, encouragement, and expertise were invaluable throughout my research. I am deeply grateful for their mentorship and for the opportunity to work under their supervision.

I would also like to express my sincere gratitude to the members of the Ericsson team — Dr. Makan Pourzandi, Dr. Amine Boukhtouta, Dr. Luis Suárez, and Dr. Boubakr Nour — for their invaluable support, insightful discussions, and constructive feedback, which significantly enriched the quality and impact of this work.

Finally, I thank all those who, in one way or another, contributed to the completion of this thesis.

Contents

List of Figures	x
List of Tables	xii
List of Acronyms	xiii
1 Introduction	1
1.1 Motivation and problem statement	1
1.2 Contributions	3
1.2.1 Inter-slice Defender: An Anomaly Detection Solution for Distributed Slice Mobility Attacks	4
1.2.2 PUL-Inter-slice Defender: An Anomaly Detection Solution for Distributed Slice Mobility Attacks	5
1.2.3 IS^2 -DriftGuard: Concept Drift-Based Adaptive Detection of Distributed Slice Mobility Attacks	6
2 Background and Literature Review	8
2.1 Background	8
2.1.1 Network Slices Overview	9
2.1.2 Life Cycle of Network Slices and DDoS Attacks	11
2.1.3 Inter-Slice Communication and DDoS Attacks	13
2.1.4 Intra-Slice Communication and DDoS Attacks	14
2.2 Literature Review	15

2.2.1	Distributed Slice Mobility (DSM) Attack	15
2.2.2	Traditional DDoS Attacks Targeting Network Slices	15
2.2.3	Positive-Unlabeled (PUL) Learning in DDoS Attack Detection	16
2.2.4	Concept Drift in 5G Environment	17
3	Inter-slice Defender: An Anomaly Detection Solution for Distributed Slice Mobility Attacks	19
3.1	Introduction	19
3.2	DSM Attack - Threat Model and Variations	20
3.2.1	Assumptions	20
3.2.2	DSM Attack Variations - Threat Model	21
3.2.3	Illustrating Example	22
3.3	Inter-slice defender: Network slice anomaly detection solution	23
3.3.1	Data Collection and Pre-processing Module	23
3.3.2	Feature Extraction Module	25
3.3.3	Anomaly Detection Module	25
3.4	Environmental Setup And Data Simulation	27
3.4.1	5G Testbed	27
3.4.2	Simulation of Normal and Attack Network Traffic	27
3.4.3	Datasets for Anomaly Detection Model	29
3.5	DSM Attack Impact on 5G Control Plane	30
3.5.1	Impact on AMF	30
3.5.2	Overall Impact on 5G CP NFs	31
3.6	Experimental Results	32
3.6.1	LSTM-Autoencoder Architecture Selection	32
3.6.2	Reconstruction Error Threshold Selection	33
3.6.3	Inter-slice Defender Performance	34
4	PUL-Inter-slice Defender: An Anomaly Detection Solution for Distributed Slice Mobility Attacks	38

4.1	Introduction	38
4.2	Inter-Slice Defender	39
4.3	PUL-Inter-Slice Defender	40
4.3.1	PUL Assumptions	41
4.3.2	Data Collection and Pre-processing Module and Feature Extraction Module	42
4.3.3	PUL-Anomaly Detection Module	42
4.4	Datasets Generation	45
4.4.1	Datasets for Anomaly Detection Model	45
4.5	Experimental Results	47
4.5.1	LSTM-Autoencoder Architecture Selection	48
4.5.2	K-means architecture	49
4.5.3	Benchmark Models	50
4.5.4	PUL-Inter-Slice Defender Performance	51
4.6	Implementation Considerations in Real-World 5G Environment	53
4.6.1	Federated Learning for PUL-Inter-Slice Defender	53
4.6.2	Concept Drift Considerations	54
5	<i>IS²-DriftGuard: Concept Drift-Based Adaptive Detection of Distributed Slice Mobility Attacks</i>	55
5.1	Introduction	55
5.2	PUL-Inter-Slice Defender and Concept Drift	56
5.2.1	PUL-Inter-Slice Defender: Overview and Shortcomings	56
5.2.2	Concept Drift Adaptation in PUL: The Challenges	57
5.3	<i>IS²-DriftGuard</i>	59
5.3.1	Distributed Slice Mobility Drift Generator	59
5.3.2	Concept Drift Adapter	63
5.4	Experimental Setup	69
5.4.1	Main Dataset Description	69
5.4.2	Datasets Preparation	70

5.4.3	Environmental Setup	72
5.5	Experimental Results	72
5.5.1	CWGAN-GP Architecture	72
5.5.2	LSTM-Autoencoder - Architecture and Continuous Learning	74
5.5.3	MiniBatch K-means and Continuous Learning	75
5.5.4	Generation of Drifted RSA and TSA Distributions	76
5.5.5	Batch Size Selection	78
5.5.6	Wasserstein Threshold Selection for Drift Detection	80
5.5.7	Decay Factor Selection	81
5.5.8	IS^2 -DriftGuard Performance	82
5.5.9	Computational Overhead	87
5.6	Discussion	87
6	Real-World Deployment Considerations and Limitations	89
6.1	Deployment Aspects in Realistic 5G Environments	89
6.2	Experimental Setup and Deployment Limitations	90
6.3	Impact of Feature Quality on Model Performance	90
6.4	Scalability Considerations	91
7	Conclusion and Future Directions	92
	Bibliography	96

List of Figures

Figure 3.1	DSM attack and its variations (i.e., RSA, TSA).	20
Figure 3.2	Inter-slice defender architecture.	23
Figure 3.3	5G testbed.	28
Figure 3.4	AMF CPU consumption during benign and attack simulations.	31
Figure 3.5	CPU consumption ratio during RSA and TSA.	32
Figure 3.6	Threshold selection.	34
Figure 3.7	Inter-slice defender performance over different lookback values (i.e., timesteps).	35
Figure 3.8	Detailed Inter-slice defender performance with a timesteps = 1 and threshold $\alpha = 0.1408$.	36
Figure 3.9	Inter-slice defender performance in the presence of contaminated data.	37
Figure 4.1	Architecture of Inter-Slice Defender and PUL-Inter-Slice Defender. Inter-Slice Defender uses an LSTM-Autoencoder trained on benign data to detect attacks via reconstruction error and a threshold, while PUL-Inter-Slice Defender adopts PUL by leveraging the latent space for feature extraction to classify benign and DSM attack samples in the presence of contaminated training data through K-means.	39
Figure 4.2	2D PCA plot showing positive and negative classes in the training dataset with different levels of negative samples	47
Figure 4.3	F1-score comparison of PUL-Inter-Slice Defender, Inter-Slice Defender, PUL-OCSVM-RF, and PUL-OCSVM-XGBoost for detecting RSA and TSA under varying training data contamination levels (10%–40%).	51

Figure 5.1	PUL-Inter-Slice Defender.	56
Figure 5.2	IS^2 -DriftGuard’s detection and adaptation approach.	59
Figure 5.3	Drifted distributions containing RSA and TSA.	77
Figure 5.4	Average F1-score of adaptive PUL-Inter-Slice defender across different batch sizes.	79
Figure 5.5	Wasserstein Distance per Batch.	80
Figure 5.6	Sensitivity analysis of the decay factor.	82
Figure 5.7	Analysis of drifted distribution number 5.	85

List of Tables

Table 3.1	Features of Inter-slice defender model.	24
Table 3.2	Logical dependency between 5G procedures.	29
Table 3.3	Datasets statistics.	29
Table 3.4	LSTM-Autoencoder hyperparameters.	33
Table 4.1	Datasets statistics.	45
Table 4.2	LSTM-Autoencoder Training Parameters	49
Table 4.3	K-means Clustering Parameters	49
Table 5.1	Datasets statistics (N/A = Not Applicable).	70
Table 5.2	CWGAN-GP hyperparameters	74
Table 5.3	LSTM-Autoencoder hyperparameters	75
Table 5.4	MiniBatch K-Means hyperparameters	76
Table 5.5	Comparison of PUL-Inter-Slice Defender Ayala Molina, Assem Alameddine, Pourzandi, and Assi (2025) and IS^2 -DriftGuard under incremental RSA and TSA drift.	84
Table 7.1	Publications and Contributions during the Ph.D. Program	94
Table 7.2	Additional Contributions	95

List of Acronyms

Acronym	Full Term
3GPP	Third Generation Partnership Project
5G	Fifth Generation
ADWIN	Adaptive Windowing
AMF	Access and Mobility Management Function
API	Application Programming Interface
B5G	Beyond 5G
CL	Continual Learning
CGAN	Conditional Generative Adversarial Network
CP	Control Plane
CWGAN-GP	Conditional Wasserstein Generative Adversarial Network with Gradient Penalty
DDoS	Distributed Denial of Service
DL	Deep Learning
DDM	Drift Detection Method
DSM	Distributed Slice Mobility
eMBB	Enhanced Mobile Broadband
EWC	Elastic Weight Consolidation
FL	Federated Learning
GAN	Generative Adversarial Network
IoT	Internet of Things

ISS	Inter-Slice Switching
KPI	Key Performance Indicator
LC	Lifecycle
LSTM	Long Short-Term Memory
MMD	Maximum Mean Discrepancy
ML	Machine Learning
MNO	Mobile Network Operator
MSE	Mean Squared Error
mMTC	Massive Machine Type Communication
NF	Network Function
NFV	Network Function Virtualization
NEF	Network Exposure Function
NSSF	Network Slice Selection Function
NS	Network Slice
NSC	Network Slice Component
NWDAF	Network Data Analytics Function
OCSVM	One-Class Support Vector Machine
O-RAN	Open Radio Access Network
PCA	Principal Component Analysis
PDU	Packet Data Unit
PM	Performance Measurement
PUL	Positive-Unlabeled Learning
QoS	Quality of Service
RAN	Radio Access Network
RF	Random Forest
RSA	Random Slice Attack
SCAR	Selected Completely At Random
SDN	Software-Defined Networking

SIR	Self-Information Regularization
SMF	Session Management Function
TCP	Transmission Control Protocol
TSA	Target Slice Attack
UDP	User Datagram Protocol
UE	User Equipment
UERANSIM	User Equipment and Radio Access Network (RAN) simulator
UPF	User Plane Function
uRLLC	Ultra-Reliable Low Latency Communication
VM	Virtual Machine
VRNN	Variational Recurrent Neural Network
WGAN-GP	Wasserstein Generative Adversarial Network with Gradient Penalty

Chapter 1

Introduction

1.1 Motivation and problem statement

Network slicing stands as a pivotal technology and a key enabler, empowering Fifth Generation (5G) networks to provide intelligent services that necessitate a diverse range of requirements such as high data rates, ultra low latency, varying levels of security, dense connectivity, among others [Chahbar, Diaz, Dandoush, Cérin, and Ghoumid \(2020\)](#). Its significance is underscored by its essential role in the establishment of virtual end-to-end networks known as Network Slices (NSs), each dedicated to specific vertical industry services [Abbas, Khan, Afaq, and Song \(2021\)](#); [Chahbar et al. \(2020\)](#); [Zhang \(2019\)](#).

The adaptability afforded by network slicing allows a User Equipment (UE) to seamlessly switch between NSs to meet its specific needs, such as Quality of Service (QoS), security level, service cost, and other factors [Sajjad, Bernardos, Jayalath, and Tian \(2022\)](#). Furthermore, contingent upon the policies set by Mobile Network Operators (MNOs), such as NS load management, UEs can be seamlessly transferred from one NS to another. In line with the 3rd Generation Partnership Project (3GPP) standards, a UE can be subscribed to up to sixteen NSs [3GPP TS 23.501 \(2023\)](#) and can connect concurrently to a maximum of eight NSs [Sajjad et al. \(2022\)](#).

The inherent flexibility of NSs introduces a new security vulnerability susceptible to exploitation by attackers [De Alwis, Porambage, Dev, Gadekallu, and Liyanage \(2023\)](#); [Olimid and Nencioni](#)

(2020). In fact, whenever a UE transitions from one NS to another, also known as Inter-Slice Switching (ISS) [Sajjad et al. \(2022\)](#), it must undergo a re-initiation of its authentication and registration procedures, among others [3GPP TS 23.502 \(2023\)](#). These procedures generate a flood of signals within the Control Plane (CP) and between it and the UE. As such, attackers can trigger a high number of ISS events to generate an influx of signaling messages which can overwhelm Network Functions (NFs) such as those involved in the ISS procedures (e.g., Access Mobility Management Function (AMF), Session Management Function (SMF), etc.). Consequently, such attack, known as Distributed Slice Mobility (DSM) attack, can cause a Distributed Denial of Service (DDoS) which can disrupt the connection of legitimate UEs [Sathi and Murthy \(2020\)](#).

DSM attack along with its economical and performance damage was first discussed in [Sathi and Murthy \(2020\)](#). The same authors then proposed a protocol to secure the network against DSM attacks [Sathi and Murthy \(2021\)](#). The proposed protocol automatically selects an NS for a UE based on its subscription and services offered by external networks. Subsequently, [Bisht, Patra, and Kumar \(2023\)](#) developed a DSM attack detection solution based on evaluating the average waiting time and the average switching rate. To the best of our knowledge, [Bisht et al. \(2023\)](#) is the only work which addressed DSM attack detection while other such as [Khan, Farzaneh, Shahriar, Saha, and Boutaba \(2022\)](#); [Kuadey, Maale, Kwantwi, Sun, and Liu \(2021\)](#); [Sattar and Matrawy \(2019\)](#); [Thantharate, Paropkari, Walunj, Beard, and Kankariya \(2020\)](#) focused on DDoS attack detection in NSs based on flow-level features using machine learning (ML) techniques without evaluating the effectiveness of their solutions in detecting DSM attacks.

Despite these initial efforts, research on DSM attack detection remains limited and has yet to address real-world data challenges, none of the aforementioned works examined the robustness of their Machine Learning (ML) models under contaminated training conditions, raising concerns about their effectiveness in real-world deployments. In practice, training data often contain noise, mislabeled instances, or samples from overlapping distributions, all of which can undermine model accuracy and reliability [Papič, Kononenko, and Bosnić \(2023\)](#). Additionally, these studies commonly assume that the data distribution used during model development remains stable over time. Although this simplifies the development and evaluation of their solution, it fails to reflect the operational realities of 5G networks, which are inherently dynamic. It further overlooks probabilities

of concept drift [Hinder, Vaquet, and Hammer \(2024\)](#), where the statistical properties of data evolve, as well as adversarial concept drift [Sethi and Kantardzic \(2018\)](#) occurrences, in which attackers deliberately manipulate the statistical properties of their attacks to evade detection. By gradually mimicking benign traffic or modifying ISS patterns, adversaries can craft DSM attack variants that static ML models fail to recognize. As a result, models trained under the assumption of a fixed distribution [Bisht et al. \(2023\)](#); [Khan et al. \(2022\)](#); [Kuadey et al. \(2021\)](#); [Sathi and Murthy \(2021\)](#); [Sattar and Matrawy \(2019\)](#); [Thantharate et al. \(2020\)](#) often lack the required robustness to generalize in the face of evolving threats.

These limitations highlight the pressing need for DSM attack detection solutions that can maintain reliability under realistic network conditions and continuously evolving data distributions. In light of the growing complexity and dynamism of 5G environments, developing resilient detection mechanisms that adapt to distributional changes has become imperative. Static approaches based on idealized assumptions can no longer ensure robustness against adversarial manipulation or traffic evolution. Consequently, this thesis is motivated by the need to design a robust and adaptive framework capable of detecting DSM attacks in realistic 5G environments. By integrating learning models capable of handling contaminated datasets and adapting to drifted data distributions, this research aims to strengthen the robustness and reliability of DSM attack detection, contributing to the development of secure and resilient network slicing systems.

1.2 Contributions

This dissertation advances the state of the art in securing 5G network slicing by addressing key cybersecurity challenges associated with DSM attacks. The focus is on detecting, analyzing, and adapting to DSM-induced anomalies that compromise the stability and performance of the 5G core. Through a progressive research framework, the study introduces three major contributions: the development of Inter-Slice Defender for detecting DSM attacks using deep learning in a real 5G environment; the enhancement of this model through PUL-Inter-Slice Defender to ensure robust detection under contaminated data; and the design of IS^2 -DriftGuard, an adaptive solution capable of learning from concept drift in evolving attack environments. An overview of these contributions

is provided below, with each detailed in its respective subsection

1.2.1 Inter-slice Defender: An Anomaly Detection Solution for Distributed Slice Mobility Attacks

In Chapter 3, we present Inter-Slice Defender, a foundational anomaly detection framework that lays the groundwork for the subsequent contributions of this research. This work focuses on the design and implementation of a deep learning-based solution capable of detecting DSM attacks in realistic 5G environments. It details the architecture, data sources, and experimental setup employed to build and validate the model, emphasizing how Inter-Slice Defender captures temporal dependencies across sequential 5G procedures to profile normal slice mobility behavior and detect deviations caused by DSM attack variations. To achieve this, we make the following specific contributions in this chapter.

- We develop Inter-slice defender, a novel network slicing anomaly detection solution leveraging the Long Short-Term Memory (LSTM)-Autoencoder ML model to detect variations of the DSM attack. LSTM-Autoencoder combines the advantages of both; LSTM in capturing long-term dependencies in a sequence of data, and Autoencoder in learning the most relevant features. This makes LSTM-Autoencoder a good candidate to profile ISS events that involve multiple intertwined and sequential 5G procedures such as UE authentication, registration, Packet Data Unit (PDU) session establishment and deregistration.
- We train and test our Inter-slice defender model using time-based 3GPP Key Performance Indicators (KPIs) and Performance Measurements (PM) counters to profile CP NFs and NSs behavior. To the best of our knowledge, we are the first to explore the efficiency of these features in detecting network slicing attacks.
- Using the open source Free5GC testbed [Free5GC \(2023\)](#) and the UERANSIM [Aligungr \(2023\)](#) simulator, we build a 5G network with four different NSs and we adapt it to simulate different variations of the DSM attack.
- We simulate normal and DSM attack network traffic on our 5G testbed and collect the related

data. We leverage the collected data to calculate 3GPP KPIs and PM counters that are used by our Inter-slice defender solution to detect DSM attack. To the best of our knowledge, the generated dataset is the first of its kind for network slicing anomaly detection.

- We design two variations of the DSM attack, namely, Random Slice Attack (RSA) and Target Slice Attack (TSA) and evaluate their impact in causing a DDoS on the 5G network as a result of overloading the AMF.
- We perform an experimental analysis of our novel Inter-slice defender solution and show its robustness in detecting RSA and TSA with an average F1-score of 98.75%.

1.2.2 PUL-Inter-slice Defender: An Anomaly Detection Solution for Distributed Slice Mobility Attacks

In Chapter 4, we introduce PUL-Inter-Slice Defender, an enhanced anomaly detection framework that builds upon Inter-Slice Defender to address one of the most common challenges in real-world deployments, namely contaminated training data. This chapter focuses on extending the original deep learning model with a Positive Unlabeled Learning (PUL) paradigm, enabling reliable detection of DSM attacks when the training dataset contains a mixture of benign and attack samples. It presents the methodological integration of LSTM-Autoencoder-based feature extraction with K-Means clustering to differentiate between normal and anomalous behavior in partially labeled data. The design, experimental setup, and benchmarking results demonstrate the robustness and generalization capability of the model under realistic contamination levels. To achieve this, we make the following specific contributions in this chapter.

- We develop PUL-Inter-Slice Defender, a novel network slicing anomaly detection solution that incorporates PUL to detect DSM attack variations in contaminated training datasets comprising a blend of normal behavior samples and attack instances (e.g., RSA and TSA). Our approach employs an LSTM-Autoencoder to extract latent features profiling ISS events across key 5G procedures, including UE authentication, registration, PDU session establishment, and deregistration. These features are then clustered using K-Means to differentiate between benign and attack patterns.

- We train and test the PUL-Inter-Slice Defender using a dataset containing both benign and attack samples within a 5G network slicing environment. To the best of our knowledge, this is the first study to assess the effectiveness of PUL for detecting slicing-related attacks under such conditions.
- We developed two PUL-based solutions: one combining One-Class Support Vector Machine (OCSVM) with Random Forest (RF), hereafter called PUL-OCSVM-RF, and another combining OCSVM with XGBoost, subsequently termed PUL-OCSVM-XGBoost. We use these as benchmarks, along with the Inter-Slice Defender, to compare against PUL-Inter-Slice Defender.
- Our experimental results demonstrate PUL-Inter-Slice Defender robustness in detecting RSA and TSA, with an average F1-score exceeding 98% across training datasets containing 10% to 40% attack samples. This performance substantially surpasses that of the benchmark models, including Inter-Slice Defender, PUL-OCSVM-RF, and PUL-OCSVM-XGBoost, none of which exceed an F1-score of 89.87% under the same conditions.

1.2.3 IS^2 -DriftGuard: Concept Drift-Based Adaptive Detection of Distributed Slice Mobility Attacks

In Chapter 5, we present IS^2 -DriftGuard, an adaptive anomaly detection framework designed to extend the capabilities of PUL-Inter-Slice Defender by addressing the impact of evolving DSM attack patterns in dynamic 5G environments. This work introduces a concept drift-aware detection mechanism capable of maintaining consistent performance as traffic distributions change over time. It integrates Continual Learning (CL) principles into the anomaly detection process, combining a drift generator based on Conditional Wasserstein Generative Adversarial Network - Gradient Penalty (CWGAN-GP) and an adaptive learning module using LSTM-Autoencoder and MiniBatch K-Means to preserve prior knowledge while adapting to new attack behaviors. The chapter also details the design, training strategy, and experimental evaluation demonstrating the model's ability to sustain high detection accuracy under incremental drift conditions. To achieve this, we make the following specific contributions in this chapter.

- We propose IS^2 -DriftGuard, a CL-based anomaly detection and drift adaptation framework designed to handle incremental adversarial concept drift in DSM attacks within 5G networks under the PUL paradigm. It integrates an LSTM-Autoencoder for latent feature extraction and MiniBatch K-Means for clustering. To the best of our knowledge, this is the first work that tackles concept drift in the PUL setting, where the joint behavior of feature learning and clustering poses unique challenges to the model stability and its performance.
- We formulate a multi-objective loss function to enhance the model’s sensitivity to incremental drift in DSM attacks and support CL under the PUL paradigm. This loss combines: 1) Maximum Mean Discrepancy (MMD) to align feature distributions over time and preserve clustering consistency; 2) Elastic Weight Consolidation (EWC) to prevent catastrophic forgetting by protecting critical parameters; 2) Self-Information Regularization (SIR) to encourage diverse and non-correlated latent features for better representation; and 4) Mean Squared Error (MSE) to minimize reconstruction error, hence, ensuring retention of essential input information. These elements enable the model to adapt to evolving DSM attack patterns while maintaining robust performance over time.
- We develop a CWGAN-GP to simulate evolving DSM attack patterns by generating drifted distributions. This synthetic data introduces adversarial concept drift, enabling realistic and controlled training scenarios that support the evaluation and adaptation of our detection framework under incremental drift conditions. The combination of a conditional Generative Adversarial Network (GAN) with Wasserstein loss and gradient penalty for generating drifted attack distributions is the first of its kind in the literature.
- Our experimental results demonstrate the strong resilience and adaptability of IS^2 -DriftGuard to incremental concept drift induced by evolving RSA and TSA attack patterns. IS^2 -DriftGuard consistently achieves F1-scores above 90% when retraining is triggered through CL, outperforming its baseline, PUL-Inter-Slice Defender [Ayala Molina et al. \(2025\)](#).

Chapter 2

Background and Literature Review

2.1 Background

The 5G of mobile networks marks a major evolution in telecommunications, introducing unprecedented levels of speed, reliability, and connectivity [Altowaijri and Ayari \(2025\)](#); [Chaouki, Iqdour, and Boulouird \(2024\)](#). Unlike previous generations, 5G is designed not only to enhance data throughput but also to enable a highly flexible, programmable, and service-oriented network environment. This transformation is made possible through several key enablers, such as Edge Computing, NFV, SDN, and Network Slicing, which collectively allow dynamic resource allocation and service customization across heterogeneous use cases [Chaouki et al. \(2024\)](#); [Dias, Pinto, Santos, and Malta \(2025\)](#); [Venkatapathy et al. \(2024\)](#). Among these, network slicing stands out as one of the most transformative components of 5G architecture, providing the foundation for building isolated, virtualized networks tailored to specific application requirements, recognizing its pivotal role in delivering differentiated quality of service and ensuring effective resource management [Ali and Siyad \(2024\)](#); [Venkatapathy et al. \(2024\)](#) .

However, while these technological advancements greatly enhance flexibility and performance, they also introduce a broader attack surface and new security challenges [Dias et al. \(2025\)](#). The dynamic, software-defined, and virtualized nature of 5G networks makes them inherently more complex and therefore more vulnerable to cyber threats. Among these, DDoS attacks represent one of the most critical and disruptive threats [Bisht et al. \(2023\)](#); [Dias et al. \(2025\)](#); [Farzaneh, Shahriar,](#)

[Al Muktadir, Towhid, and Khosravani \(2024\)](#). By overwhelming network functions or specific slices with a massive volume of malicious traffic, DDoS attacks can degrade service quality, exhaust computational resources, and even cause service outages that impact multiple verticals simultaneously. The problem is exacerbated in 5G environments where network slicing enables multi-tenant infrastructures; a successful attack targeting one slice can potentially propagate to shared control-plane elements, compromising the performance and reliability of other slices [Bisht et al. \(2023\)](#); [Dias et al. \(2025\)](#). Consequently, securing 5G against DDoS attacks requires continuous monitoring, intelligent traffic analysis, and adaptive defense mechanisms capable of coping with the highly dynamic and distributed nature of modern network architectures [Dias et al. \(2025\)](#); [Farzaneh et al. \(2024\)](#).

2.1.1 Network Slices Overview

5G network slicing is an advanced networking technique that enables the formation of several virtual networks, or NSs, on a single physical infrastructure [De Alwis, Porambage, Dev, Gadekallu, and Liyanage \(2024\)](#); [Singh, Singh, Hegde, and Gupta \(2024\)](#). Each NS can be tailored and assigned resources based on its particular requirements, thereby maximizing network resource efficiency and effectiveness [De Alwis et al. \(2024\)](#). This method allows the development of unique, isolated virtual networks optimized for specific services and applications. These applications leverage key 5G capabilities, such as eMBB, uRLLC, and mMTC, to achieve exceptional performance and functionality [Salahdine, Han, and Zhang \(2022\)](#). As a result, network slicing not only increases service customization and flexibility but also plays a fundamental role in unlocking the transformative potential of 5G across a wide range of applications.

- eMBB is dedicated to delivering ultra-fast data speeds and handling media-rich content over wireless networks, setting a new benchmark in capacity compared to previous generations. This key 5G capability supports a range of high-demand applications, including high-definition video streaming, virtual reality, augmented reality, and other bandwidth-intensive services. In professional environments, eMBB enhances productivity by enabling rapid access to large files, improving cloud service efficiency, and supporting high-quality video conferencing experiences, thus transforming enterprise connectivity and user experience [De Alwis et al.](#)

(2024); [Singh et al. \(2024\)](#)).

- uRLLC is engineered to meet the demands of applications that require exceptionally reliable, near-instantaneous communication. This 5G feature is indispensable for mission-critical tasks such as remote surgery, autonomous driving, and industrial automation, where even a momentary delay can have serious repercussions. By enabling precise control in environments like manufacturing, where machinery must operate with exacting accuracy, and facilitating real-time data exchange between autonomous vehicles, uRLLC supports the highest standards of safety and operational efficiency in advanced technological applications [De Alwis et al. \(2024\)](#); [Singh et al. \(2024\)](#)).
- mMTC is built to manage vast, low-power communications across a high volume of devices and sensors with minimal human intervention. As a core component of the Internet of Things, mMTC supports an exceptionally high density of connected devices per square kilometer, making it indispensable for applications like smart cities, automated agriculture, and extensive environmental monitoring, where thousands of devices collect and transmit data simultaneously. This capability enables efficient resource management by ensuring that even with a high concentration of devices, the network effectively manages data transmission, optimizing energy consumption and bandwidth usage [Wijethilaka and Liyanage \(2021\)](#)).

The diverse applications and advanced functionalities enabled by 5G network slicing (e.g., such as eMBB, uRLLC, and mMTC) have significantly expanded network capabilities and service quality [Dias et al. \(2025\)](#); [S. Gao, Lin, Fu, Li, and Cao \(2024\)](#). However, this flexibility that allows the customization and isolation of services also increases the risk of DDoS attacks [Cybersecurity and Infrastructure Security Agency \(CISA\) \(2024\)](#); [Min-Gyu and Kim \(2025\)](#). As network slicing becomes a fundamental element of 5G architecture, adversaries can exploit some vulnerabilities to launch sophisticated DDoS campaigns that disrupt individual slices or shared network functions [Dias et al. \(2025\)](#); [S. Gao et al. \(2024\)](#). For example, attackers may overwhelm the CP by flooding signaling procedures or exhaust the resources of a targeted NS to indirectly affect others through shared infrastructure [Min-Gyu and Kim \(2025\)](#). Such attacks undermine NS isolation and

degrade the overall performance and reliability of the 5G coreCybersecurity and Infrastructure Security Agency (CISA) (2024). Therefore, while the adaptability and specificity of NSs deliver clear benefits in efficiency and scalability, they also introduce critical security concerns that must be addressed through proactive DDoS detection and mitigation strategies Dias et al. (2025); S. Gao et al. (2024). Security concerns within network slicing can be systematically divided into three primary areas De Alwis et al. (2023); S. Gao et al. (2024): LC Security, which addresses the stages of NS creation and management; Inter-Slice Security, which focuses on safeguarding interactions between different network slices; and Intra-Slice Security, which ensures protection within a single network slice. These three categories will be explored in the following section.

2.1.2 Life Cycle of Network Slices and DDoS Attacks

The NS LC is a structured framework that governs the planning, deployment, management, and eventual retirement of NSs in a 5G environment De Alwis et al. (2024). This LC consists of four key phases: preparation, commissioning, run time, and decommissioning. This framework optimizes resource use, enhances service delivery, and ensures security and reliability. By following it, operators can dynamically configure NSs to support diverse applications, such as high speed data and mission critical services, maintaining peak performance through upgrades and decommissioning. This approach improves operational efficiency, agility, cost effectiveness, and service quality for end users. The LC of NSs is composed of the following four phases De Alwis et al. (2024); S. Gao et al. (2024).

- **Preparation:** This phase involves setting up the network environment and creating NS templates, which define the components, structure, and configuration of a NS. The actual NS doesn't exist yet; it's only defined in template form.
- **Commissioning:** In this phase, resources and NFs are created, configured, and activated based on the template to establish the slice.
- **Run Time:** During operation, NSs can be modified (e.g., configuration changes, updates, association or dissociation of resources). Monitoring and reporting occur in this phase to ensure functionality and performance.

- Decommissioning: This final phase involves deactivating and dismantling the NS, freeing up resources and NFs for reuse.

While the structured NS LC enhances operational efficiency and service quality, each phase also introduces potential risks and vulnerabilities that could be exploited [S. Gao et al. \(2024\)](#); [Singh et al. \(2024\)](#). From configuration weaknesses during the preparation and commissioning stages to security gaps in resource management during run time and decommissioning, these vulnerabilities require careful oversight to prevent unauthorized access, data breaches, and service disruptions.

DDoS Security Issues in the Life Cycle of Network Slices

In the context of 5G network slices, DDoS attacks pose a significant threat due to the shared infrastructure and resources among NSs [De Alwis et al. \(2024\)](#); [S. Gao et al. \(2024\)](#); [Singh et al. \(2024\)](#). Network slicing enables multiple virtual networks on a shared physical network, which can lead to vulnerabilities if isolation is inadequate. DDoS attacks can target specific NSs or critical NFs, like the SMF or AMF, leading to resource exhaustion and service disruption.

- Preparation Stage: During initial resource allocation and configuration, improper setups can be leveraged by attackers to flood key resources, such as network bandwidth and processing capacity within NFs like the AMF. Additionally, Network Element (e.g. routers, switches, NFs) registration processes are susceptible; attackers may exploit unprotected configurations or weak access controls to overload registration procedures, potentially leading to denial of service conditions [S. Gao et al. \(2024\)](#).
- Commissioning Stage: When NSs are being created and initialized, APIs and open interfaces used for this purpose are at risk. For instance, DDoS attacks can overwhelm the system by generating repeated requests to create NSs. Additionally, attackers may exploit vulnerabilities in template management, which defines NS characteristics, to introduce configurations that further increase susceptibility to DDoS attacks, draining resources during NS activation [De Alwis et al. \(2024\)](#).
- Run Time Stage: In active NSs, DDoS attacks can affect multi-slice scenarios by exploiting shared resources or communication links, enabling cross-slice attacks that disrupt multiple

NSs simultaneously. Additionally, monitoring and control interfaces (e.g., NEF, network slice management function, etc.) that manage NSs can be overwhelmed by false data or excessive management requests, leading to resource exhaustion. Attackers may also exploit UE registration processes, targeting NFs involved in this process to disrupt registration and authentication, potentially degrading the performance of affected NSs. The NEF and other open APIs are particularly vulnerable, as they expose 5G capabilities to external entities, creating entry points for DDoS attacks that can degrade operational status and quality of service [Dangi et al. \(2022\)](#); [Singh et al. \(2024\)](#).

- **Decommissioning Stage:** In the final phase, vulnerabilities in Resource release and cleanup processes can be exploited if residual connections or incomplete releases allow attackers to send bogus decommissioning requests or capitalize on leftover data, preventing proper cleanup and leading to resource drain [De Alwis et al. \(2024\)](#).

Overall, the NS LC provides a structured framework for efficient deployment and management, but each stage introduces distinct vulnerabilities that adversaries can exploit to launch DDoS attacks. Understanding these weaknesses across the preparation, commissioning, run-time, and decommissioning phases is crucial for developing resilient defense mechanisms.

2.1.3 Inter-Slice Communication and DDoS Attacks

Inter-slice communication, enabling interconnected operations across multiple NSs, improves overall communication efficiency and effectiveness. However, the diversity in network resources, configurations, and requirements introduces significant security challenges [De Alwis et al. \(2024\)](#). These challenges extend across multiple components, including 5G user devices, interactions between services, communications within and between NSs, network management frameworks, and foundational resource infrastructures.

In the context of inter-slice security, DDoS attacks pose a particular threat [De Alwis et al. \(2024\)](#). DDoS attacks may exploit inter-slice communication channels as a pathway to spread disruptions across different NSs. This vulnerability is especially concerning in multi-slice scenarios, where shared resources or communication links between NSs mean that an attack on one NS can impact

others, either through resource exhaustion or by leveraging shared infrastructure elements. Malicious actors can potentially exploit these inter-slice communication links to target specific functions within a NS, indirectly affecting related NFs in other slices connected through shared resources or dependencies. These risks emphasize the need for robust isolation and monitoring mechanisms to prevent cross-slice disruptions caused by targeted DDoS attacks.

2.1.4 Intra-Slice Communication and DDoS Attacks

Intra-slice communication differs from inter-slice communication by using dedicated resources instead of shared ones, tailoring each solution, component, and function to the specific needs of customer applications. Despite the use of dedicated resources, intra-slice communication faces some of the same security challenges as inter-slice communication due to similar control mechanisms, including technical elements and strategies employed to manage, direct, and secure network operations within and between NSs [De Alwis et al. \(2024\)](#); [Singh et al. \(2024\)](#). Additionally, intra-slice communication has unique security concerns related to elements like 5G user devices, slice service interfaces, subslices, NS management, and security tied to NFs.

In the context of intra-slice security, DDoS attacks remain a significant concern. While each NS is designed with dedicated resources, attackers may still target specific NFs or resources within an individual NS, such as UE registration or slice management functions. By overwhelming these components, a DDoS attack can degrade the NS's performance, affecting the quality of service and potentially disrupting critical applications tied to that NS, which underscore the importance of robust intra-slice protections, including stringent access controls and advanced monitoring to detect and mitigate DDoS threats within individual NSs.

The preceding sections established that DDoS attacks represent one of the most severe threats to 5G network slicing, capable of degrading performance and availability across both inter-slice and intra-slice domains. Within this context, a specific type of DDoS attack, known as the Distributed Slice Mobility (DSM) attack, has emerged, exploiting the mobility and flexibility of UE across NSs to induce signaling floods and disrupt network stability [Sathi and Murthy \(2020\)](#). To better situate this research within existing work, the following section presents a comprehensive review of related studies addressing DSM attacks, traditional DDoS detection in the context of NSs, and the

application of PUL and concept drift adaptation techniques for DSM attack detection. .

2.2 Literature Review

2.2.1 Distributed Slice Mobility (DSM) Attack

A limited number of research studies have been dedicated to analyzing and detecting DSM attacks. For instance, [Sathi and Murthy \(2020\)](#) were the first to propose the DSM attack and theoretically study the economic and performance damage it can cause on the network. They note that such damage exceeds those resulting from other DDoS attacks. [Sajjad et al. \(2022\)](#) detailed 3GPP-based procedures enabling UE mobility and its transition between NSs. They highlighted critical UE mobility challenges and suggested potential avenues for future research. [Bisht et al. \(2023\)](#) illustrated the consequences of ISS events that result in a DDoS attack. They employed an algorithm based on two metrics, average waiting time and switching rate, to identify and block compromised UEs. From a different perspective, [Sathi and Murthy \(2021\)](#) introduced a preventive approach through a novel NS selection protocol. The protocol suggests that the network chooses the best NS for the UE based on its subscription and data network services which contradicts 3GPP 5G standards that allow the UE to request the services of a specific NS [3GPP TS 23.501 \(2023\)](#).

The work tackling DSM attack is limited to theoretical studies and does not consider the practical implementation of this attack in a real 5G network. Further, DSM attack detection solutions are limited to [Bisht et al. \(2023\)](#), which followed a simplistic algorithmic approach based on two metrics that were not tested on a real 5G network slicing-based dataset, thus overlooking the complexity of this attack. In addition, none of the existing studies have evaluated their proposed solutions under conditions involving contaminated data or drifting DSM attack distributions. As a result, their effectiveness in real-world scenarios, where the statistical properties of attacks may evolve over time, remains uncertain.

2.2.2 Traditional DDoS Attacks Targeting Network Slices

DDoS attacks are known to be the most disruptive attacks in the realm of cybersecurity [Al-Shareeda, Manickam, and Ali \(2023\)](#). While DSM has been identified as a DDoS attack on NSs,

DDoS flooding attacks, such as User Data Protocol (UDP) lag and Transmission Control Protocol (TCP) SYN, among others, have also gained attention. DDoS flooding attacks can be caused by exploiting vulnerabilities in the high number of UEs in 5G networks and targeting its NSs. To detect these attacks, DL and mathematical models have been proposed. For example, [Khan et al. \(2022\)](#) used a bidirectional-LSTM model to detect DDoS attacks (i.e., UDP flooding and TCP SYN attacks) in two NSs. They evaluated the impact of these attacks by measuring the incurred bandwidth and latency. Similarly, the authors in [Kuadey et al. \(2021\)](#) devised a framework leveraging LSTM to identify DDoS attacks and detect if UEs' NS requests are either legitimate or indicate an attack.

Further, [Thantharate et al. \(2020\)](#) developed Secure5G that employs a convolutional neural network for early DDoS attack detection by analyzing network traffic patterns collected from a 5G testbed. Secure5G detects anomalous UEs' requests targeting multiple NSs and redirects suspicious UEs to a quarantine NS. [Sattar and Matrawy \(2019\)](#) proposed an NS isolation mechanism as a mitigation measure against DDoS attacks using a mathematical model. They evaluated their model's performance by measuring its impact on bandwidth, response time, and round-trip time.

Most of the aforementioned work on NS attack detection focuses on general DDoS attacks that are not NS specific. Despite the efforts presented in some of these works to test these attacks on a 5G testbed and use the generated dataset for DDoS attack detection, their anomaly detection models remain limited to flow-based features that do not capture any NS-specific characteristics. Moreover, these studies do not examine nor evaluate the performance of their detection solutions in detecting the DSM attack and its variations.

2.2.3 Positive-Unlabeled (PUL) Learning in DDoS Attack Detection

Although PUL has been widely studied in various domains, its potential in 5G technology remains underexplored, particularly for anomaly detection. Conversely, the concept of learning from positive and unlabeled data has gained significant attention in the machine learning community for anomaly detection applications in other technologies, due to its practical relevance in deployment scenarios where obtaining fully clean or comprehensively labeled datasets is infeasible.

For example, [Long and Zhang \(2024\)](#) proposed PUNet, a PUL-based semi-supervised anomaly detection model that addresses distribution shift and data imbalance using Variational Recurrent

Neural Network-based features and reconstruction-loss-guided pseudo-labeling. It uses CatBoost for classification and effectively detects DDoS and encrypted traffic anomalies. [Dilworth and Gudla \(2024\)](#) applies PUL to cloud-based DDoS detection using the BCCC-cPacket-Cloud-DDoS-2024 dataset. It frames DDoS attacks as positive samples and evaluates multiple classifiers. The study highlights PUL’s effectiveness in handling limited labeled data for detecting DDoS attacks in cloud environments. [Fan, Yao, Du, and Du \(2023\)](#) This work proposes SRPU, a PUL-based solution for malicious traffic detection under class imbalance. It combines self-paced learning with a reweighting strategy to gradually select confident samples and manage noisy data. SRPU effectively detects various attack types, including DDoS, even with limited labeled samples. [Lv et al. \(2020\)](#) introduced a method that uses non-negative risk estimator learning for intrusion detection in network systems. This approach treats cyber-attacks as positive samples in a PUL scenario and utilizes a risk estimator to calculate binary classification loss, specifically enhanced with focal loss to address data imbalance issues [Ayala Molina et al. \(2025\)](#).

Given the demonstrated success of PUL in various anomaly detection applications, we adopt this approach in our methodology for detecting DSM attacks in 5G networks [Ayala Molina et al. \(2025\)](#), as obtaining clean, fully labeled datasets is often impractical.

2.2.4 Concept Drift in 5G Environment

Several approaches have explored the integration of concept drift management mechanisms within DDoS detection frameworks in 5G networks. [Beshah, Abebe, and Melaku \(2024\)](#) proposed an adaptive online DDoS detection solution for Internet of Things (IoT) systems. Their approach leverages ensemble learning to address concept drift, incorporating two drift detection techniques: ADWIN and the DDM. ADWIN detects drift by maintaining a sliding window and applying statistical tests to identify significant distribution changes, while DDM raises alarms based on increases in the model’s error rate. To enhance adaptability, the authors introduced an accuracy update weighted probability averaging ensemble, which assigns weights to base learners based on their real-time performance. [Chu, Jin, Qiao, and Feng \(2024\)](#) proposed a concept drift-aware intrusion detection solution for IoT data streams that detects evolving attacks such as DDoS. They

introduced Ensemble Multiple Non-parametric Concept Localization Detectors (EMNCD), an ensemble concept drift localization method that combines nonparametric statistical tests, including Kolmogorov–Smirnov, Wilcoxon rank sum, and Mann–Kendall to identify distributional changes in the data stream. Anomaly detection is performed using Isolation Forest, while model adaptation relies on XGBoost.

It is also worth noting that concept drift in 5G has been investigated outside the scope of DDoS detection. For instance, [Manias, Chouman, and Shami \(2022\)](#) proposed a drift detection and adaptation approach for 5G core networks, targeting concept drift in user behavior patterns. Using a functional 5G core prototype, the authors simulate drift by introducing different UE behavior profiles, where each profile involves unique traffic characteristics such as request frequency and image size. An LSTM model is trained to predict packet lengths, and drift is detected through performance degradation. Adaptation is achieved via incremental retraining using either persistent or non-persistent memory.

[Gudepu et al. \(2024\)](#) proposed a GAN-based methodology for drift and anomaly detection in O-RAN, aiming to preserve ML model reliability under changing traffic patterns. The system combines GAN-generated synthetic traffic with a Kolmogorov–Smirnov test to identify deviations from LSTM-predicted behavior. Validated using real-time data from the Colosseum testbed, their method enables early detection of distributional shifts. Building on this work, the same authors later introduced GEN-DRIFT [Gudepu et al. \(2025\)](#), a Generative AI-based architecture designed to detect and adapt to concept drift in B5G networks. By leveraging Variational Autoencoders and GANs, the framework generates synthetic data and applies the Kolmogorov–Smirnov test to detect distributional shifts in user traffic. Integrated into the O-RAN architecture, the approach enables efficient and timely model adaptation through incremental retraining.

While existing studies present promising methods for handling concept drift in dynamic network environments, they neither address DSM attacks nor consider adversarial concept drift in the context of PUL. As a result, their applicability to detecting DSM attacks in realistic and evolving 5G deployments is limited. This emphasizes the need for dedicated approaches that can address deliberate distribution shifts introduced by adversaries to evade detection.

Chapter 3

Inter-slice Defender: An Anomaly Detection Solution for Distributed Slice Mobility Attacks

3.1 Introduction

This chapter investigates the security implications of DSM attacks within 5G network slicing and presents Inter-Slice Defender, a robust anomaly detection framework based on an LSTM-Autoencoder model designed to detect these threats. It begins with a detailed examination of DSM attack mechanisms, emphasizing their impact on the CP through excessive ISS events that can overwhelm critical NFs. The chapter then introduces the architecture of Inter-Slice Defender, which comprises data-collection, feature-extraction, and anomaly-detection modules leveraging 3GPP KPIs and PM counters to model slice-mobility behavior. The LSTM-Autoencoder captures temporal dependencies across sequential 5G procedures to profile normal slice mobility and detect deviations induced by DSM attacks. The proposed framework is validated on a functional 5G testbed through the simulation of both benign traffic and two crafted DSM attack variations (e.g., RSA and TSA). The experimental evaluation demonstrates that Inter-Slice Defender achieves an average F1-score of 98.75%, confirming its effectiveness and robustness in detecting DSM attacks

under realistic 5G operating conditions.

3.2 DSM Attack - Threat Model and Variations

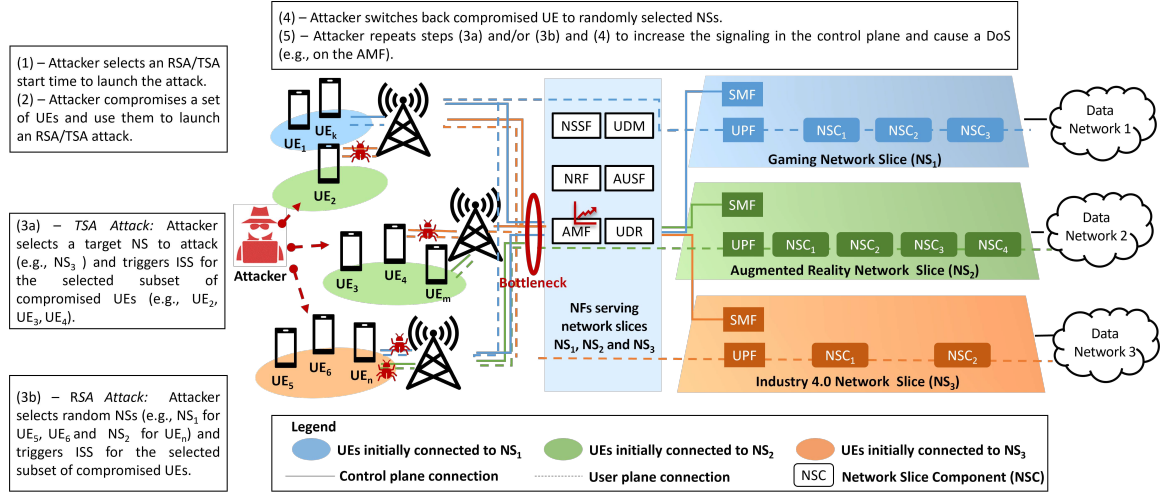


Figure 3.1: DSM attack and its variations (i.e., RSA, TSA).

The DSM attack is a DDoS attack that exploits UEs ISS events to disrupt the performance of both the 5G network CP and its NSs. Such disruption is caused by the signaling associated with the high number of malicious ISS events. In fact, ISS triggers the PDU session release procedure to release the UE PDU session in its current NS, registration, and PDU session establishment to a new desired NS, and other procedures, such as UE authentication to the network [3GPP TS 23.502 \(2023\)](#); [Sajjad et al. \(2022\)](#). These procedures involve a sheer volume of signaling messages within the 5G CP and the NSs. In the following, we detail the DSM attack assumptions, and threat model while highlighting two of its variations (i.e., RSA and TSA).

3.2.1 Assumptions

To perform RSA and TSA, the following assumptions are made:

- (1) **Access to a set of compromised UEs.** UEs and IoT devices are known to be highly vulnerable to attacks such as those noted in [Neshenko, Bou-Harb, Crichigno, Kaddoum, and Ghani \(2019\)](#). An attacker, can thus compromise a set of UEs and use them as a botnet to perform a DSM attack.

- (2) **UEs NS configurations and credential information.** We assume that the attacker has access to compromised UEs credentials (i.e., cryptographic keys, NSs configuration, etc.) and can use them to successfully connect to the MNO network and its NSs.
- (3) **Remote activation of ISS.** We assume that the attacker is able to remotely access and manipulate the compromised UEs in order to force them to trigger ISS to switch between their accessible NSs.

3.2.2 DSM Attack Variations - Threat Model

In this work, we exploit two variations of the DSM attack, illustrated in Fig. 3.1:

Random Slice Attack (RSA). To perform a RSA, the attacker selects different subsets of compromised UEs, and simultaneously switch them to randomly selected NSs. RSA creates varying loads on these random NSs as a result of the varying number of connected UEs.

Target Slice Attack (TSA). To perform a TSA, the attacker selects a target NS to attack and triggers ISS events to switch the compromised set of UEs to the selected NS.

RSA and TSA can be performed as followed:

- (1) **Select RSA/TSA start time.** The attacker identifies the network's peak times, during which a substantial number of UEs are connected to the network in order to perform the RSA/TSA. This will impose a more pronounced impact on the CP's NFs by augmenting the peak network load with a high frequency of ISS events that will be initiated by the attack.
- (2) **Compromise UEs.** The attacker compromises a set of UEs and use them as botnet to launch the RSA/TSA.
- (3) **Launch RSA/TSA** To launch a RSA, the attacker identifies multiple random NSs (i.e., one per each compromised UE) and switches the compromised UEs to these NSs by triggering ISS for each of them. In contrast, to launch a TSA, the attacker switches the compromised UEs to a target, pre-selected NS.
- (4) **Switch back compromised UEs to random NSs.** The attacker switches back some or all the compromised UEs to random NSs in order to introduce random ISS patterns for UEs and

make the attack look stealthy.

- (5) **Repeat steps (3) and (4).** For increased and varying impact on the network and its NSs, the attacker can repeat steps (3) and (4) using the same or different subset of compromised UEs at different times, making the attack harder to detect while always causing disruptions to the network.

3.2.3 Illustrating Example

To better explain RSA and TSA, we represent in Fig. 3.1, a telecommunication network with multiple base stations and connected UEs. Note that the limited number of UEs in the figure is only for illustrative purposes and a larger number is assumed in real networks. The network is configured with three different NSs, each providing services for a specific vertical industry. For simplicity and without loss of generality, we account for a gaming NS (i.e., NS_1), an augmented reality NS (i.e., NS_2) and an industry 4.0 NS (i.e., NS_3). Each of these NSs has its own SMF, User Plane Function (UPF) and a set of Network Slice Components (NSCs). An NSC is a capability (i.e., abstraction of an NF) that fulfills the functional requirements of an end-to-end NS [Sathi, Srinivasan, Thiruvassagam, and Chebiyyam \(2018\)](#). These NSs are served by a common set of CP NFs represented in the blue box in Fig. 3.1. Particularly, the sharing of AMF by-design (i.e., based on 3GPP) between NSs is intrinsic to allow the UEs to connect simultaneously many of them [3GPP TS 23.501 \(2023\)](#).

We consider UEs initially connected to a specific NS identified by the circle they belong to (i.e., $\{UE_1, UE_k\}$ connected to NS_1 , $\{UE_2, UE_3, UE_4, UE_m\}$ connected to NS_2 , and $\{UE_5, UE_6, UE_n\}$ connected to NS_3). We assume that the attacker compromises some of these UEs and uses them to perform RSA and TSA. More precisely, UE_2 , UE_3 , and UE_4 are compromised and used to perform a TSA towards NS_3 (i.e., the targeted NS). UE_5 , UE_6 and UE_n are used to perform an RSA. The random NS selected for ISS of UE_5 , UE_6 is NS_1 while NS_2 is selected for UE_n . The user and CP connections shown in Fig. 3.1 reflect the UEs connections after the attacks. RSA and TSA lead to a DDoS on the network, particularly impacting the AMF. The AMF becomes a bottleneck as it is the first point of contact between the UEs and the network and is involved in the ISS related procedures.

3.3 Inter-slice defender: Network slice anomaly detection solution

We present Inter-slice defender (Fig. 3.2), our novel NS anomaly detection solution that leverages LSTM-Autoencoder, 3GPP KPIs [3GPP TS 28.554 \(2023\)](#) and PM counters [3GPP TS 28.552 \(2023\)](#) (Table 3.1) to detect DSM attack and its variations. The Inter-slice defender is composed of three modules detailed below.

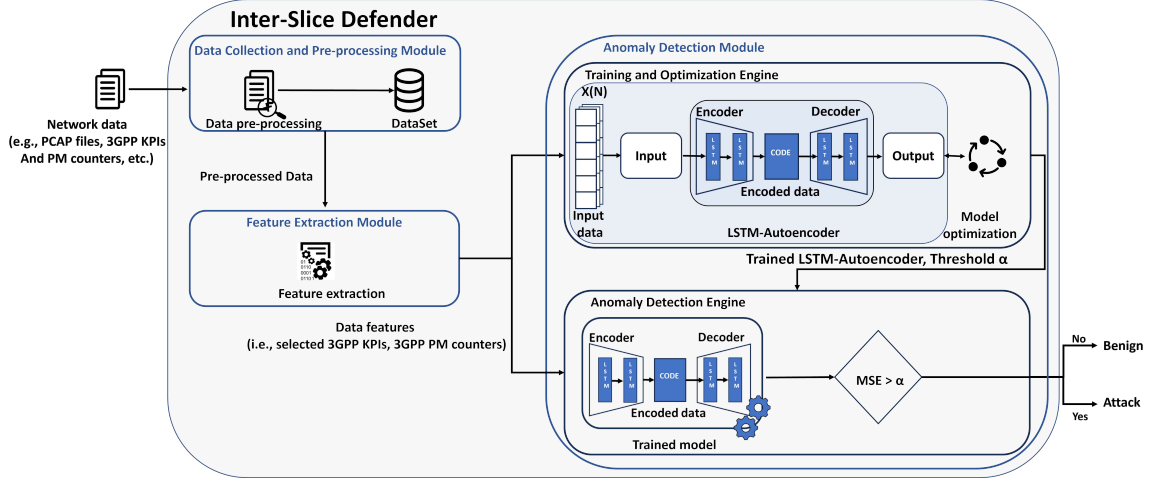


Figure 3.2: Inter-slice defender architecture.

3.3.1 Data Collection and Pre-processing Module

The data collection and pre-processing module (Fig. 3.2) collects data (e.g., PCAP files, 3GPP KPI, PM counters, etc.) from the 5G network and pre-processes it to be ready for use by the feature extraction module. The collected data can include PCAP files depicting signaling messages between the 5G network components (i.e., RAN and core NFs) and which pertain to the different 5G procedures such as ISS, registration, etc. It can also account for 3GPP KPIs and PM counters (Table 3.1) that are usually calculated at the different 5G NFs, and can be available at the NWDAF [Garcia-Martin, Gramaglia, and Serrano \(2023\)](#). NWDAF is a 5G NF that facilitates data collection and analysis. It can collect KPIs from the 5G NFs and calculate others such as those KPIs related to NSs [Garcia-Martin et al. \(2023\)](#).

The data collected by the data collection and pre-processing module then undergoes a transformation that simplifies the extraction of features. For instance, in this work, we use TShark

TSHARK.DEV (2023) to collect PCAP files from the network. We merge these files into a single one while maintaining their chronological order. This will serve the accurate calculation of time series based features by the feature extraction module. Then, we pre-process the unified PCAP file by extracting the most relevant information (e.g., IP source, IP destination, port destination, HTTP/2 header, etc.) for feature extraction. We save them in a CSV file that is then shared with the feature extraction module.

Table 3.1: Features of Inter-slice defender model.

Type	3GPP KPI features	Definition
3GPP-NS	*Registration success rate of one single NS	Success ratio of registration procedures (i.e., ratio of number of successful registrations over total number of attempted registrations) within a single NS for a specific AMF set.
	*PDU session establishment success rate of one NS	Rate of successful PDU session establishment request over total number of attempted requests across all SMFs associated with a specific NS.
	*Mean number of PDU sessions of network and NS	Average number of successful PDU session within a specific NS.
	*Maximum number of PDU sessions of NS	Maximum number of successfully established PDU sessions within a single NS.
3GPP PM Counter features		Definition
3GPP-AMF	Number of initial registration requests	Total number of initial registration requests that AMF receives.
	Number of successful initial registrations	Count of successful initial registrations processed by the AMF.
	Total number of attempted service requests	Number of attempted service requests including those initiated by the network and those initiated by UEs.
	Total number of successful service requests	Cumulative count of successful service requests accounting for those initiated by both the network and by UEs.
3GPP-SMF	+Number of PDU session creation requests	Number of PDU session creation requests received by the SMF.
	+Number of successful PDU session creations	Number of PDU sessions successfully established by the SMF.
	+Number of failed PDU session creations	Count of PDU sessions successfully created by the SMF.
	*Max time of PDU session establishment	Maximum time for PDU session establishment in each granularity period divided into sub-counters for each NS.
	Number of released PDU sessions (AMF initiated)	Number of PDU sessions released at SMF with initiation originating from the AMF.
3GPP-NSSF	Number of NS selection requests	Total number of NS selection requests that the NSSF receives.
	Number of successful NS selections	Total successful NS selections executed by the NSSF.
	Number of failed NS selections	Number NS selection attempts that failed at the NSSF.
Non-3GPP PM Counter feature		Definition
AMF	Number of failed initial registrations	Number of unsuccessful registrations.

* Feature computed per each NS; + Feature computed per each SMF

3.3.2 Feature Extraction Module

The feature engineering module acquires the pre-processed data from the data collection and pre-processing module and uses it for feature extraction. This data is used for feature extraction, selection, and normalization to extract 3GPP KPI [3GPP TS 28.554 \(2023\)](#) and PM counters [3GPP TS 28.552 \(2023\)](#) such as those listed in Table 3.1. These features capture the 5G network and its NSs normal behavior, along with the abnormalities that can be caused by a DSM attack. The calculated features are time-series based, suitable for LSTM-Autoencoder required input format. To select the most pertinent features, a variance threshold process is employed which eliminates features characterized by minimal variations or perceived as noise.

Consequently, a total of 17 distinctive features (Table 3.1) are selected to comprehensively scrutinize, analyze, and establish the normal behavioral patterns exhibited by 5G networks. They can be classified per type that reflects their calculation. For instance, some features are calculated per NF (i.e., AMF, SMF, NSSF), mainly those NF involved in ISS related procedures (e.g., “Number of PDU session creation requests” feature is calculated at the SMF as it is involved in PDU session establishment procedure) while others are calculated per NS. With the exception of “Number of failed initial registrations” feature that is not standardized by 3GPP, the computation of the remaining features follows 3GPP specifications. These features can be collected from the NWDAF or the NFs if available or can be calculated from the PCAPs as in this work. Finally, it is worth noting that in the case where any of the aforementioned NFs is dedicated to an NS, its related features will then reflect the NS it serves.

3.3.3 Anomaly Detection Module

The anomaly detection module is composed of two engines, mainly the training and optimization engine and the anomaly detection engine, detailed hereafter.

Training and optimization engine. The training and optimization engine is dedicated to training the anomaly detection model based on LSTM-Autoencoder [Said Elsayed, Le-Khac, Dev, and Jurcut \(2020\)](#), optimizing its architecture, hyper-parameters, and selecting a threshold that will lead

to good detection performance. The choice of LSTM-Autoencoder lies in its capability of capturing temporal dependencies in the data through an LSTM and combining it with an Autoencoder. Autoencoder creates an effective representation of the sequential patterns in the data by encoding and decoding them. This yields very efficient for DSM attack detection where dependencies and precedence constraints of the different 5G procedures representing ISS are highly descriptive of the attack. For instance, an ISS event to a new NS cannot occur before de-registering from the current NS and registering to the new one (Section 3.4.2). Further, the frequency and attack patterns over time can be captured by LSTM through its cells capabilities that can remember values over different time intervals [Said Elsayed et al. \(2020\)](#). With Autoencoder coming into play, the most relevant features representing the data are learned.

An Autoencoder is an unsupervised model that learns to reconstruct the input data [Said Elsayed et al. \(2020\)](#). When used for anomaly detection, it is trained on mostly benign data. Hence, it will fail to reconstruct any anomalous input, which will result in a relatively big difference between the input and its reconstructed version. This difference is known as the reconstruction error and is used to depict anomalies by comparing against a threshold as we explain next. The training and optimization engine determines this threshold such that the F1-score is maximized (see Section 3.6.2).

Anomaly detection engine. The anomaly detection engine uses the trained LSTM-Autoencoder model provided by the training and optimization engine along with the selected threshold α to perform real-time anomaly detection (Fig. 3.2). Such detection is based on comparing the reconstruction error provided by the LSTM-Autoencoder against the selected threshold α . In this work, we use the MSE [Said Elsayed et al. \(2020\)](#) as the reconstruction error metric given its sensitivity to outliers, however, other metrics such as the root mean squared error and mean absolute error can also be used. The MSE represents the degree of alignment between the model predicted outcomes and the actual ground truth. Given that LSTM-Autoencoder is trained to learn benign behavior, it is expected to succeed in reconstructing benign data which will lead to a low reconstruction error (i.e., MSE). In contrast, a high reconstruction error is expected in case of an anomaly. Thus, if the $MSE \leq \alpha$, the data is classified as benign, while an $MSE > \alpha$ depicts malicious data.

3.4 Environmental Setup And Data Simulation

Given the lack of a 5G dataset suitable for training and testing our Inter-slice defender solution, we present, in this section, our environmental setup and data simulation strategy. We highlight our 5G testbed that we deploy to simulate normal network traffic in addition to RSA and TSA.

3.4.1 5G Testbed

We employ the open source free5GC-compose and free5GC all-in-one implementation [Free5GC \(2023\)](#), adhering to the 3GPP standard [3GPP \(2023\)](#) to build our testbed, depicted in Fig. 3.3. The testbed runs free5GC-compose version 3.3.0 as the CP on a VM operating Ubuntu 20.04 – Focal. The VM features 8 virtual CPUs, 8 GB of RAM, and a 60 GB of hard drive, with each NF encapsulated within this virtualized environment. To enhance the realism of our simulation environment, we opt to segregate the RAN from the CP. This segregation involves installing UERANSIN 3.2.6 [Ali-gungr \(2023\)](#), a UE and RAN simulator, on a distinct VM. Our devised testbed comprises four NSs, each featuring a dedicated UPF installed in a separate VM, and a dedicated SMF. For the installation and configuration of all UPFs, we use the free5GC all-in-one version 3.3.0 [Free5GC \(2023\)](#) while the SMFs are deployed on containers in the VM hosting the CP. Furthermore, our testbed hosts 92 UEs configured to be able to connect to the existing four different NSs. The creation and management of the VMs within our framework are orchestrated through OpenStack [OpenStack \(2023\)](#).

3.4.2 Simulation of Normal and Attack Network Traffic

To collect benign and attack data, we perform three different simulations. The initial simulation replicates a 5G network operating under normal conditions without any external attacks. In contrast, the subsequent two simulations were conducted to assess the impact of RSA and TSA.

Normal network traffic simulation. Throughout the 120-minute duration dedicated to this simulation, 92 UEs are used to simulate varying loads of normal network traffic. Each UE randomly triggers many 5G procedures from those detailed in Table 3.2. Notably, the 5G procedures listed under the “Possible subsequent procedures” column are executable solely and exclusively if the 5G procedure indicated under the “Triggered procedure” column has been successfully completed.

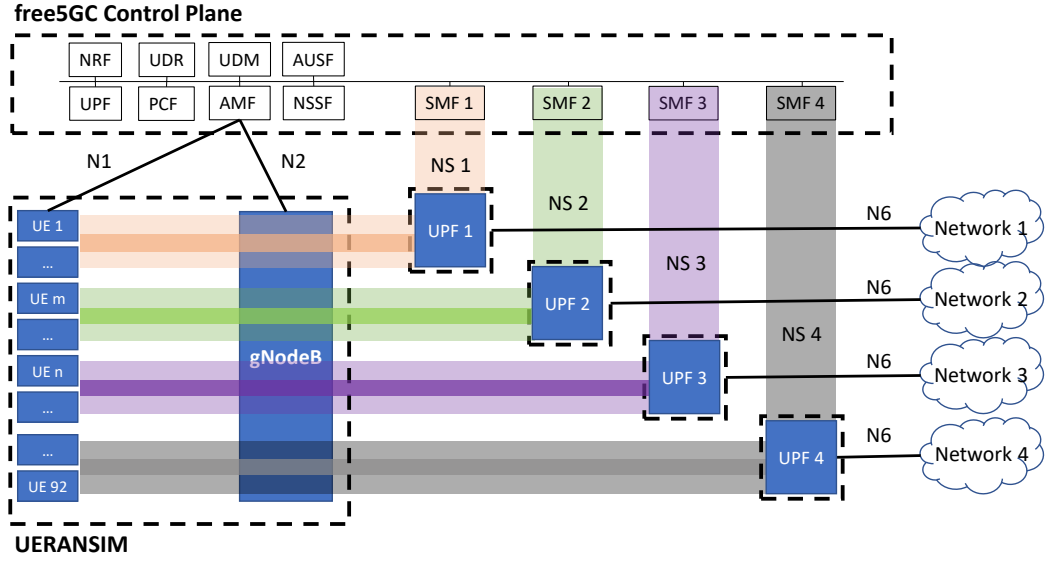


Figure 3.3: 5G testbed.

This logical interdependence underscores the sequential nature of these procedures. For instance, as outlined in Table 3.2, following the successful completion of a registration procedure, several events can be triggered: ISS, Uplink, Downlink, UE release PDU session, and gNodeB release PDU session. Finally, the benign activity of the UEs is recorded in PCAP files. The latter constitutes the benign dataset that will be used for our LSTM-Autoencoder model training and testing.

RSA and TSA Simulations. To conduct the RSA and TSA simulations, we adhere to the same approach as in the benign simulation. We perform the simulation of each attack using a total of 92 connected UEs out of which 28 are compromised and used for the attack. Following our threat model (Section 3.2), the attacker strategically decides on the time to launch the RSA or TSA such that it coincides with the network’s peak activity. As a result, during the attack simulations, the network operates normally for the first 60 minutes, after which the attack is initiated when the load on the network is designed to be at its peak. When the attacks start, the 28 compromised UEs are connected to any of the four NSs and are used to perform ISS events in the quest of overloading the CP.

Table 3.2: Logical dependency between 5G procedures.

Triggered procedure	Possible subsequent procedures
ISS	ISS, Uplink, Downlink, UE release PDU session, gNodeB release PDU session
Registration	ISS, Uplink, Downlink, UE release PDU session, gNodeB release PDU session
Uplink	ISS, Downlink, UE release PDU session, gNodeB release PDU session
Downlink	ISS, Uplink, UE release PDU session, gNodeB release PDU session
UE release PDU session	ISS, Downlink, Uplink, gNodeB release PDU session
gNodeB release PDU session	ISS, Uplink, Downlink

3.4.3 Datasets for Anomaly Detection Model

The data generated from the aforementioned simulations is used to create different datasets to facilitate the training and evaluation of our Inter-slice defender model. These datasets include benign and/or attack records (Table 4.1).

Table 3.3: Datasets statistics.

Dataset type	Total number of records	Benign records	Attack records	
			RSA	TSA
Training + validation	40000	40000	0	0
Optimization	10000	5000	2500	2500
RSA Test	20000	10000	10000	0
TSA Test	20000	10000	0	10000

Training and validation dataset

This dataset exclusively comprises benign data obtained from the normal network traffic simulation. It serves as the foundation for training and validating our model.

Optimization dataset

The optimization dataset includes both benign and malicious data extracted from the data generated from the RSA and TSA simulations. An equal number of benign and attack records were extracted from each of the attack simulations to generate this dataset. The optimization dataset

is used to determine the threshold that maximizes the F1-score for distinguishing between benign patterns and potential attacks, as we explain in Section 4.5.1.

Test datasets

To rigorously evaluate the performance of our anomaly detection model, three different test datasets incorporating both benign and malicious data are considered. A dataset per each DSM attack variant (i.e., RSA test dataset, TSA test dataset) is generated as a result of the RSA and TSA simulations respectively. This approach allows for a thorough assessment of the model’s ability to discern between normal and anomalous patterns.

It is crucial to highlight that these datasets are meticulously designed to be mutually exclusive, ensuring that there are no redundant records between them.

3.5 DSM Attack Impact on 5G Control Plane

To assess the impact of the RSA and TSA on the 5G network, we observe the CPU utilization of the different 5G CP NFs during RSA and TSA simulations and compare it with their CPU utilization during normal network traffic.

3.5.1 Impact on AMF

We first focus on the impact of the attacks on the AMF given that it is the CP NF that is involved the most in UE to 5G network communication and is the first CP point of contact. In fact, despite the role AMF plays in UE registration, authentication, and NS selection and allocation, it is usually shared among different NSs (as explained in Section 3.2.3). Thus, when a significant number of UEs simultaneously perform ISS, such as in the case of RSA and TSA, we observe a significant increase in the AMF CPU utilization which leads to a DDoS (Fig. 5.7).

As explained in Section 3.4.2, we simulate varying loads of normal network traffic and trigger the RSA and TSA at the peak load depicted at $time = 60\text{ minutes}$. Fig. 3.4a shows the AMF CPU utilization during normal network traffic simulation and depicts the highest utilization of 49.95% at $time = 60\text{ minutes}$. In contrast, Fig. 3.4b and 3.4c show that the AMF CPU consumption

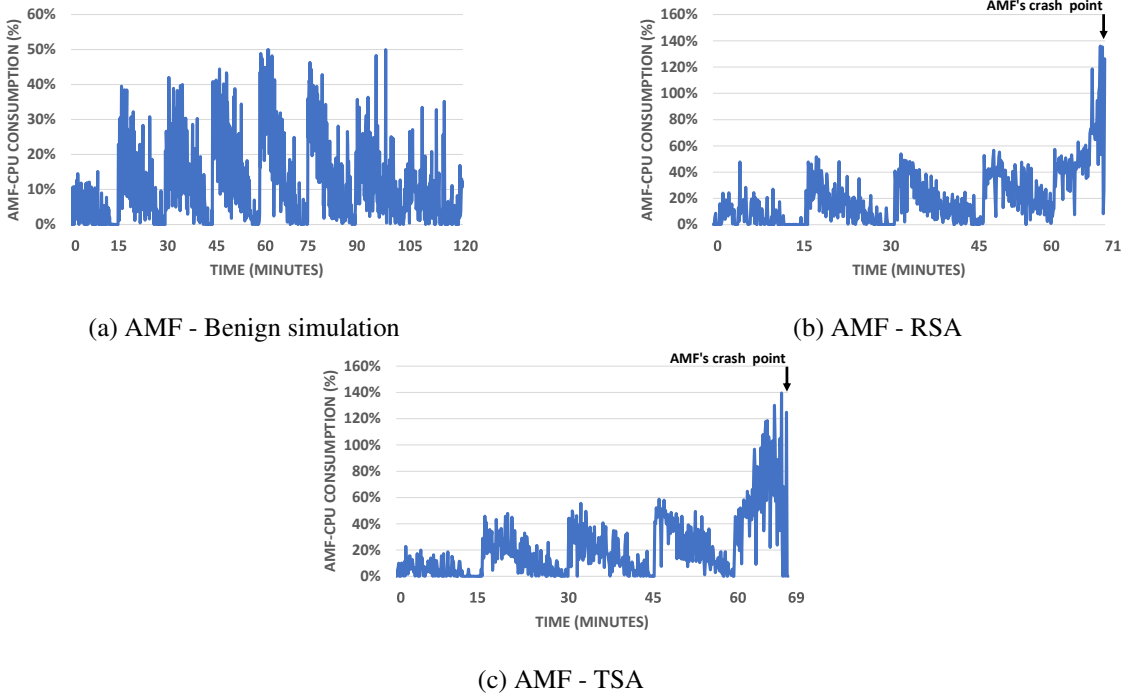


Figure 3.4: AMF CPU consumption during benign and attack simulations.

reaches 135.98% and 139.55% during the RSA and TSA respectively which are launched at the peak network load (i.e., *time* = 60 *minutes*). The figures show that RSA and TSA last for 11 *minutes* and 9 *minutes* respectively before the crash of the AMF, and hence the whole network. This shows the DDoS impact the DSM attack can have on the 5G network and further emphasizes the need to secure 5G networks against it. Finally, it is worth noting that the AMF CPU utilization could exceed 100% in our testbed because our CP NFs are containerized and can borrow from each other the CPU unused resources. Nonetheless, if the network does not allow sharing of unused CPU resources, the DDoS impact would have been observed earlier.

3.5.2 Overall Impact on 5G CP NFs

Despite the significant impact RSA and TSA have on the AMF, we observe a similar increase in the CPU consumption on other CP NFs as shown in Fig. 3.5. In fact, this figure shows the average CPU ratio for all the CP NFs during both RSA and TSA simulations. This ratio is calculated by dividing the average CPU consumption for each NF during the attack period (i.e., [60 *minutes* – 71 *minutes*] for RSA, [60 *minutes* – 69 *minutes*] for TSA) over that during the benign simulation

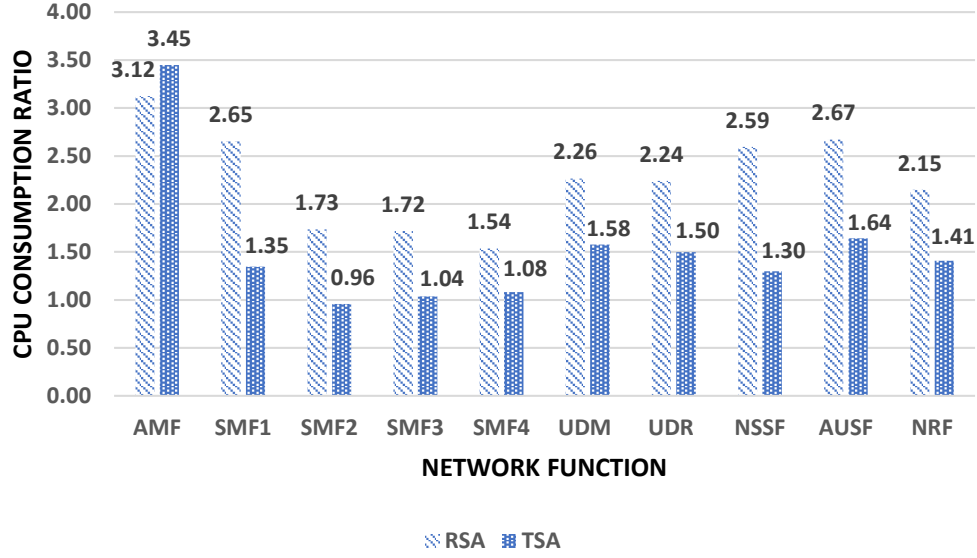


Figure 3.5: CPU consumption ratio during RSA and TSA.

for that same period.

Note that except for the AMF, the increase in the CPU utilization for all the NFs is greater during the RSA than during the TSA. Fig. 3.5 shows that RSA has a bigger impact on the 5G CP NFs. However, TSA has a higher impact on the AMF than the RSA, which results in degrading its performance in handling the requests, many of which ended up being dropped. Thus, fewer requests were forwarded to other CP NFs during TSA than during RSA which explains a smaller increase in their CPU utilization. Finally, unlike the AMF which CPU consumption exceeds 100% during the attacks, we observe that CPU consumption of other CP NFs remains under 50%.

3.6 Experimental Results

In this section, we evaluate Inter-slice defender and assess its effectiveness in detecting RSA and TSA while examining its performance against contaminated data.

3.6.1 LSTM-Autoencoder Architecture Selection

To determine the best architecture for Inter-slice defender LSTM-Autoencoder that is able to efficiently detect DSM attack and its variations, we train and validate the performance of multiple

architectures and examine their performance. Thus, we allocate 20% of the training dataset as a validation dataset, and we train the model using the remaining portion of the training dataset (Table 4.1). To select the model hyperparameters, we apply the K-fold cross-validation technique Wong and Yeh (2020) for both training and validation. K-fold cross-validation is an effective method for validating and refining DL models, as it helps prevent overfitting and offers a more thorough approach to evaluate model performance. Additionally, to further mitigate the risk of overfitting, our model incorporates L1 regularization and dropout techniques within the encoder.

We test and evaluate various LSTM-Autoencoder architectures and select $\{64, 32, 32, 32, 64\}$ as Inter-slice defender model architecture given that it provides the best detection performance. This architecture is composed of two LSTM layers of 64 and 32 neurons respectively, forming the encoder, and a decoder having the encoder’s mirrored architecture. The code of the LSTM-Autoencoder is 32 neurons. We train our model with the selected architecture after fine-tuning its hyperparameters (Table 3.4). Its training time is equal to 21.92 seconds.

Table 3.4: LSTM-Autoencoder hyperparameters.

Hyperparameter	Value
Epochs	20
Dropout	0.1
L1 regularization	0.15
Batch size	16
Loss Function	MSE
Learning rate	0.01
Optimizer	Adam
Hidden activation function	Relu

3.6.2 Reconstruction Error Threshold Selection

To detect anomalies, there is a need to compare the reconstruction error (i.e., MSE) provided by Inter-slice defender model against a selected threshold α (Section 4.3.3). In this work, we resort to the F1-score, that represents the harmonic mean between precision and recall, to determine α . We evaluate the performance of our model against different threshold values, as shown in Fig. 3.6, using our optimization dataset (Table 4.1). Fig. 3.6 depicts that with the increase of the threshold value, the precision increases and stabilizes at 100% while the recall decreases as attack data starts getting

misclassified as benign. After calculating and comparing the F1-score for the different threshold values presented in Fig. 3.6, we select $\alpha = 0.1408$ that maximizes the F1-score. We use this value in the remainder of the experiments to evaluate the model performance.

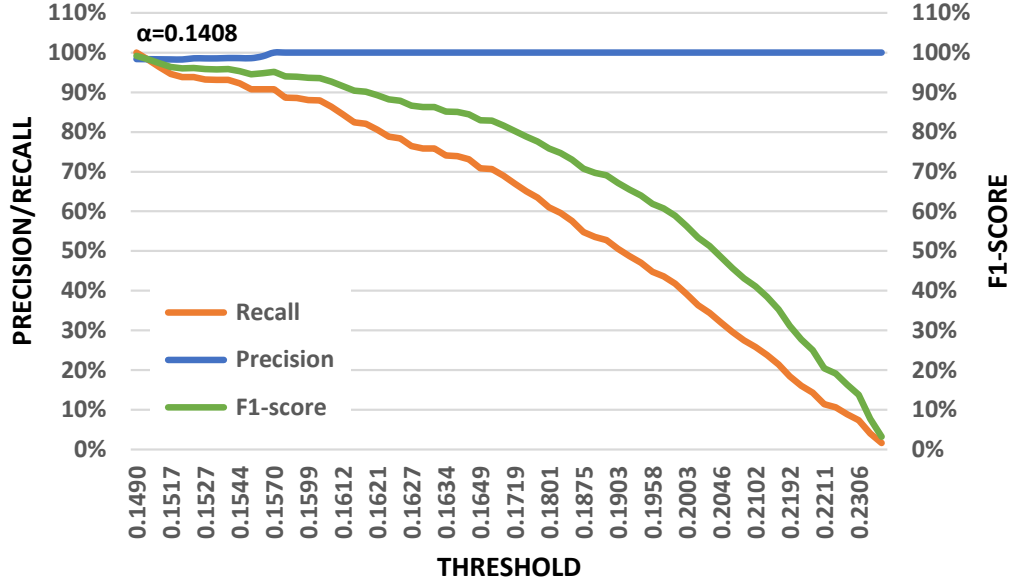


Figure 3.6: Threshold selection.

3.6.3 Inter-slice Defender Performance

Impact of Timesteps

Recall that LSTM is a recurrent neural network that is known for its ability to detect dependencies in a sequence of data. It uses a lookback hyperparameter that determines the number of previous timesteps within a sequence that will be used to predict the next timestep. Thus, we evaluate in Fig. 3.7, the performance of Inter-slice defender over multiple values of the lookback hyperparameter (i.e., number of timesteps) when tested on RSA and TSA test datasets (Table 4.1). Our results show that Inter-slice defender detection performance degrades for these test datasets when the number of timesteps increases. This is because UEs are highly dynamics and their long term communication patterns and resulting load on the network may not necessarily be indicative of their most recent behavior. Thus, the most updated KPIs and counters are better suited for predicting the next benign or attack behavior. Nonetheless, the model performance remains satisfactory with an F1-score above

87% even when $timesteps = 8$. In the remainder of the experiments, we evaluate the model performance with $timesteps = 1$ as it provides the best detection performance with an F1-score of 99.1% and 98.4% when tested with RSA and TSA datasets, respectively.

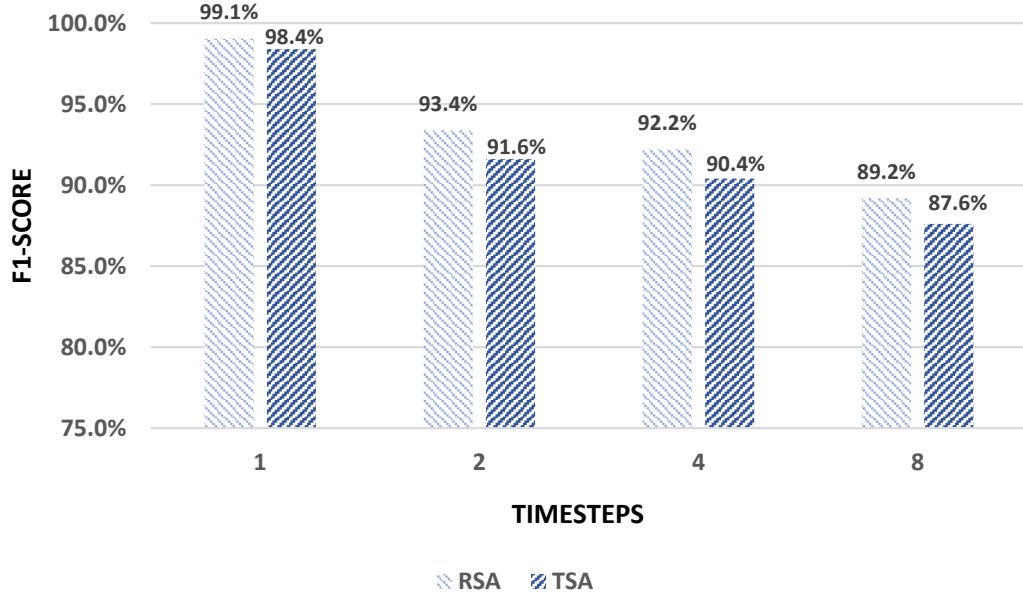


Figure 3.7: Inter-slice defender performance over different lookback values (i.e., timesteps).

Detailed performance with $timesteps = 1$

Given that Inter-slice defender model has shown the best performance with $timesteps = 1$ and a threshold $\alpha = 0.1408$, we take a closer look at this performance in Fig. 3.8. Fig. 3.8 depicts F1-scores of 99.1% and 98.4% recorded for RSA and TSA datasets, respectively, leading to an average F1-score of 98.75%. Further, our tests show that the model exhibits a perfect recall for RSA and TSA test datasets revealing that the model is able to correctly discern RSA and TSA. However, the degraded precision depicts that the model misclassifies benign data as anomalous.

Impact of contamination

Given that it is hard to obtain purely benign data in real-world deployment, we assess the robustness of Inter-slice defender model in the presence of contaminated data. Thus, we retrain our model with the training dataset (Section 4.4.1) after contaminating it with different percentages of

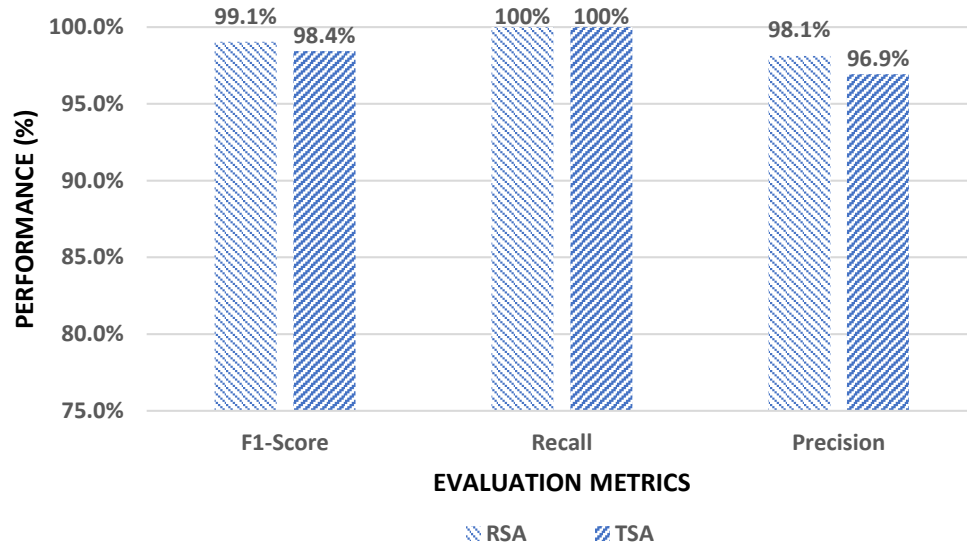


Figure 3.8: Detailed Inter-slice defender performance with a $timesteps = 1$ and threshold $\alpha = 0.1408$.

RSA and TSA records. Then, we test the model with the RSA and TSA test dataset (i.e., after excluding RSA and TSA records used for training) while fixing the $timesteps = 1$ and the threshold $\alpha = 0.1408$. Fig. 3.9 shows that the performance of our model significantly degrades, especially with 1% and 2% of contamination, with the increase of the contamination percentage. However, starting with 3% of contamination, only a slight degradation in the performance is observed while maintaining an F1-score above 84%. Nonetheless, this test shows that further fine-tuning of the model needs to be performed, upon the need, in real network deployment.

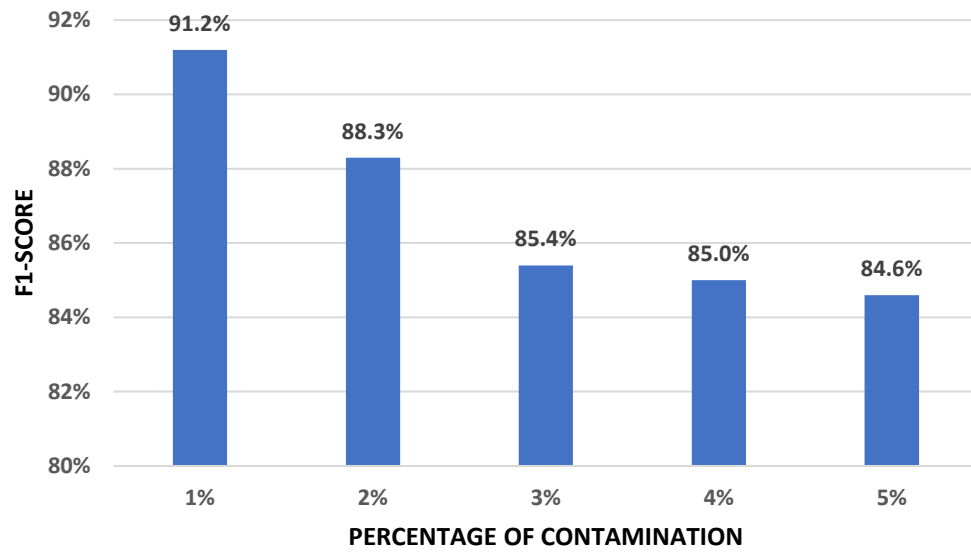


Figure 3.9: Inter-slice defender performance in the presence of contaminated data.

Chapter 4

PUL-Inter-slice Defender: An Anomaly Detection Solution for Distributed Slice Mobility Attacks

4.1 Introduction

This chapter presents PUL-Inter-Slice Defender, an enhanced anomaly detection framework built upon the foundation of Inter-Slice Defender. It addresses the limitations encountered when training the original model with contaminated datasets, a frequent challenge in real network deployments where benign and attack samples may coexist. To overcome these constraints, the proposed framework incorporates the PUL paradigm, which enables effective classification when only a subset of benign samples is labeled. The chapter details the theoretical assumptions underlying PUL, the architecture of the proposed model that combines an LSTM-Autoencoder with K-Means clustering, and the methodology for handling unlabeled data representing both normal and attack behaviors. The performance of PUL-Inter-Slice Defender is benchmarked against three models, the original Inter-Slice Defender and two custom PUL-based baselines, to demonstrate its robustness in detecting the devised DSM attack variations under contaminated data conditions. The experimental evaluation demonstrates that PUL-Inter-Slice Defender achieves an average F1 score exceeding

98%, confirming its effectiveness and reliability in detecting DSM attacks within realistic 5G network environments.

4.2 Inter-Slice Defender

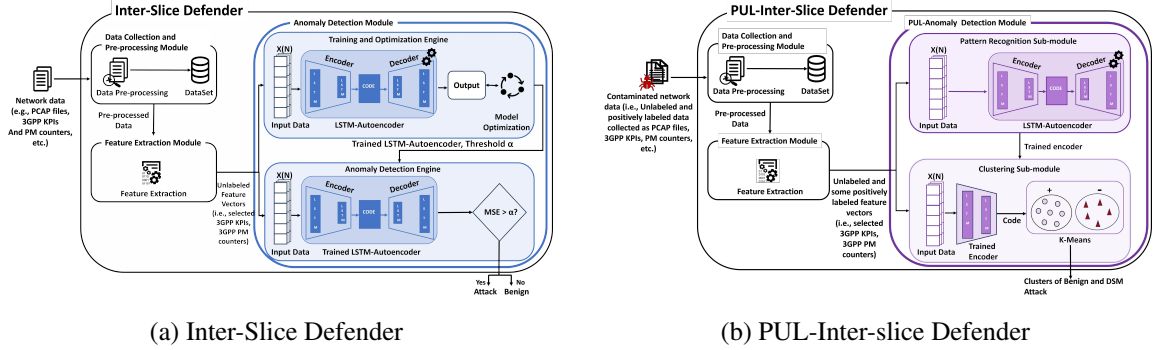


Figure 4.1: Architecture of Inter-Slice Defender and PUL-Inter-Slice Defender. Inter-Slice Defender uses an LSTM-Autoencoder trained on benign data to detect attacks via reconstruction error and a threshold, while PUL-Inter-Slice Defender adopts PUL by leveraging the latent space for feature extraction to classify benign and DSM attack samples in the presence of contaminated training data through K-means.

As this study extends the research conducted in our previous work [Molina, Wehbe, Alameddine, Pourzandi, and Assi \(2024\)](#), this section presents its proposed solution, named Inter-Slice Defender (Fig. 4.1a), for detecting DSM attacks, with emphasis on its key findings and limitations. Fig. 4.1a illustrates this approach, which comprises three modules: data collection and pre-processing, feature extraction, and anomaly detection.

The data collection and pre-processing module gathers 5G network data and prepares it for the feature extraction module, which extracts, selects, and normalizes 3GPP KPIs [3GPP TS 28.554 \(2023\)](#) and PM counters [3GPP TS 28.552 \(2023\)](#) that are shown in Table 3.1. The anomaly detection module is composed of two engines: the training and optimization engine, and the anomaly detection engine. The training and optimization engine is responsible for training the anomaly detection model based on an LSTM-Autoencoder [Said Elsayed et al. \(2020\)](#), optimizing its architecture and hyperparameters, and selecting a threshold that ensures good detection performance. The choice of LSTM-Autoencoder is based on LSTM’s suitability for processing and learning from sequential

data that capture long-term dependencies, while the Autoencoder is chosen for its ability to compress input data into a lower-dimensional latent space and then reconstruct it. Inter-Slice Defender leverages the Autoencoder’s ability as an unsupervised model to reconstruct input data [Said Elsayed et al. \(2020\)](#). Trained primarily on benign data, the model poorly reconstructs anomalous inputs, resulting in a high reconstruction error. This error, compared against a selected threshold α , is used by the anomaly detection engine to perform real-time anomaly detection.

Inter-Slice Defender (Fig. 4.1a) achieved notable results, reaching an average F1-score of 98.75%, demonstrating its robustness in detecting DSM attacks and their variations [Molina et al. \(2024\)](#). However, this performance was based on training exclusively with benign data under the assumption of a clean training environment, which is unrealistic in real-world deployments where operational data may include noise or stealthy attacks. To better reflect practical scenarios, we retrained Inter-Slice Defender using contaminated datasets composed of typical operational data mixed with defined percentages of attack samples. This resulted in a significant performance drop, with the F1-score declining to 84.6% when only 5% of the training data was contaminated [Molina et al. \(2024\)](#). The presence of such anomalous data introduces conflicting patterns that compromise the model’s ability to learn a reliable representation of normal behavior. This weakens the discriminative power of the reconstruction error, as the model begins to internalize features of both benign and attack inputs. Consequently, the error gap narrows, leading to false negatives and underscoring the need for further adjustments to ensure robust anomaly detection in real-world network environments.

4.3 PUL-Inter-Slice Defender

We present the PUL-Inter-Slice Defender (Fig. 4.1b), our novel NS anomaly detection solution that leverages LSTM-Autoencoder, K-Means, 3GPP KPIs [3GPP TS 28.554 \(2023\)](#), and PM counters [3GPP TS 28.552 \(2023\)](#) (Table 3.1) to detect DSM attacks and their variations using a training dataset composed of positive and negative samples. Our approach relies on several key assumptions inherent to PUL, which are fundamental in handling this type of dataset. PUL-Inter-Slice Defender

is composed of three modules: data collection and pre-processing module, feature extraction module, and PUL-anomaly detection module. We begin with a description of the PUL assumptions, followed by detailed explanations of each module.

4.3.1 PUL Assumptions

The PUL is a binary classification problem where only a subset of labeled positive and unlabeled samples are available. This approach addresses the practical limitations of acquiring fully labeled datasets due to factors such as time, expertise, and financial investment [Bekker and Davis \(2020\)](#); [Papič et al. \(2023\)](#). The aim of PUL is to construct a classifier that can effectively distinguish between positive and negative samples, even when the training data consists of only a subset of labeled positive examples and a large set of unlabeled examples, thereby alleviating the cost of fully labeling data. It is important to note that the labeling process in PUL requires satisfying specific assumptions, as outlined in [Bekker and Davis \(2020\)](#). In light of this, our approach incorporates several of these assumptions, which are enumerated subsequently:

- (1) **Separability:** The data used for PUL is assumed to be separable, that is positive and negative classes are naturally separable, indicating the existence of a classifier that can perfectly distinguish between the two. This assumption facilitates the development of approaches that concentrate on delineating a clear boundary between these two classes.
- (2) **Smoothness:** This assumption states that samples close to each other in the feature space are likely to share the same label, thereby enabling the application of techniques such as clustering. This enhances learning by assuming that proximity in spatial or feature dimensions is indicative of label similarity.
- (3) **Unlabeled data contains positive and negative samples:** In PUL, only a subset of positive samples is labeled, while negative samples remain without explicit labels. Consequently, our unlabeled dataset comprises both positive samples that were not chosen for labeling and negative samples. This constitutes a core principle of PUL, differentiating it from other learning methods such as supervised learning, where all instances are explicitly labeled.

- (4) **Selected Completely At Random (SCAR):** This assumption implies that the labeled samples are a uniform subset of the positive samples, selected without bias towards any specific attributes, which simplifies the application of PUL by enabling it to be treated like a binary classification problem.

4.3.2 Data Collection and Pre-processing Module and Feature Extraction Module

Since this study extends the research presented in Inter-Slice Defender [Molina et al. \(2024\)](#), our enhanced methodology for handling contaminated data retains the same data collection and preprocessing module, as well as the feature extraction module, described in the original architecture. Both modules perform the same designated tasks as in the previous implementation (Section 3.3.1 and Section 3.3.2), ensuring methodological consistency. The data collection and preprocessing module gathers both unlabeled data and a subset of positively labeled data (such as PCAP files, 3GPP KPIs, and performance measurement counters) from various 5G network functions. These data sources capture signaling interactions between the RAN and the core, reflecting procedures such as registration and ISS. The collected PCAP files are merged chronologically to preserve event sequence integrity and are preprocessed to extract key attributes (e.g., IP addresses, port numbers, and protocol headers), facilitating the construction of time-series data. The subsequent feature extraction module transforms this preprocessed data into measurable indicators of network behavior. Specifically, it derives 17 3GPP-standardized features representing KPIs and PM counters across network functions such as the AMF, SMF, and NSSF. These features encode both normal and anomalous network states associated with DSM attacks and are normalized and filtered using a variance threshold to retain only the most informative ones. The resulting time-series features provide structured, noise-reduced inputs suitable for LSTM-Autoencoder training and K-means clustering, enabling the PUL-Inter-Slice Defender to effectively distinguish between benign and attack patterns despite the presence of contaminated training data.

4.3.3 PUL-Anomaly Detection Module

The PUL-anomaly detection module (Fig. 4.1b) leverages our approach's training process, incorporating PUL. This module utilizes an LSTM-autoencoder architecture combined with K-means

clustering, guided by a PUL strategy that can be considered a variation of the two-step PUL technique. Standard two-step PUL techniques comprise two steps [Bekker and Davis \(2020\)](#). The first step involves the extraction of negative, and optionally positive samples from the unlabeled dataset. The second step involves training a model using the identified negatives alongside labeled positive samples [Bekker and Davis \(2020\)](#). However, in our approach, we first use the encoder of the LSTM-autoencoder to extract features (step 1) that characterize positive and negative samples, and then apply K-means clustering to separate these features into positive and negative clusters (step 2). These two steps are realized in conjunction with the pattern recognition sub-module and the clustering sub-module of the PUL-anomaly detection module, detailed hereafter.

Pattern Recognition Sub-module. Pattern recognition sub-module is developed to extract the most critical patterns, correlations, temporal dependencies, and informative features from those provided by the feature extraction module. It employs an LSTM-Autoencoder [Said Elsayed et al. \(2020\)](#) to this end, training and preparing its encoder for effective feature extraction in the subsequent sub-module (i.e., the clustering sub-module). The selection of an LSTM-Autoencoder is based on its capability to effectively manage sequences and time-based patterns through LSTM. This capability is coherently integrated with an Autoencoder, which learns an abstract representation of its inputs (i.e., the normal behavior of the network and attack scenarios), accomplishing this through the compression and reconstruction of the input data [Said Elsayed et al. \(2020\)](#). The encoder effectively reduces the input dimensionality, thereby generating a latent space representation that captures the most significant characteristics and relationships in the training dataset, while the decoder is tasked with reconstructing the original input data from the compressed representation generated by the encoder. Thus, LSTM-Autoencoder yields a good depiction of the 5G network under both normal conditions and DSM attack scenarios, where the dependencies and precedence constraints of the various 5G procedures that represent ISS events are thoroughly detailed. For instance, an ISS event to a new NS cannot occur before deregistering from the current NS and registering with the new one (Section [3.4.2](#)). Furthermore, the frequency of normal network behavior and attack patterns over time can be captured by an LSTM through its cell capabilities, which can remember values over various time intervals [Said Elsayed et al. \(2020\)](#).

Clustering sub-module. The clustering sub-module is designed to identify and classify unlabeled data as positive or negative samples, alleviating the problem of learning the decision boundary without a fully labeled dataset and achieving the goal of the PUL approach. Additionally, this module is employed to detect DSM attacks.

Upon completing the training phase of the LSTM-autoencoder model in the pattern recognition sub-module, its encoder component is employed by the clustering sub-module to extract features from the data processed by the feature extraction module, which characterized the unlabeled data and the available subset of positively labeled data. Inside the encoder, these features are compressed and encoded into a compact and meaningful representation that preserves the essential characteristics of the sequential data that represent normal and attack behavior. This refined representation is leveraged by the K-means clustering algorithm. K-means clustering is an unsupervised machine learning algorithm, simple and easy to implement, and widely used in anomaly detection applications [Jain, Kaur, and Saxena \(2022\)](#). It is designed to partition a dataset into K distinct, non-overlapping clusters, assigning each data point to one of the K clusters based on predefined similarity measures. By iteratively updating cluster centroids and reallocating data points, K-means effectively identifies the underlying structure within the data.

In our approach, K-means clustering algorithm leverages the extracted features from the encoder to enhance its performance by improving clustering accuracy, thereby potentially augmenting the separability of the samples, which aligns with the smoothness and separability assumptions described in Section [4.3.1](#). This process facilitates the grouping of the data into two distinct clusters: one comprising positive samples (i.e., benign instances), and the other encompassing negative samples (i.e., attack instances). Furthermore, given the inclusion of an initial subset of positively labeled data within the training dataset (Section [4.4.1](#)), it becomes feasible to assign labels to each cluster. By mapping these labeled samples to the resulting clusters, this subset facilitates the determination of which cluster has to be designated as positive and which as negative.

Once the K-means training is completed, new data entries are processed through the predefined sequence involving LSTM-Autoencoder's encoder for feature extraction followed by clustering. The classification of a new data point, whether it is anomalous or not, is then assessed based on its association with a specific cluster. This assessment relies on the premise that the clusters distinctly

represent either normal or anomalous instances.

4.4 Datasets Generation

Given the lack of publicly available 5G datasets suitable for training and evaluating our PUL-Inter-Slice Defender, we created a set of contaminated datasets derived from the simulations of normal and attack network traffic described in 3.4.2 of Inter-Slice Defender [Ayala Molina, Wehbe, Alameddine, Pourzandi, and Assi \(2024\)](#). Those simulations provided a realistic representation of 5G CP signaling behavior under benign conditions and DSM attacks, specifically the RSA and TSA. Building on these previously generated data, we constructed datasets containing unlabeled samples alongside a subset of positively labeled ones, consistent with the PUL paradigm. This approach reflects real-world 5G environments where clean, fully labeled data are seldom available, allowing our model to learn and adapt to contaminated conditions that include a mixture of benign and attack samples. Consequently, these contaminated datasets serve as the foundation for training and testing the PUL-Inter-Slice Defender.

Table 4.1: Datasets statistics.

Dataset type	Contamination	No. of records	Benign records	Attack records	
				RSA	TSA
Training Dataset	10%	40000	36000	2000	2000
Training Dataset	20%	40000	32000	4000	4000
Training Dataset	30%	40000	28000	6000	6000
Training Dataset	40%	40000	24000	8000	8000
RSA Test dataset	—	20000	10000	10000	0
TSA Test dataset	—	20000	10000	0	10000

4.4.1 Datasets for Anomaly Detection Model

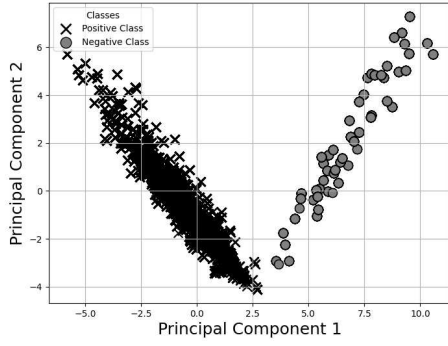
The data generated from the aforementioned emulations is used to create different datasets to facilitate the training and evaluation of PUL-Inter-slice Defender model.

Training Dataset

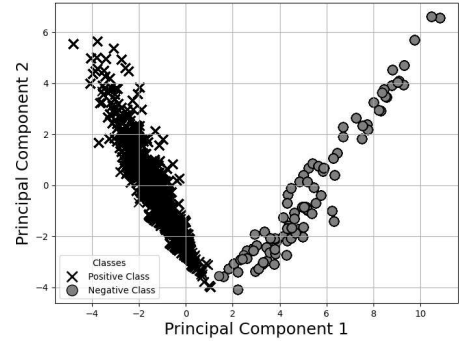
To develop an effective anomaly detection solution for contaminated data, we employ four training datasets (Table 4.1) encompassing different proportions of both benign samples, representing

the positive class, and DSM attack samples (i.e., RSA and TSA samples), representing the negative class. Specifically, the proportions of attack samples are set at 10%, 20%, 30%, and 40% of the total training dataset, with the remaining samples being benign. The proportions of benign versus attack samples in our dataset are selected to allow careful evaluation of PUL-Inter-Slice Defender robustness under different training conditions. In fact, small proportions of attack samples (e.g., 10%, 20%) in the training dataset reflect model training using data collected during normal network operations, which may include some anomalies that were not identified. In contrast, higher proportions of attack samples (e.g., 30%, 40%) are considered to further evaluate the robustness of our solution under severe contamination of the training data in case of adversarial attacks, for example. Each dataset includes 10% labeled benign data, identified by a security expert who randomly selects a diverse and representative subset of positive samples, ensuring unbiased and high-quality data. This enhances the effectiveness of our PUL-anomaly detection module’s binary classifier, as established by the SCAR assumption (Section 4.3.1). It also aligns with the PUL assumption, which asserts that a dataset contains a subset of labeled positives and unlabeled samples, where the latter comprises both positive and negative samples (Section 4.3.1).

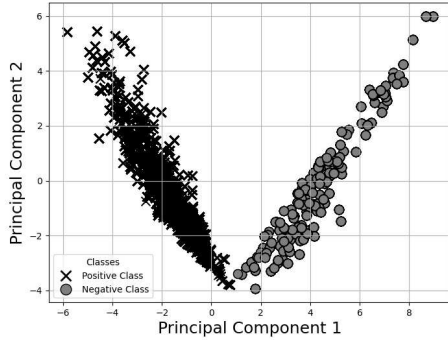
Furthermore, to analyze the distribution and assess the class separability and smoothness assumptions within our training dataset, we employed PCA [Greenacre et al. \(2022\)](#). PCA reduces the dataset’s dimensionality to two dimensions, enabling the visualization of data in a scatter plot to verify the separability of classes in a reduced-dimensional space. Fig. 4.2 illustrates a clear separation between positive and negative classes across all four training datasets, indicating that they are distinguishable within the transformed feature space. This suggests that training a model with positive and unlabeled data could effectively differentiate between positive and negative samples. Fig. 4.2 also demonstrates that our training datasets meet the smoothness assumption across the different proportions of positive and negative samples. Both classes, positive and negative, exhibit compact clustering within each class, indicating similarity and supporting this principle. Furthermore, the boundary between classes is fairly distinct with minimal overlap, reinforcing that data points of the same class are closer to each other than to those of the other class.



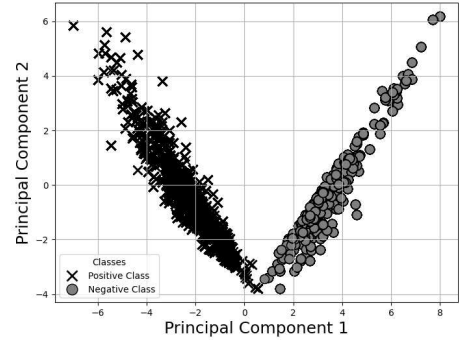
(a) 10% of negative samples



(b) 20% of negative samples



(c) 30% of negative samples



(d) 40% of negative samples

Figure 4.2: 2D PCA plot showing positive and negative classes in the training dataset with different levels of negative samples

Test datasets

To rigorously evaluate our anomaly detection model, we use two test datasets, each containing benign and malicious data from RSA and TSA attack emulations (Section 3.4.2). This approach thoroughly assesses the model’s ability to distinguish between normal and anomalous patterns.

It is crucial to highlight that the training and test datasets are meticulously designed to be mutually exclusive, ensuring that there are no redundant records between them.

4.5 Experimental Results

In this section, we evaluate PUL-Inter-Slice Defender and assess its effectiveness in detecting RSA and TSA when its training dataset contains samples reflecting normal behavior in the 5G

network and samples indicating DSM attacks.

4.5.1 LSTM-Autoencoder Architecture Selection

To determine the optimal architecture for the PUL-Inter-Slice Defender’s LSTM-Autoencoder, which effectively learns patterns and dependencies in the sequential data representing normal and attack behaviors in our training dataset, we train and validate multiple architectures and evaluate their performance. To this end, we use a training dataset composed of 30% attack data and 70% benign samples (Table 4.1), as a dataset with a moderate proportion of negative samples allows our proposed approach to learn a more diverse representation of what constitutes a negative sample, which is crucial in PUL for avoiding false positives and preventing the model from overfitting to the positive class. Thus, we allocate 20% of the training dataset as a validation dataset and train the model using its remaining portion (Table 4.1). To select the model hyperparameters, we apply the K-fold cross-validation technique [Wong and Yeh \(2020\)](#), which helps prevent overfitting and thoroughly evaluates model performance. Specifically, we train the model using a batch size of 32, a learning rate of 0.01, and the Adam optimizer, with a dropout rate of 0.2 to mitigate overfitting. We use ReLU as the hidden activation function and Mean Squared Error (MSE) as the loss function. The model is trained for up to 50 epochs. Additionally, we incorporate early stopping, which continuously monitors the cross-validation set’s performance throughout the training process. If the validation error starts to increase, or if there is no improvement over a predetermined number of epochs, the training is stopped and the model with the best performance on the cross-validation set is selected as the final model.

We test and evaluate various LSTM-Autoencoder architectures and select {100, 50, 50, 50, 100} as it depicts the best performance in learning the characteristics of the training data and reconstructing them with low reconstruction errors. It is composed of two LSTM layers of 100 and 50 neurons, respectively, forming the encoder, and a decoder having the encoder’s mirrored architecture. The code of the LSTM-Autoencoder is 50 neurons. We train our model with the selected architecture (Table 4.2 after fine-tuning its hyperparameters).

Table 4.2: LSTM-Autoencoder Training Parameters

Parameter	Value / Setting
Training dataset composition	70% benign samples, 30% attack samples
Batch size	32
Learning rate	0.01
Optimizer	Adam
Dropout rate	0.2
Hidden activation function	ReLU
Loss function	Mean Squared Error (MSE)
Number of epochs	50 (with early stopping)

4.5.2 K-means architecture

We apply the K-means clustering algorithm using the standard implementation provided by the scikit-learn library [scikit-learn developers \(2024\)](#). The number of clusters is set to two to align with the binary classification objective of our approach. To achieve consistent model behavior, the random seed is fixed to 42. The algorithm adopts the “k-means++” initialization method to enhance centroid selection and accelerate convergence. To improve clustering robustness, the model is configured to run ten times with different centroid seeds. The maximum number of iterations allowed for convergence is defined as 300, and the tolerance threshold for convergence is specified as $1e-4$. These settings, which are summarized in Table 4.3, ensure a stable and effective clustering process, enabling the model to accurately separate benign and attack patterns within the latent feature space extracted by the LSTM-Autoencoder.

Table 4.3: K-means Clustering Parameters

Parameter	Value
Number of clusters (<code>n_clusters</code>)	2
Random seed (<code>random_state</code>)	42
Initialization method (<code>init</code>)	<i>k-means++</i>
Number of initializations (<code>n_init</code>)	10
Maximum iterations (<code>max_iter</code>)	300
Convergence tolerance (<code>tol</code>)	$1e-4$

4.5.3 Benchmark Models

To conduct a thorough evaluation of our proposed solution, we compare the PUL-Inter-Slice Defender against three benchmark models. The first is the original Inter-Slice Defender [Molina et al. \(2024\)](#), which serves as a natural baseline since our PUL-based model builds upon its architectural foundation.

In addition, we implemented two custom PUL-based models: PUL-OCSVM-RF and PUL-OCSVM-XGBoost, to serve as comparative baselines within the PUL paradigm. Both models follow the same data pre-processing and feature extraction pipeline used in our main solution (Section 3.3.1, Section 3.3.2). In accordance with the standard two-steps PUL approach [Bekker and Davis \(2020\)](#), both models first apply OCSVM to the positively labeled benign samples to identify reliable negatives from the unlabeled set. OCSVM has strong capability in detecting outliers and anomalies in high-dimensional data [Agyemang \(2024\)](#), making it particularly effective for identifying reliable negatives within the PUL context. It employs an Radial Basis Function (RBF) kernel with $\gamma = \text{'scale'}$. The ν parameter, which controls the trade-off between training errors and model complexity, was set to 0.025 for *PUL-OCSVM-RF* and 0.03 for *PUL-OCSVM-XGBoost*. Samples predicted as -1 (i.e., attacks) were selected as reliable negatives. These reliable negatives were then combined with the labeled benign data to train a supervised binary classifier. For the PUL-OCSVM-RF model, we utilize a RF classifier, whose ensemble nature helps improve classification accuracy and robustness when working with labeled data [Awotunde et al. \(2023\)](#), making it particularly suitable for the second step of the two-step PUL technique [Bekker and Davis \(2020\)](#). It was configured with 100 estimators and a fixed random seed of 42 to ensure reproducibility. For the PUL-OCSVM-XGBoost model, we train an XGBoost classifier, selected for its demonstrated success in supervised anomaly detection and its ability to efficiently handle structured data, making it well suited for classification tasks within the PUL setting [Dilworth and Gudla \(2024\)](#). The model is configured with the label encoder disabled, a fixed random seed of 42, an evaluation metric based on log loss, 100 estimators, a maximum tree depth of 5, and a learning rate of 0.1.

These three benchmark models (Inter-Slice Defender, PUL-OCSVM-RF, and PUL-OCSVM-XGBoost) were trained and evaluated using the same datasets as the PUL-Inter-Slice Defender

(Section 4.4.1), including training sets with varying levels of contamination and a common set of test datasets. The corresponding evaluation results are presented in the following subsection.

4.5.4 PUL-Inter-Slice Defender Performance

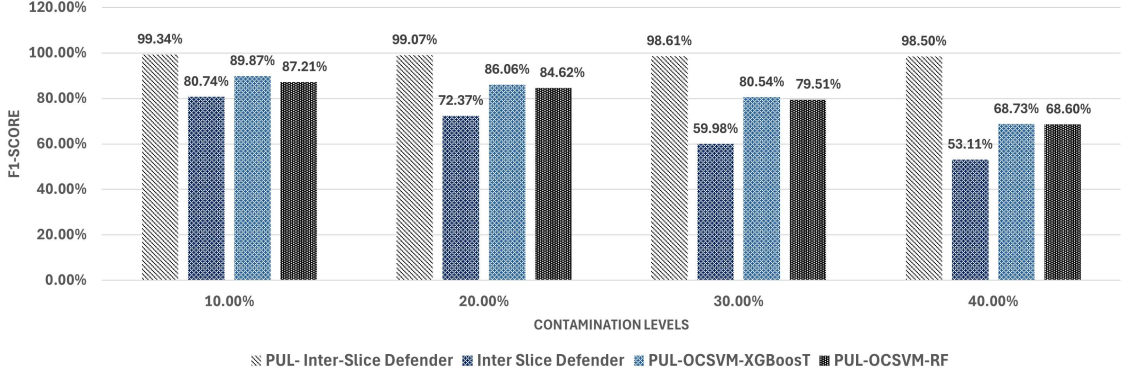


Figure 4.3: F1-score comparison of PUL-Inter-Slice Defender, Inter-Slice Defender, PUL-OCSVM-RF, and PUL-OCSVM-XGBoost for detecting RSA and TSA under varying training data contamination levels (10%–40%).

We now present the performance of the proposed PUL-Inter-Slice Defender and compare it against the benchmark models (Section 4.5.3) across varying contamination levels.

We retrain Inter-Slice Defender using each of our four training datasets (Section 4.4.1) to evaluate its performance under different levels of contamination (i.e., 10%, 20%, 30%, 40%). The model’s lookback hyperparameter (i.e., timesteps) is fixed at 1, and the detection threshold is set to 0.1408, based on findings from Molina et al. (2024), which identified these values as optimal for detection performance. Fig. 4.3 shows Inter-Slice Defender’s F1-score values ranging from 80.74% to 53.11% as contamination levels increase from 10% to 40% for detecting RSA and TSA. This performance drop is attributed to the model’s assumption of a clean training dataset (i.e., entirely benign), an assumption that does not hold in real-world deployments, where contamination is frequently present and compromises the model’s practical effectiveness.

In contrast, the performance of PUL-OCSVM-XGBoost and PUL-OCSVM-RF reveals a clear advantage compared to the Inter-Slice Defender across all contamination levels (10%, 20%, 30%, and 40%), as shown in Fig. 4.3, highlighting the benefits of PUL. Specifically, PUL-OCSVM-XGBoost maintains an F1-score of 89.87% at 10% contamination, which gradually decreases to

68.73% as the contamination level reaches 40%. Similarly, PUL-OCSVM-RF records an F1-score ranging from 87.21% to 68.60% over the same contamination range. Despite these improvements, both PUL-OCSVM-XGBoost and PUL-OCSVM-RF models exhibit a noticeable decline in performance as contamination increases, reflecting their limited capacity to capture temporal dependencies in the data. This poses a critical limitation for DSM attack detection, where RSA and TSA unfold through sequences of 5G control procedures such as registration, authentication, and PDU session establishment. These procedures generate time series patterns essential for distinguishing anomalous from benign behavior. While RF treats each input independently and is not tailored for capturing temporal dependencies; XGBoost does not account for sequential dependencies like LSTM. This limitation can lead to misclassifications when subtle temporal patterns become decisive under high contamination conditions.

On the contrary, PUL-Inter-Slice Defender demonstrates superior performance in detecting both RSA and TSA attacks, as illustrated in Fig. 4.3. It achieves F1-scores ranging from 99.34% to 98.50% across training datasets with contamination levels varying between 10% and 40%, significantly outperforming the benchmark models: Inter-Slice Defender, PUL-OCSVM-XGBoost, and PUL-OCSVM-RF. This performance improvement is attributed to three key architectural strengths:

- (1) The integration of LSTM-autoencoder to effectively handle the sequential nature of the data. In the context of DSM attacks such as RSA and TSA, the behavior of the system unfolds over time through a series of control plane procedures. These procedures naturally generate time series data, where the order and timing of events are critical for identifying normal and abnormal patterns.
- (2) The model leverages the LSTM-autoencoder’s latent space to extract abstract features, facilitating effective feature clustering and pattern separation.
- (3) PUL enhances adaptability by enabling robust decision boundary learning from limited labeled positives and a large pool of unlabeled samples, making it well-suited for real-world scenarios with contaminated training data.

Overall, by combining PUL with the LSTM-autoencoder’s latent space, the PUL-Inter-Slice Defender effectively captures the sequential structure and timing of 5G CP procedures. This enables

robust detection under contaminated training conditions and allows the model to generalize well despite label uncertainty and the complexity of time-dependent behaviors during a DSM attack.

4.6 Implementation Considerations in Real-World 5G Environment

PUL-Inter-Slice Defender leverages 3GPP KPIs and PM counters as features for DSM attack detection. These metrics are usually calculated by different 5G NFs and shared with the Network Data Analytics Function (NWDAF), which is a core 5G NF designed to collect, process, and analyze data across various network components [Garcia-Martin et al. \(2023\)](#). NWDAF provides a granular view of network behavior and can be configured to monitor targeted NSs and NFs such as AMF, SMF, and NSSF, which are critical components for maintaining the operational integrity and security of the 5G network. By leveraging NWDAF's data collection and analytics capabilities, PUL-Inter-Slice Defender enhances its effectiveness in detecting DSM attacks, offering a data-driven approach that strengthens the overall resilience of the 5G network. Additionally, PUL-Inter-Slice Defender can complement existing intrusion detection systems to trigger automated responses such as rate limiting or slice isolation, mitigating threats before they escalate.

However, deploying PUL-Inter-Slice Defender in real-world 5G networks presents several challenges, particularly due to the fragmented nature of NS deployments across multiple administrative domains. These domains often enforce distinct security policies and data-sharing restrictions, limiting the holistic analysis of network behavior. Privacy concerns further complicate collaboration, as operators may be unwilling to share sensitive operational data. These limitations hinder centralized anomaly detection and necessitate approaches that balance accuracy, efficiency, and data privacy.

4.6.1 Federated Learning for PUL-Inter-Slice Defender

PUL-Inter-Slice Defender's integration with NWDAF should support distributed monitoring, enabling each operator's administrative domain to maintain autonomy while contributing to a collective detection framework. To attain such an objective, Federated Learning (FL) [Wen et al. \(2023\)](#) can be used. Through FL, each operator trains its own instance of the framework locally using

its available KPIs and PM counters. Instead of sharing raw data, only model updates (e.g., encoder weights or cluster centroids) are exchanged and securely aggregated (e.g., via federated averaging), ensuring that sensitive operational or subscriber data never leaves the operator’s domain [Akhtarshenas et al. \(2024\)](#). Techniques such as differential privacy or homomorphic encryption [Lu, Pan, Dai, Si, and Zhang \(2024\)](#) can be employed to further ensure secure and private sharing of model updates, thereby protecting against data leakage and adversarial attacks. In addition, as operators may adopt different NS configurations and security policies, the problem of non-Independent and Identically Distributed (non-IID) data may arise, leading to challenges in achieving robust detection performance and model generalization. Techniques such as those noted in [Lu et al. \(2024\)](#) (e.g., clustering-based approaches) can be used to address this issue.

4.6.2 Concept Drift Considerations

Network behavior may change over time, and new variations of DSM attacks can appear. This can cause concept drift that can degrade PUL-Inter-Slice Defender detection performance. To address this problem in practical deployments, periodic model retraining or incremental updates through continual learning using recent network data can ensure sustained detection accuracy. Therefore, investigating concept drift induced by RSA and TSA and designing adaptive maintenance strategies to mitigate its effects remains an important open research direction in the dynamic environment of 5G networks.

Chapter 5

***IS*²-DriftGuard: Concept Drift-Based Adaptive Detection of Distributed Slice Mobility Attacks**

5.1 Introduction

This chapter presents *IS*²-DriftGuard, an adaptive anomaly detection and drift adaptation framework that builds upon PUL-Inter-Slice Defender to address the challenges of evolving DSM attack patterns in dynamic 5G environments. It introduces the design and implementation of an adaptive detection mechanism capable of maintaining reliable performance under incremental concept drift, where continuous changes in attack network traffic distributions may affect detection performance over time. To achieve this, *IS*²-DriftGuard is designed with two main modules: a Distributed Slice Mobility Drift Generator, which leverages a CWGAN-GP to simulate realistic drifted RSA and TSA distributions, and a Concept Drift Adapter, which integrates an LSTM-Autoencoder and MiniBatch KMeans clustering with a custom multi objective loss combining MMD, EWC, SIR, and MSE. Together, these components enable continual learning by preserving previously acquired knowledge while adapting to new behavioral patterns. The chapter also describes the experimental

setup and evaluates IS^2 -DriftGuard against PUL-Inter-Slice Defender under incremental drift conditions. The experimental evaluation demonstrates that IS^2 -DriftGuard achieves F1-scores above 90%, confirming its capability to sustain robust detection performance amid evolving adversarial behaviors in 5G networks.

5.2 PUL-Inter-Slice Defender and Concept Drift

This section briefly describes the PUL-Inter-Slice Defender [Ayala Molina et al. \(2025\)](#), a PUL anomaly detection solution that detects DSM attacks and serves as a baseline to compare our proposed solution against. It first highlights the model’s performance on contaminated data and highlights its limitations when exposed to incremental DSM drift. Then, it discusses the key challenges involved in managing such a drift within the PUL and CL paradigms.

5.2.1 PUL-Inter-Slice Defender: Overview and Shortcomings

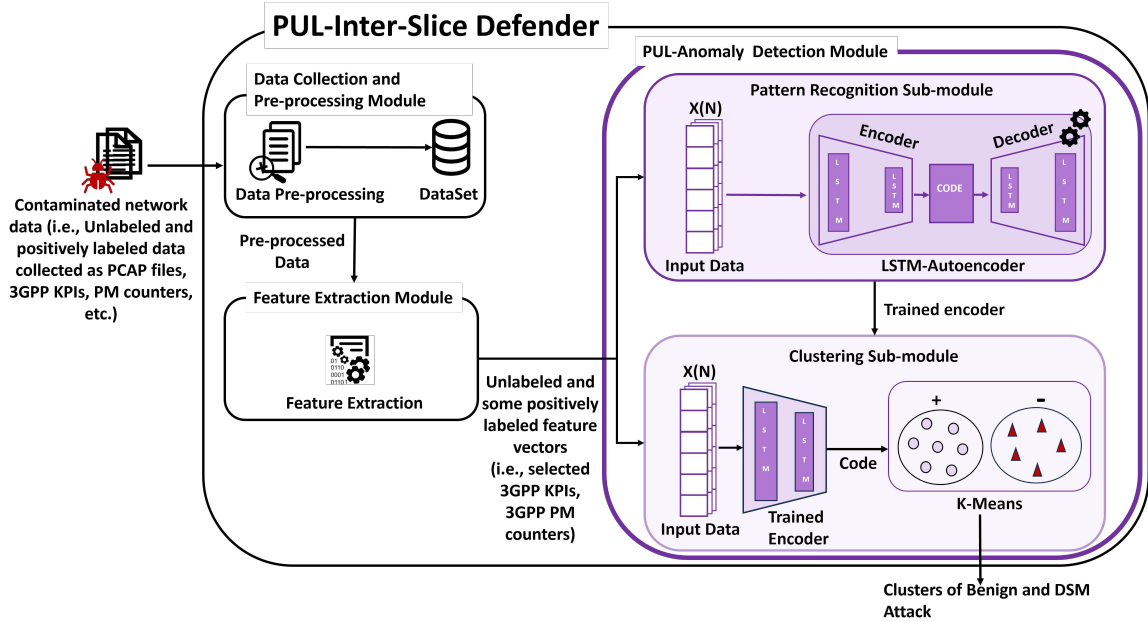


Figure 5.1: PUL-Inter-Slice Defender.

PUL-Inter-Slice Defender [Ayala Molina et al. \(2025\)](#) (Figure 5.1) extends the architecture of the

original Inter-Slice Defender [Ayala Molina et al. \(2024\)](#) by incorporating PUL to enhance robustness against contaminated training data. This solution combines LSTM-Autoencoder with K-means clustering for feature extraction and anomaly detection while using 3GPP KPIs and PM counters to characterize both normal and malicious slice mobility behavior. A notable strength of the PUL-Inter-Slice Defender is its capacity to identify RSA and TSA attacks even in the absence of explicitly labeled attack samples. It leverages the advantages brought by PUL, which consists of training using a small set of positively labeled (benign) instances alongside a larger pool of unlabeled data that may contain both benign and malicious samples. This design enables reliable inference in real-world scenarios where attack labels are often incomplete or unavailable. Across varying levels of contamination in the training data, ranging from 10% to 40%, PUL-Inter-Slice Defender achieves an F1 score exceeding 98% [Ayala Molina et al. \(2025\)](#).

However, despite its enhanced resilience to contaminated data, PUL-Inter-Slice Defender assumes that the underlying data distribution remains stable over time. This assumption does not hold in the presence of adversarial concept drift, where RSA and TSA attacks may evolve to exhibit new behavioral patterns. Under such conditions, its effectiveness declines notably from an F1-score of 96.29% to 80.11% (Table 5.5) when tested under varying drifted RSA and TSA distributions (Section 5.5.8). This degradation in performance highlights a key limitation in coping with evolving RSA and TSA attack patterns, particularly in changing environments where adversaries employ stealthy techniques. Hence, there is a need for adaptive detection mechanisms that can identify distributional shifts in data and maintain robust performance under adversarial drift.

5.2.2 Concept Drift Adaptation in PUL: The Challenges

In settings characterized by incremental concept drift and contaminated training data, where PUL intersects with CL to detect and adapt to evolving distributions, several critical challenges arise during model training and adaptation. These challenges are outlined below.

- **Uncoordinated Models Adaptation (C1):** In the presence of incremental concept drift, the interaction between two models (e.g., LSTM-Autoencoder with K-means in [Ayala Molina et al. \(2025\)](#) (Figure 5.1)), poses a significant challenge within the PUL and CL. As the data distribution evolves, updating one model may unintentionally destabilize the other, particularly

in CL settings. Without coordinated adaptation, such misalignment can lead to performance degradation and reduced system robustness.

- **Forgetting previously acquired knowledge (C2):** As the model incrementally adapts to new data, there is a risk that updates will overwrite information acquired from earlier phases of learning. This phenomenon, known as catastrophic forgetting [Aleixo, Colonna, Cristo, and Fernandes \(2023\)](#); [van de Ven, Soures, and Kudithipudi \(2024\)](#), is especially problematic in CL, where long-term retention of prior knowledge is crucial for detecting recurring patterns and maintaining performance consistency across time.
- **Loss of discriminative power in latent representations (C3):** Incremental updates to the encoder, such as the encoder in the LSTM-Autoencoder of PUL-Inter-Slice Defender (Section 5.2.1), may lead to the emergence of redundant or highly correlated latent features, particularly in the presence of noisy or overlapping data. This redundancy reduces the diversity and separability of the learned features [Wu, Yu, Zhai, and Liu \(2025\)](#), making it difficult for some downstream tasks, like clustering (e.g., K-means of PUL-Inter-Slice Defender), to effectively distinguish between evolving benign and malicious classes.
- **Latent space instability due to distributional shift (C4):** Incremental concept drift causes subtle but persistent changes in the input distribution, which in turn lead to gradual drift in the model’s latent representations (e.g., LSTM-Autoencoder latent space in PUL-Inter-Slice Defender). Without proper constraints, this can result in unstable latent encodings that drift away from previously learned representations. Such misalignment impairs downstream tasks such as clustering and undermines the detection of emerging malicious behaviors.

These challenges, driven by incremental concept drift within the context of PUL and CL paradigms, highlight the difficulty of maintaining model’s stability and adaptability over time. Latent instability, forgetting, reduced discriminative power, and uncoordinated models’ interaction threaten the overall robustness of the anomaly detection solution. Addressing these issues calls for a solution that preserves prior knowledge, stabilizes latent representations, and ensures synchronized adaptation to evolving data.

5.3 IS^2 -DriftGuard

IS^2 -DriftGuard (Figure 5.2), our novel anomaly detection and drift adaptation framework, comprises two main modules designed to detect and adapt to incremental concept drift in DSM attacks while leveraging the principles of PUL and CL. The first module, the distributed slice mobility drift generator, employs a CWGAN-GP to generate class-conditioned, drifted samples of RSA and TSA traffic, simulating adversarial evolution. The second module, the concept drift adapter, integrates Wasserstein distance, an LSTM-Autoencoder, and MiniBatch K-means clustering to monitor distributional shifts and enable adaptive anomaly detection. To facilitate continual drift adaptation, a custom loss function that combines MMD, EWC, SIR, and MSE allows the LSTM-Autoencoder to retain previously acquired knowledge while learning new behavioral patterns.

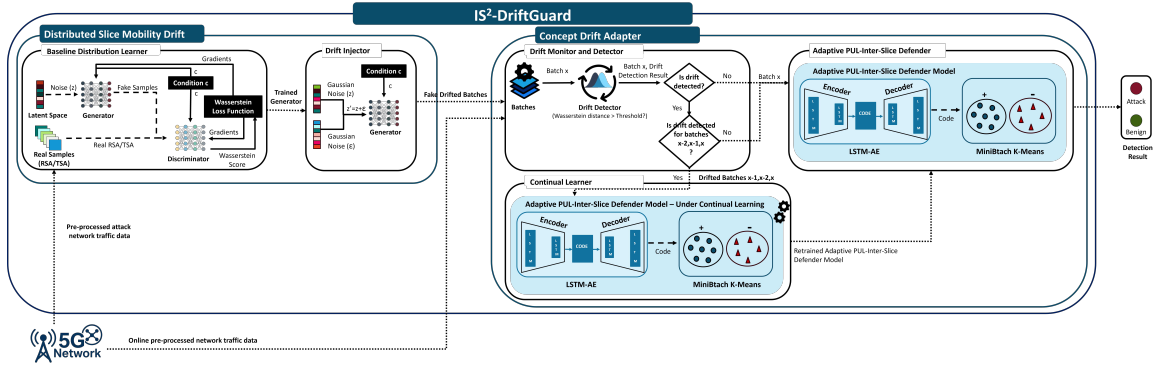


Figure 5.2: IS^2 -DriftGuard's detection and adaptation approach.

5.3.1 Distributed Slice Mobility Drift Generator

In the context of detecting DSM attacks, where attackers can strategically evolve their attack strategies, it is essential to anticipate and model adversarial shifts in data distributions. To address this, the *Distributed Slice Mobility Drift Generator* (Figure 5.2) is designed with a synthetic data generation solution based on CWGAN-GP to simulate realistic and controllable incremental concept drift. It comprises two sub-modules: the *Baseline Distribution Learner*, which trains a CWGAN-GP model to replicate the baseline behavior patterns observed in RSA and TSA sequences, and the *Drift Injector*, which introduces controlled perturbations into the latent space of the trained generator by varying noise parameters. This produces class-conditioned samples that simulate incremental and

realistic deviations from RSA and TSA traffic. We detail these sub-modules in the following.

Baseline Distribution Learner

The baseline distribution learner employs a CWGAN-GP model, which yields an advanced generative model that extends the standard GAN. The latter is a deep learning model composed of two neural networks, a generator, trained to create convincing synthetic data that can fool a discriminator into believing it is a real one, while the discriminator learns to distinguish between real and generated data [Dunmore, Jang-Jaccard, Sabrina, and Kwak \(2023\)](#). Thus, CWGAN-CP follows a GAN architecture but extends it by building upon both the CGAN [Mirza and Osindero \(2014\)](#) and the WGAN-GP [Gulrajani, Ahmed, Arjovsky, Dumoulin, and Courville \(2017\)](#). Through CGAN, it incorporates class conditioning into the adversarial training process, where both the generator and discriminator receive explicit class labels (i.e., 0 for RSA and 1 for TSA) as additional inputs. This enables class-aware training that allows the generation of RSA and TSA-specific sequences. In addition, building on WGAN-GP ensures a more stable training process through employing the Wasserstein loss [Arjovsky, Chintala, and Bottou \(2017\)](#). This is especially important when learning the subtle, high-dimensional patterns of DSM attacks. In addition, CWGAN-GP includes a gradient penalty that further improves convergence and enhances the quality of the generated data [Gulrajani et al. \(2017\)](#).

To effectively model the sequential nature of DSM attacks, we adopt LSTM-based architectures for both the generator and discriminator of CWGAN-GP. This enhanced recurrent neural network (i.e., LSTM) design enables the CWGAN-GP to capture the temporal dependencies and feature correlations present in RSA and TSA data, which are inherently time-series in nature due to the 5G messages that are triggered in order as a result of ISS. By learning these evolving patterns, LSTM can better approximate the dynamic distributions associated with DSM attacks. To further enhance the learning capacity of the LSTM-based generator and discriminator, both models incorporate embedding layers that transform binary condition labels (i.e., 0 for RSA and 1 for TSA) into dense vectors. These embeddings serve as compact yet expressive representations that guide the generation and discrimination processes by encoding semantic class information. In addition to the embeddings, each of the generator and discriminator integrates a dense layer that pre-processes the

inputs before passing them to the LSTM blocks. This design choice helps align the input dimensionality and facilitates a more informative initialization of the temporal modeling process. The combination of dense and embedding layers provides several advantages: 1) it improves the representation of discrete labels; 2) it enables better conditioning of sequential data, and 3) allows the LSTM layers to focus on capturing complex temporal dependencies. Together, these architectural enhancements contribute to the models' (i.e., generator and discriminator) abilities to generate and differentiate realistic RSA and TSA sequences under dynamic network conditions.

To support stable training, the CWGAN-GP is optimized using a conditional Wasserstein loss function with gradient penalty, which not only guides the adversarial learning process but also mitigates issues such as mode collapse. Mode collapse is a common problem in GAN where the generator fails to capture the diversity of the data distribution, which results in highly similar or repetitive outputs [Gulrajani et al. \(2017\)](#).

Further, in our implementation, we extend the WGAN-GP objective by incorporating class labels as conditioning variables, allowing the model to learn class-specific data distributions. The discriminator takes as input a pair consisting of a data sample (i.e., real or generated) and its corresponding label, and is trained to assign higher scores to real samples and lower scores to generated ones, conditioned on the same class label. To maintain the 1-Lipschitz constraint [Gulrajani et al. \(2017\)](#) required for Wasserstein distance estimation, a class-conditioned GP is applied to the discriminator's gradients with respect to its inputs. This class-aware and regularized training setup effectively estimates the conditional Wasserstein-1 distance $W(P_r^c, P_g^c)$, where P_r^c and P_g^c denote the real and generated distributions conditioned on class c (i.e., 0 for RSA and 1 for TSA) respectively. The CWGAN-GP objective function is defined in Eq.(1);

$$\begin{aligned} \min_G \max_D \mathbb{E}_{x \sim P_r^c} [D(x, c)] - \mathbb{E}_{\tilde{x} \sim P_g^c} [D(\tilde{x}, c)] \\ + \lambda \cdot \mathbb{E}_{\hat{x} \sim P_{\hat{x}}^c} \underbrace{(\|\nabla_{\hat{x}} D(\hat{x}, c)\|_2 - 1)^2}_{\text{GP}} \end{aligned} \quad (1)$$

where x represents real data samples drawn from the class-conditioned distribution P_r^c , while $\tilde{x}$ denotes synthetic samples generated by the generator G from P_g^c . The discriminator D takes a sample

x and a condition c as input and outputs a real-valued score indicating how realistic the sample is under that condition. $\hat{x}$ represents interpolated samples obtained by linearly mixing real and generated data, and $P_{\hat{x}}^c$ denotes their distribution conditioned on class c . The GP term $(\|\nabla_{\hat{x}} D(\hat{x}, c)\|_2 - 1)^2$, scaled by a coefficient λ (i.e., set to 1 in our implementation), is computed from the discriminator's gradients with respect to $\hat{x}$. This regularization enforces the 1-Lipschitz constraint and enables stable approximation of the conditional Wasserstein-1 distance between real and generated class-specific distributions.

The overall min-max objective in Eq.(1) defines the adversarial game between the generator and the discriminator. Its optimization process is decomposed into two alternating steps. During each CWGAN-GP training iteration, the discriminator is optimized to maximize the difference between its scores on real and generated samples, while incorporating the GP (i.e., defined in Eq.(1)) [Gulrajani et al. \(2017\)](#). Its loss is computed as in Eq.(2);

$$\mathcal{L}_D = \mathbb{E}_{\tilde{x} \sim P_g^c} [D(\tilde{x}, c)] - \mathbb{E}_{x \sim P_r^c} [D(x, c)] + \text{GP} \quad (2)$$

where GP is as defined in Eq.(1). The generator is then updated separately to minimize the discriminator's score on generated samples, with its loss defined in Eq.(3).

$$\mathcal{L}_G = -\mathbb{E}_{\tilde{x} \sim P_g^c} [D(\tilde{x}, c)] \quad (3)$$

$\tilde{x}$, c , P_g^c , P_r^c in Eq.(2) and Eq.(3) are as defined previously. Ultimately, this alternating training scheme stabilizes convergence, enables the model to learn class-specific data distributions effectively, mitigates the vanishing or exploding gradients, and promotes the generation of high-quality synthetic samples that are reflective of evolving attack categories.

Drift Injector

The drift injector is responsible for generating synthetic samples that reflect adversarial incremental concept drift, capturing how attackers may gradually alter RSA and TSA behaviors to evade detection. This is achieved by introducing controlled perturbations into the latent space of

the trained CWGAN-GP generator. Specifically, noise vectors are first sampled from a standard normal distribution and then perturbed with additive Gaussian noise. By systematically varying the noise parameters (i.e., mean and/or standard deviation) across multiple iterations, class-conditioned synthetic distributions that exhibit incremental shifts from the original RSA and TSA data are generated [Ayala Molina et al. \(2024\)](#). These drifted samples are then passed to the concept drift adapter module which performs inference to evaluate the adaptability of the model under evolving attack patterns. This approach supports systematic assessment of detection systems in a dynamic 5G environment, closely mimicking strategies that adversaries may use to alter DSM traffic characteristics and evade detection.

5.3.2 Concept Drift Adapter

The concept drift adapter comprises three key sub-modules: the *Drift Monitor and Detector*, the *Adaptive PUL-Inter-Slice Defender*, and the *Continual Adapter*. Together, they enable the detection of distributional shifts, support adaptive updates under incremental drift, and maintain robust DSM attack detection.

Drift Monitor and Detector

The drift monitor and detector sub-module is responsible for identifying concept drift in incoming traffic using the Wasserstein distance. The Wasserstein distance is a robust statistical metric that captures both the magnitude and structure of the shift between two probability distributions [Müller, Abdelaal, and Stjelja \(2024\)](#); [Shen, Qu, Zhang, and Yu \(2018\)](#), making it particularly effective in detecting subtle changes in high-dimensional latent spaces. This capability is critical in adversarial settings, where attackers may gradually modify RSA and TSA traffic patterns to mimic benign behavior or diverge from previously observed attack patterns to evade detection. To detect concept drift, this sub-module monitors deviations from a **reference window** in the latent space. When a predefined number (e.g., three) of consecutive batches (i.e., each containing RSA, TSA, and benign samples) exceed a pre-determined **Wasserstein distance threshold**, indicating a sustained shift, the drift monitor and detection sub-module will trigger the continual adapter sub-module to initiate an adaptation process that incrementally updates the DSM attack detection model (i.e., adaptive

PUL-Inter-Slice defender Section 5.3.2) internal parameters within using CL (Section 5.3.2).

The reference window is initially constructed from a portion of the training data that is unaffected by concept drift and that captures pre-drift behavior in the latent space. Upon drift detection, it is incrementally updated using the latent representations of the trigger batches, weighted by a **decay factor** to blend historical patterns with emerging trends. This decay-based update mechanism ensures that the reference window evolves gradually, thus ensuring alignment with the assumption of incremental drift, without abruptly discarding the stable pre-drift state.

In this work, we account for a drift in three consecutive batches to trigger DSM attack detection model re-training to adapt to concept drift. The decision to trigger retraining after three consecutive batches that exceed the Wasserstein distance threshold is based on the following:

- *Responsiveness:* A single or two-batch deviation may result from random noise, transient anomalies, or short-lived fluctuations in traffic patterns, which do not justify immediate model retraining.
- *Stability:* Requiring more than three batches could delay the system’s response to actual incremental drift, allowing model performance to degrade further before corrective action is taken.
- *Incremental Drift Assumption:* IS^2 -DriftGuard is designed to detect incremental drift, hence a consistent increase in Wasserstein distance across three consecutive batches strongly suggests an evolving shift in the underlying distribution, rather than a temporary fluctuation.

Thus, requiring threshold exceedance in three consecutive batches offers a statistically grounded and practically efficient criterion for detecting incremental drift without overfitting to short-term fluctuations, particularly in evolving environments under adversarial conditions such as RSA and TSA attack strategies.

Adaptive PUL-Inter-Slice Defender

This sub-module is responsible for classifying incoming traffic as benign or attack (i.e., RSA or TSA). It builds upon the original PUL-Inter-Slice Defender [Ayala Molina et al. \(2025\)](#), which

applies PUL to detect RSA and TSA traffic in the presence of contaminated datasets. While the original PUL-Inter-Slice Defender leverages latent clustering to infer classes from limited positive labels and abundant unlabeled data, it cannot model temporal dependencies and adapt to evolving DSM distributions (Section 5.2.1).

To overcome these limitations, we extend the original PUL-Inter-Slice Defender architecture [Ayala Molina et al. \(2025\)](#), which consists of an LSTM-Autoencoder and K-means, by integrating a CL mechanism that supports the incorporation of new data while maintaining performance on previously learned distributions, thereby reducing the need for full retraining [Wang, Zhang, Su, and Zhu \(2024\)](#). Specifically, the Adaptive PUL-Inter-Slice Defender augments the LSTM-Autoencoder functionality with a custom loss function that combines MMD, SIR, EWC, and MSE, as detailed in Section 5.3.2. These components collectively enable the model to preserve previously acquired knowledge, maintain clear cluster boundaries, and ensure alignment between latent distributions across time. Additionally, the original K-means algorithm is replaced with MiniBatch K-means to support partial retraining, which is consistent with CL principles that emphasize incremental updates over full retraining [Wahyuningrum et al. \(2021\)](#). This replacement allows the adaptive PUL-Inter-Slice defender model to efficiently adapt to changes in the latent space while preserving previously learned representations. As a result, it offers a robust and scalable solution for DSM attack detection while effectively addressing the challenges posed by contaminated and shifted traffic distributions.

Continual Adapter

The continual adapter sub-module is responsible for retraining the adaptive PUL-Inter-Slice defender upon drift detection using a custom loss function that preserves prior knowledge through CL [Wang et al. \(2024\)](#), as detailed in Section 5.3.2. This loss integrates MSE, EWC, SIR, and MMD, as defined in Eq.(4);

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{MSE}} + \lambda_{\text{EWC}} \cdot \mathcal{L}_{\text{EWC}} + \lambda_{\text{SIR}} \cdot \mathcal{L}_{\text{SIR}} + \lambda_{\text{MMD}} \cdot \mathcal{L}_{\text{MMD}} \quad (4)$$

where $\mathcal{L}_{\text{MSE}}$, $\mathcal{L}_{\text{EWC}}$, $\mathcal{L}_{\text{SIR}}$, and $\mathcal{L}_{\text{MMD}}$ represent the loss terms associated with the MSE, EWC, SIR, and MMD, respectively. The coefficients λ_{EWC} , λ_{SIR} , and λ_{MMD} are regularization weights that

control the relative contribution of each corresponding term to the total loss. Each component of this objective function is defined and discussed in detail below.

- *Mean Squared Error (MSE)*: MSE is a widely used loss function, denoted $\mathcal{L}_{\text{MSE}}$, that quantifies the average squared difference between each input and its reconstruction [Ayala Molina et al. \(2024\)](#). In our LSTM-Autoencoder, it ensures that the model accurately reconstructs the temporal patterns of the input sequences, thereby preserving key structural information during compression and reconstruction. The MSE loss is computed as:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{n} \sum_{i=1}^n \|x_i - \hat{x}_i\|^2 \quad (5)$$

where x_i represents the input sample and $\hat{x}_i$ denotes its associated output representing its corresponding reconstructed version, while n refers to the total number of samples in a batch.

- *Elastic Weight Consolidation (EWC)*:

To address the limitations outlined in C2 (Section 5.2.2), specifically the difficulty of preserving previously learned knowledge while adapting to new patterns, EWC is incorporated into the custom loss function of the adaptive PUL-Inter- Slice defender (Eq.(4)). EWC mitigates catastrophic forgetting, a phenomenon where new learning overrides prior knowledge, leading to degradation in previously acquired capabilities [Aleixo et al. \(2023\)](#); [van de Ven et al. \(2024\)](#). By constraining important weights from shifting too much, it helps the model to remain robust to transient anomalies such as noise, isolated outliers, or short-term traffic fluctuations [Aich \(2021\)](#). The importance of each weight is quantified through the Fisher information matrix [Kirkpatrick et al. \(2017\)](#) which is computed after the initial training. During this process, gradients of the reconstruction loss are computed for randomly sampled inputs, and the Fisher information for each trainable parameter is estimated as the average squared gradient across these samples. For each trainable parameter θ_i , its contribution to the EWC penalty is given by:

$$\mathcal{L}_{\text{EWC}} = \sum_{i=1}^P F_i (\theta_i - \theta_i^*)^2 \quad (6)$$

where P is defined as the total number of trainable parameters; θ_i^* denotes the value of the trained parameter θ_i after the previous task (i.e., following the initial training phase before the drift detection), and serves as a reference point for penalizing future updates, and F_i is the Fisher information value. During retraining, IS^2 -DriftGuard integrates this penalty into the custom loss function (Eq. (4)) alongside MSE, MMD, and SIR. Additionally, the Fisher matrix is dynamically updated using the most recent data from batches identified as exhibiting incremental drift, and it is smoothed by averaging it with its previous version to retain stability across updates. This integration ensures that important parameters are retained while still allowing the model to adapt to evolving distributions of RSA and TSA traffic.

- *Self-Information Regularization (SIR):*

To address the issues described in C3 (Section 5.2.2), which relate to the need for diverse and well-separated feature representations, SIR is applied to the encoded features produced by the LSTM-Autoencoder within the adaptive PUL-Inter-Slice defender. This suggests that the latent representations derived from RSA, TSA, and benign samples remain informative and decorrelated across dimensions Wu et al. (2025). This is accomplished by computing the empirical covariance matrix $\mathbf{C} \in \mathbb{R}^{d \times d}$ of the latent vectors. The diagonal elements, which represent individual variances, are removed to isolate the off-diagonal components capturing inter-feature dependencies. The SIR loss is then defined as:

$$\mathcal{L}_{\text{SIR}} = \sum_{i \in D} \sum_{j \in D: i \neq j} |C_{ij}| \quad (7)$$

where D is the latent dimensionality, and C_{ij} denotes the off-diagonal elements of the covariance matrix. This formulation enhances feature diversity by enabling each latent dimension to encode unique and non-redundant information. As a result, separability within the MiniBatch K-means component is improved, enhancing our model's ability to capture refined variations in attack patterns.

- *Maximum Mean Discrepancy (MMD):*

To address C4 (Section 5.2.2) which pertains to the latent space instability caused by distributional shift, MMD is incorporated into the custom loss function (Eq. (4)). As a non-parametric statistical measure, MMD quantifies the dissimilarity between two distributions in the latent space and is particularly well-suited for detecting and adapting to distributional changes that occur gradually over time [R. Gao et al. \(2021\)](#), aligning well with IS^2 -DriftGuard’s objective of handling incremental drift. MMD provides a flexible and data-driven mechanism to quantify and penalize deviations between latent representations of current input batches and a stable reference distribution [R. Gao et al. \(2021\)](#). In implementation, three pairwise kernel matrices are computed: one within the current batch, one within the reference window, and one between the two. These are evaluated using a Gaussian kernel function, and the MMD loss is obtained by combining them to measure the discrepancy between distributions. This process encourages the latent encoding of newly observed RSA, and TSA samples to remain close to the learned pre-drift representations, thereby promoting smooth and stable adaptation to evolving patterns while preserving continuity in the learned feature space.

The MMD regularization loss is defined as:

$$\mathcal{L}_{\text{MMD}} = \left(\frac{1}{n^2} \sum_{i=1}^n \sum_{j=1}^n k(x_i, x_j) + \frac{1}{m^2} \sum_{i=1}^m \sum_{j=1}^m k(y_i, y_j) - \frac{2}{nm} \sum_{i=1}^n \sum_{j=1}^m k(x_i, y_j) \right) \quad (8)$$

where x_i and x_j denote encoded latent vectors from the current batch of size n , while y_i and y_j correspond to latent vectors from the reference window of size m . The function $k(\cdot, \cdot)$ is a Gaussian kernel that measures the similarity between pairs of vectors. This integration of MMD allows our model to respond effectively to distributional shifts, aligning with the assumptions of incremental drift while mitigating the risk of overreaction to transient fluctuations.

Finally, it is worth noting that each previously described regularization techniques (i.e., MSE,

EWC, SIR, and MMD) contributes, either directly or indirectly, to the production of a robust and informative latent representation that supports effective clustering by MiniBatch K-means, thereby preventing latent space misalignment that could undermine the interaction between these components during continual learning, hence addressing the challenge C1 (Section 5.2.2). More precisely, MSE acts at the reconstruction level, guiding the autoencoder to preserve structural fidelity between input and output. Although this influence is indirect, it encourages the encoder to produce a meaningful and stable latent representation. EWC, however, operates at the parameter level by penalizing large updates to weights that are important for previously learned tasks. This constraint mitigates latent space drift, prevents catastrophic forgetting, and ensures consistent behavior in downstream clustering. SIR acts directly on the latent space by penalizing correlations between different dimensions, promoting decorrelated and informative features that enhance cluster separability. Finally, MMD aligns the distribution of latent representations from current batches with those of a stable reference distribution, thus maintaining consistent cluster structure. This alignment ensures that MiniBatch K-means continues to operate on a stable and coherent latent space, which facilitates accurate and robust clustering despite incremental distributional drift.

5.4 Experimental Setup

In this section, we describe the dataset used for IS^2 -DriftGuard training, validation, and testing, in addition to the environmental setup used for its implementation and evaluation.

5.4.1 Main Dataset Description

In order to train and evaluate IS^2 -DriftGuard; we leverage the dataset introduced in [Ayala Molina et al. \(2025\)](#), which we refer to in the following as *main dataset*. This dataset is specifically designed to reflect both benign and malicious ISS behaviors and is generated using the open-source free5GC testbed [Free5GC \(2023\)](#) that represents a 5G core network. This testbed has been architected to account for multiple network slices and is used in conjunction with UERANSIM [Aligungr \(2023\)](#), a UE/RAN simulator. The dataset captures the temporal dynamics and operational patterns inherent to UEs performing key 5G procedures (e.g., registration, PDU session release and establishment,

de-registration, and ISS). These temporal dynamics are reflected through 17 carefully selected features that are derived from a combination of 3GPP-defined KPIs and PM counters, that capture the internal states and activities of critical 5G Core NFs, such as the AMF, SMF, and NSSF, along with their interactions across multiple NSs under benign and attack (i.e., RSA, TSA) UEs behavior. Examples of such features include the number of registration attempts, success, and failure counts for session creation procedures, slice selection requests, among others. These features are computed and structured in a way that preserves temporal dependencies in the data. This makes the dataset suitable for time series-based ML models such as IS^2 -DriftGuard LSTM-Autoencoder, and enables the latter to learn sequential patterns characteristic of benign versus malicious behavior (i.e., RSA and TSA).

5.4.2 Datasets Preparation

To train the two modules of our framework, namely the distributed slice mobility drift generator and the concept drift adapter module, we prepared four primary datasets. In addition, two calibration datasets were created specifically to establish the initial Wasserstein distance threshold and the decay factor used in IS^2 -DriftGuard (Section 5.3.2), as shown in Table 5.1.

Table 5.1: Datasets statistics (N/A = Not Applicable).

Dataset type	Contamination	No. of records	Benign Records	Attack Records	
				RSA	TSA
Distributed Slice Mobility Drift Generator Datasets (i.e., CWGAN-GP)					
Training Dataset	N/A	15000	0	7500	7500
Validation Dataset	N/A	10000	0	5000	5000
Concept Drift Adapter Datasets (i.e., LSTM-Autoencoder and MiniBatch K-means)					
Training Dataset	10%	32000	28800	1600	1600
Validation Dataset	10%	8000	7200	400	400
Testing Dataset per drifted distribution	N/A	20000	10000	5000	5000
Calibration Datasets					
Wasserstein Dataset	N/A	20000	10000	5000	5000
Decay Dataset	15 testing datasets, each generated as specified in testing dataset per drifted distribution above				

Distributed Slice Mobility Drift Generator Datasets

- *Training Dataset:* This dataset is extracted from the main dataset (Section 5.4.1). It consists of RSA, and TSA samples and is used to train the CWGAN-GP model to generate drifted RSA and TSA data.

- *Validation Dataset:* Composed of 10000 samples (5000 RSA and 5000 TSA), this dataset is used to validate the quality of the generator’s output. It serves to compare the synthesized drifted samples against the original RSA and TSA distributions (i.e., through PCA [Greenacre et al. \(2022\)](#)), to verify the presence of incremental concept drift.

Concept Drift Adapter Datasets

- *Training Dataset:* As IS^2 -DriftGuard extends the PUL framework introduced in [Ayala Molina et al. \(2025\)](#), the adaptive PUL-Inter-Slice defender (Section 5.3.2) is initially trained on a subset of the main dataset (Section 5.4.1) with 10% contamination. This proportion was chosen based on performance trends reported in [Ayala Molina et al. \(2025\)](#) across varying levels of contamination. The dataset is employed to jointly train the LSTM-Autoencoder and MiniBatch K-means components, thereby facilitating coherent feature representation and clustering within this sub-module.
- *Validation Datasets:* This dataset is employed to validate the performance of the adaptive PUL-Inter-Slice defender model during its training. By providing a balanced representation of benign and attack samples obtained from the main dataset (Section 5.4.1), this validation dataset helps tune, monitor, and guide the training process. In particular, it supports the assessment of reconstruction quality, cluster separability, and the early detection of overfitting.
- *Testing Datasets:* We generate 15 different testing datasets, each of which includes benign data extracted from the main dataset (Section 5.4.1) and attack data generated by the CWGAN-GP within the *Distributed Slice Mobility Drift Generator* module (Section 5.3.1). These datasets differ in their attack data obtained by varying the mean and standard deviation (i.e., parameters provided for CWGAN-GP). They are used to assess the performance of IS^2 -DriftGuard in online conditions where attack data can drift. The number of records and their distributions between benign and attack samples for each of these 15 datasets are as specified in Table 5.1.

Calibration Datasets

- *Wasserstein Dataset:*

It is used to compute the initial Wasserstein distance, which serves as both the threshold and reference point for detecting distributional shifts caused by incremental drift (Section 5.3.2). All data points in this dataset are drawn from the main dataset (Section 5.4.1), ensuring a stable, non-drifted reference distribution.

- *Decay Dataset:* This dataset is used to compute the decay factor applied when the reference window is recalculated. The reference window captures the latent distribution of traffic under stable, pre-drift conditions. It is updated upon drift detection by blending historical and new data according to the decay factor, ensuring gradual rather than abrupt adaptation. The decay dataset consists of benign samples from the main dataset [Ayala Molina et al. \(2025\)](#) (Section 5.4.1) and synthetic RSA/TSA traffic produced by the distributed slice mobility drift generator (Section 5.3.1). As such, we use the same 15 testing datasets designed to test the adaptive PUL-Inter-Slice defender (Section 5.4.2) for this purpose.

5.4.3 Environmental Setup

To support the deployment and evaluation of IS^2 -DriftGuard, a Virtual Machine (VM) was configured with the following specifications: 64 GB of RAM, 16 virtual Central Processing Units (vCPUs), and 1 TB of storage. The VM operates on Ubuntu 20.04 LTS, providing a stable and efficient Linux-based environment to conduct all training, testing, and drift adaptation experiments.

5.5 Experimental Results

5.5.1 CWGAN-GP Architecture

To determine the most suitable architecture for the CWGAN-GP model responsible for generating class-conditioned RSA and TSA samples exhibiting incremental drift, we evaluated multiple generator and discriminator configurations. The CWGAN-GP model was trained on a balanced

dataset composed of 50% RSA and 50% TSA samples, as detailed in Table 5.1, ensuring it learns to synthesize realistic, class-specific patterns for both attack types.

CWGAN-GP - Generator Architecture

To identify the best generator architecture, we evaluated multiple candidate designs based on their ability to replicate the statistical properties of the original RSA and TSA distributions. The selected architecture comprises two dense layers with 512 and 768 neurons, respectively, followed by three LSTM layers with 256 neurons each. Batch normalization is applied after each dense layer to stabilize training and accelerate convergence Ioffe and Szegedy (2015). A residual connection is integrated between the first two LSTM layers to improve gradient flow and enhance temporal modeling Kim, El-Khamy, and Lee (2017), while dropout regularization is employed to reduce overfitting. This architecture was ultimately chosen for its consistent ability to generate realistic, class-conditioned sequences that exhibit coherent incremental drift across both attack classes.

CWGAN-GP - Discriminator Architecture

We assessed multiple architectural configurations to identify a discriminator capable of distinguishing between real and generated RSA/TSA sequences under class-conditional settings. The selected discriminator architecture accepts a real or synthetic sequence along with its associated binary condition label. The label is first embedded using a trainable embedding layer and reshaped to align with the temporal dimensions of the input sequence. This embedded representation is then concatenated with the input sequence along the feature dimension to enable conditional discrimination. The concatenated input is processed through a single LSTM layer composed of 100 units, followed by a flattening operation and a final dense layer that outputs a scalar representing the Wasserstein score.

To satisfy the 1-Lipschitz continuity constraint required by the WGAN framework, we apply a GP by interpolating between real and generated samples during training. This regularization enforces smooth discriminator gradients, contributing to stable and effective adversarial training. While the GP is primarily intended to maintain theoretical validity, it also indirectly reduces overfitting by discouraging the discriminator from capturing overly specific patterns that fail to generalize.

Importantly, batch normalization is omitted from the discriminator to preserve the validity of the GP [Gulrajani et al. \(2017\)](#). To maintain a balanced adversarial dynamic, the discriminator is updated three times for every generator update, following the standard WGAN training practice [Arjovsky et al. \(2017\)](#). This training schedule ensures that the discriminator remains strong enough to guide the generator’s learning process.

All selected hyperparameters and final architectural details for both the generator and discriminator are summarized in Table 5.2.

Table 5.2: CWGAN-GP hyperparameters

Hyperparameter	Value	Generator (G) Discriminator (D)
Batch size	16	G, D
Loss function	Wasserstein loss	G, D
Learning rate	0.00005	G, D
Optimizer	RMSprop	G, D
Dropout	0.3	G
Hidden activation function	ReLU	G, D
GP Coefficient (λ)	1	D

5.5.2 LSTM-Autoencoder - Architecture and Continuous Learning

To determine the best architecture for the LSTM-Autoencoder of the adaptive PUL-Inter-Slice defender, which effectively learns the temporal patterns of benign and attack behaviors in the training data while enabling adaptation to evolving RSA and TSA distributions, we adopt and refine the methodology proposed in [Ayala Molina et al. \(2025\)](#). To this end, we use the training and validation datasets dedicated to the concept drift adapter (Table 5.1), both composed of 10% attack samples and 90% benign samples.

We use the model hyperparameters listed in Table 5.3, which include both baseline and newly introduced settings. The baseline configuration, adopted from reference [Ayala Molina et al. \(2025\)](#), includes 50 training epochs, a dropout rate of 0.2, a batch size of 32, MSE as the initial loss function, a learning rate of 0.01, the Adam optimizer, and ReLU as the activation function. To enhance adaptability to evolving RSA and TSA distributions, we introduce a custom loss function that integrates three regularization terms: MMD, EWC, and SIR. The corresponding regularization weights

are $\lambda_{\text{MMD}} = 0.1$, $\lambda_{\text{SIR}} = 0.05$, and $\lambda_{\text{EWC}} = 1.0$. These values are fine-tuned using random search. This hyperparameter optimization technique identifies optimal configurations by randomly sampling combinations from predefined ranges, aiming to minimize the Wasserstein distance between the encoded reference and current representations [Franceschi et al. \(2024\)](#). Whenever drift is detected, the model adjusts its number of training epochs to 100 during retraining to reinforce its ability to generalize to the updated data distribution. We evaluated multiple LSTM-Autoencoder configurations to assess the effectiveness of the custom loss function and enable accurate reconstruction of the initial data distribution and adaptation to changes introduced by incremental drift. Among the tested architectures, the configuration $\{100, 50, 50, 50, 100\}$ demonstrated the best performance. The encoder consists of two LSTM layers with 100 and 50 neurons, respectively, while the decoder reconstructs the input by reversing this structure. The latent representation is captured in a 50-neuron vector.

Table 5.3: LSTM-Autoencoder hyperparameters

Hyperparameter	Value
Initial training epochs	50
Dropout	0.2
Batch size	32
Initial Loss Function	MSE
Learning Rate	0.01
Optimizer	Adam
Hidden Activation Function	ReLU
Custom Loss Function	$\text{MSE} + \lambda_{\text{EWC}}\text{EWC} + \lambda_{\text{MMD}}\text{MMD} + \lambda_{\text{SI}}\text{SIR}$
λ_{MMD}	0.1
λ_{SI}	0.05
λ_{EWC}	1.0
Retraining epochs	100

5.5.3 MiniBatch K-means and Continuous Learning

We apply the *MiniBatch K-Means* clustering algorithm in adaptive PUL-Inter-Slice defender of IS^2 -DriftGuard due to its ability to support partial retraining, which aligns well with the CL

paradigm, avoiding the need to retrain the clustering algorithm from scratch. We leverage the default hyperparameters provided by the `scikit-learn` library [learn developers \(2024\)](#). Specifically, we configure only three parameters: the `random_state`, which is set to a constant value for reproducibility; the number of clusters (`n_clusters`), set to 2 to reflect our binary classification setting; and the `batch_size`, set to 128. All other hyperparameters, such as the initialization method (`init`), number of initializations (`n_init`), maximum number of iterations (`max_iter`), tolerance (`tol`), early stopping criterion (`max_no_improvement`), initial sample size (`init_size`), reassignment ratio (`reassignment_ratio`), verbosity (`verbose`), and label computation (`compute_labels`), were retained at their default settings, ensuring a standard and reproducible clustering process, as summarized in Table 5.4.

Table 5.4: MiniBatch K-Means hyperparameters

Hyperparameter	Value
<code>n_clusters</code>	2
<code>random_state</code>	42
<code>batch_size</code>	128
<code>init</code>	k-means++
<code>n_init</code>	3
<code>max_iter</code>	100
<code>tol</code>	0.0
<code>max_no_improvement</code>	10
<code>init_size</code>	None
<code>reassignment_ratio</code>	0.01
<code>verbose</code>	0
<code>compute_labels</code>	True

5.5.4 Generation of Drifted RSA and TSA Distributions

To assess the robustness of our adaptive PUL-Inter-Slice defender of IS^2 -DriftGuard under adversarial concept drift, we use the drift injector (Section 5.3.1), which leverages the CWGAN-GP trained on labeled RSA and TSA samples (Table 5.1). The trained generator synthesizes drifted variants of these attack patterns by injecting controlled noise into the latent space. Specifically, each component of the latent vector $z = (z_1, z_2, \dots, z_d)$ is independently sampled from a standard normal distribution, $z_i \sim \mathcal{N}(0, 1)$. To simulate drift, we introduce an additive perturbation ϵ ,

also sampled from a Gaussian distribution with configurable mean μ and standard deviation σ , i.e., $\epsilon \sim \mathcal{N}(\mu, \sigma^2)$ [Nawara and Kashef \(2025\)](#). The perturbed latent vector is then defined component-wise as:

$$z' = z + \epsilon \quad (9)$$

By systematically varying μ and σ , we generate incremental shifts in the latent space that correspond to adversarial drift in RSA and TSA traffic. We implement a feature drift scenario in which all dimensions of the latent vector are modified by this additive Gaussian noise (Fig. 5.3)). These parameters are independently adjusted for each generated distribution to enable fine-grained control over the intensity and the nature of the drift.

The generated samples are passed through the generator, inverse-transformed using the StandardScaler fitted during training [scikit-learn developers \(2025\)](#), and rounded to match the structure of real traffic data. Each sample is labeled according to its corresponding attack class (RSA or TSA), resulting in a sequence of synthetically generated, incrementally drifted distributions.

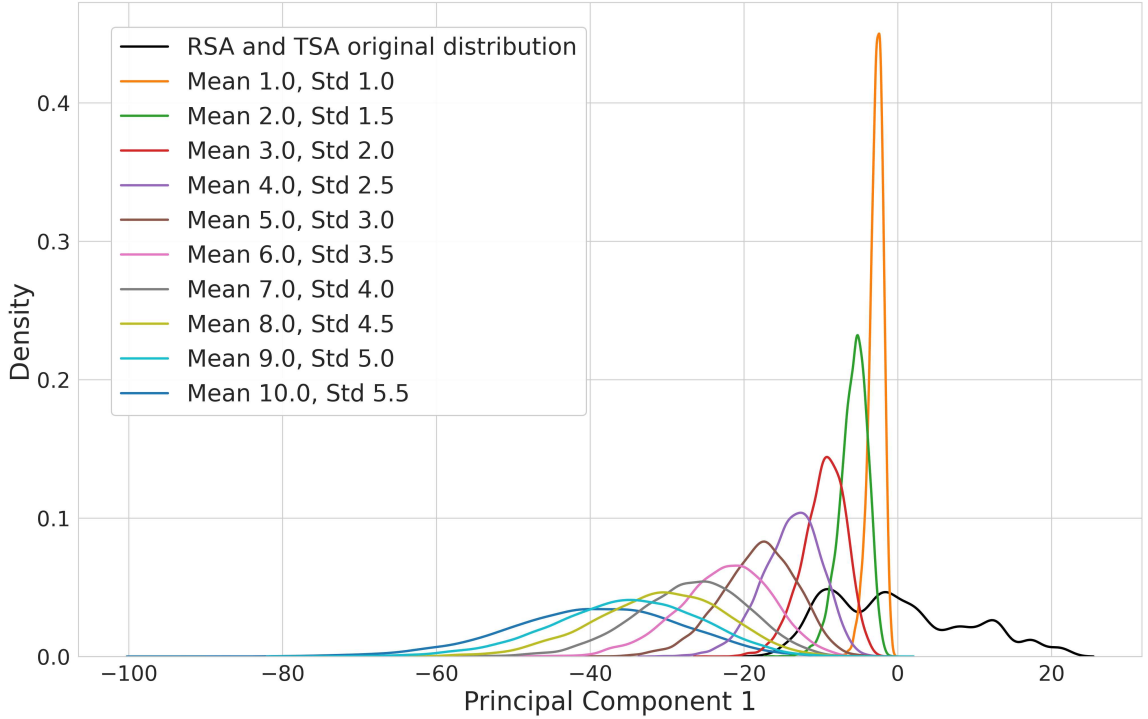


Figure 5.3: Drifted distributions containing RSA and TSA.

Figure 5.3 presents the 1D Kernel Density Estimation (KDEs) of RSA and TSA samples projected onto the first Principal Component (PC1). These KDE plots provide smoothed approximations of the Probability Density Functions along PC1, allowing a visual comparison of distributional shifts over time. The black curve denotes the original joint distribution used to train the CWGAN-GP in the baseline distribution learner (Section 5.3.1), while the colored curves represent synthetically generated, progressively drifted distributions. As the drift parameters increase i.e., the mean and the standard deviation), the distributions exhibit a leftward shift in their mean and a gradual broadening, indicating changes in both central tendency and variance. This visual evolution confirms the presence of incremental concept drift. The overlapping regions and smooth transitions between consecutive distributions further illustrate how adversarial behavior can evolve subtly over time.

5.5.5 Batch Size Selection

In our IS^2 -DriftGuard, the batch size defines the number of samples used for evaluating the adaptive PUL-Inter-slice defender performance, and for monitoring distributional shifts. Smaller batches provide finer detail to capture subtle drift, while larger batches dilute short-term variations within a broader sample. To identify the most suitable batch size, we conduct a sensitivity analysis using the training dataset of the concept drift adapter (Table 5.1), which comprises benign, RSA, and TSA samples. We varied the batch size from 1,000 to 5,000 records and noted the corresponding average F1-score of adaptive PUL-Inter-Slice defender. The results, shown in Figure 5.4, indicate that the highest performance (96.02%) occurs at a batch size of 1,000 records, in contrast to the lower F1-scores recorded at larger batch sizes.

This behavior can be explained by the relationship between the batch size and evaluation granularity. With larger batches, the F1-score captures model performance over a more heterogeneous batch composition, increasing sensitivity to classification errors. This challenge is particularly relevant in the context of NS, where RSA and TSA use legitimate UE traffic, leveraging, for instance, ISS events to launch DDoS attacks against the CP. As a result, attack traffic may closely resemble normal behavior, creating patterns that are sometimes difficult to distinguish from non-compromised UE activity. Consequently, misclassifying even a small subset of samples, such as failing to identify

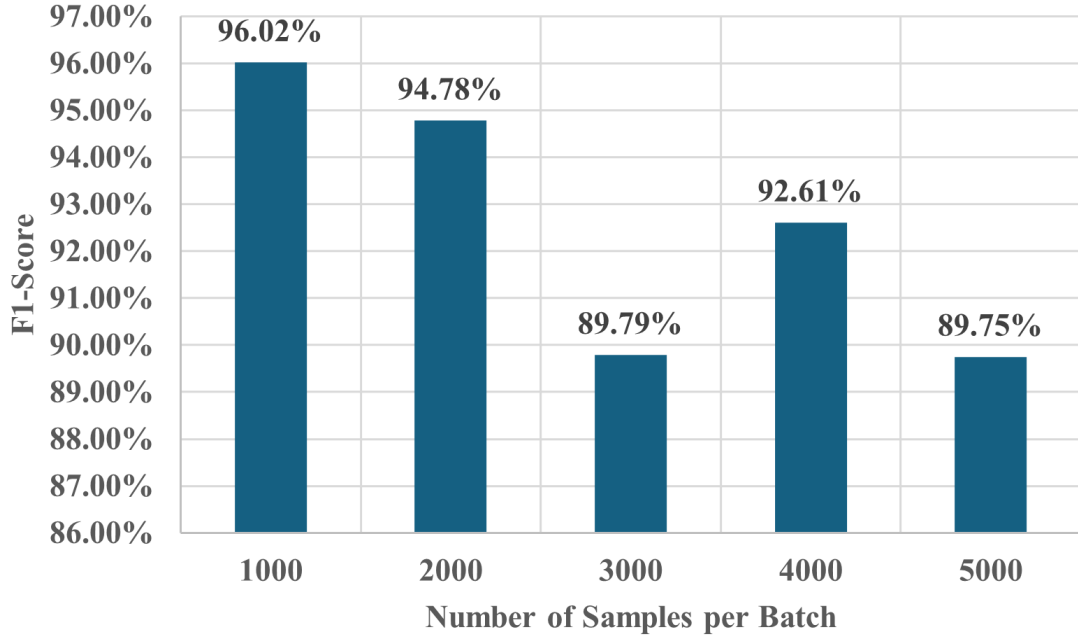


Figure 5.4: Average F1-score of adaptive PUL-Inter-Slice defender across different batch sizes.

a cluster of attack instances, can have a disproportionately large impact on the batch-level F1-score. Furthermore, larger batch sizes yield fewer evaluation intervals, causing each batch to contribute more heavily to the overall average. As a result, performance degradation in a single batch can significantly lower the aggregate metric. Conversely, smaller batch sizes produce more evaluation intervals, each representing a narrower behavioral segment. This finer resolution enhances the model’s ability to detect subtle or transient anomalies, particularly those characteristics of RSA and TSA attacks. Even when individual small batches perform poorly, their influence on the overall F1-score is minimized due to their smaller weight in the averaging process. Moreover, the increased granularity improves the likelihood that localized or short-lived attack patterns are preserved in the latent space and accurately separated during clustering. Based on this analysis, in the following experiments, we use a batch size of 1000 records.

5.5.6 Wasserstein Threshold Selection for Drift Detection

To detect incremental concept drift in the RSA and TSA distributions, IS^2 -DriftGuard establishes a drift detection threshold based on the Wasserstein distance between batch-level latent representations. Initially, this threshold is computed using the steady-state (i.e., stable and non-drifting) Wasserstein dataset (Table 5.1), which consists of 20 batches containing benign, RSA, and TSA traffic. The threshold is computed as the mean Wasserstein distance plus two times the standard deviation Müller et al. (2024), i.e., $\mu + 2\sigma$. This formulation captures normal variation under stable conditions, while deviations beyond this range are interpreted as indications of incremental drift in RSA and/or TSA distributions.

Importantly, the Wasserstein threshold is adaptive: upon drift detection, it is recalculated using the same statistical formulation (i.e., the mean Wasserstein distance plus two standard deviations defined above), applied to the Wasserstein distances observed over the five subsequent batches following the adaptive PUL-Inter-Slice defender model retraining. This empirical recalibration aligns the detection mechanism with the updated model’s perception of non-drift behavior, maintaining sensitivity to incremental drift while minimizing false positives caused by legitimate gradual changes in RSA and TSA traffic distributions.

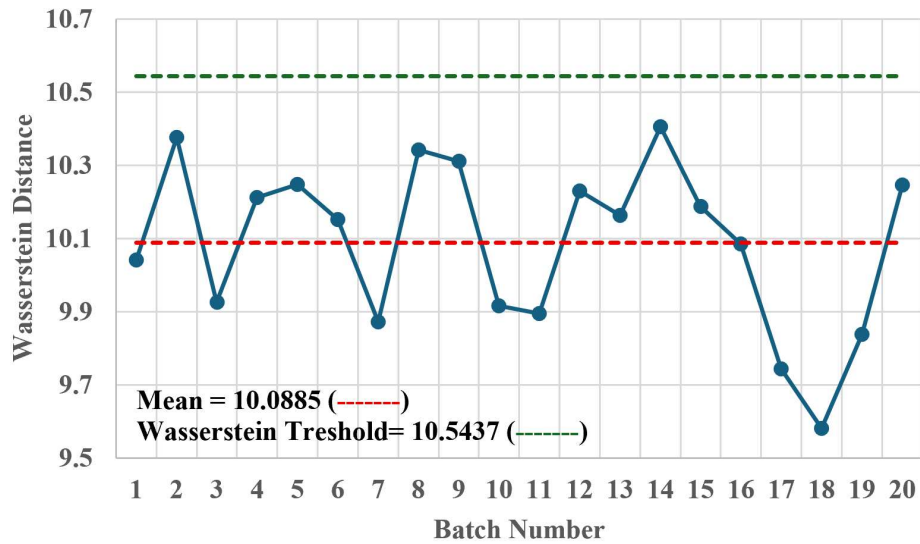


Figure 5.5: Wasserstein Distance per Batch.

Figure 5.5 illustrates the initial computation of the Wasserstein distance for each batch within the steady-state Wasserstein dataset (Table 5.1), before the occurrence of any incremental drift. The red dashed line represents the mean Wasserstein distance across batches (10.0885), while the green dotted line marks the initial threshold, calculated as the mean Wasserstein distance plus two standard deviations (10.5437). This value defines the upper bound of expected variation under stable conditions. As shown, all batch-wise distances remain below the threshold, indicating the absence of drift and validating the use of this dataset to initialize the incremental drift detection. This threshold serves as a reference to identify subsequent changes.

5.5.7 Decay Factor Selection

The decay factor is used to preserve past information in the reference window while incorporating newly observed samples during adaptive PUL-Inter-Slice defender model updates. To identify its most suitable value, we conduct a sensitivity analysis of its impact on the model performance (i.e., measured by the F1-score). Using the drift injector (Section 5.3.1), we generate drifted RSA and TSA samples and merge them with benign data to construct 15 drifted distributions (i.e., decay datasets in Table 5.1). These distributions are first tested in the concept drift adapter with the drift detection mechanism disabled to assess performance degradation under previously unseen conditions. Table 5.5, column F1-Score B.A shows that the F1-score decreases from 96.29% to 80.11%, confirming the presence of concept drift. We then re-enable the drift detection mechanism and re-evaluate the model on the same drifted distributions to analyze the F1-score recovery under different decay values. Figure 5.6 summarizes the results of this sensitivity analysis, where the reported F1-scores represent the average performance across all 15 drifted distributions (Table 5.1).

Among the tested decay values, 0.7 achieved the most effective balance for handling incremental drift. With 70% weight assigned to the stable pre-drift reference and 30% to the new drifted data, it enabled progressive adaptation without overreacting to short-term anomalies, achieving the highest F1-score of 93.18%. In contrast, a smaller decay factor of 0.5, given the incremental nature of concept drift, caused a pronounced performance drop (82.34%). Such a drop in performance is due to placing excessive weight on new samples, which destabilizes the reference window and discards too much historical information, thus leading to the loss of useful prior knowledge. However, a larger

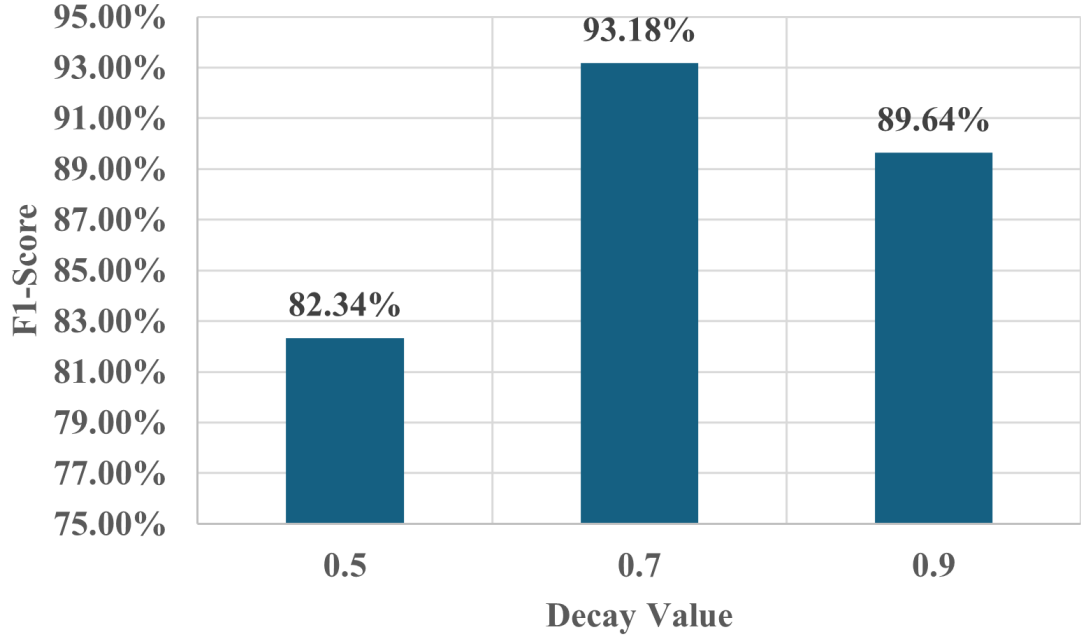


Figure 5.6: Sensitivity analysis of the decay factor.

decay factor of 0.9 preserves more historical information but yields a lower F1-score (89.64%). This outcome suggests that while emphasizing stability, assigning too much weight to past data limits the model’s ability to adjust promptly to new drifted distributions, thereby slowing adaptation and preventing optimal performance recovery. Therefore, selecting a decay value of 0.7 allows the model to learn from new data while retaining past knowledge, helping counter catastrophic forgetting. By preserving a substantial portion of the pre-drift latent space in the reference window, the model maintains continuity with earlier traffic patterns. This prevents abrupt transitions in learned representations and supports stable decision boundaries, even as the input distribution evolves. As such, a decay factor of 0.7 provides a principled trade-off between adaptability and memory retention, making it a robust choice for maintaining long-term performance in adversarial environments.

5.5.8 IS^2 -DriftGuard Performance

To assess the effectiveness of our proposed method, we use PUL-Inter-Slice Defender [Ayala Molina et al. \(2025\)](#) as the baseline for our evaluation. As a first step, both PUL-Inter-Slice Defender and the adaptive PUL-Inter-Slice defender of IS^2 -DriftGuard are trained using the dataset

detailed in Table 5.1, which consists of 90% benign samples and 10% attack samples evenly divided between RSA and TSA. Subsequently, we generate 100 incremental drifted distributions using the drift injector (Section 5.3.1) by fixing the mean at 1 and systematically increasing the standard deviation from 1, in increments of 0.1. We evaluate the performance of PUL-Inter-Slice Defender on these distributions to assess the extent of its performance degradation, as reflected in the F1-score. Based on the results, we select 15 representative drifted distributions (i.e., out of the generated 100 distributions) that exhibit a clear pattern of incremental concept drift. These 15 retained distributions are then introduced sequentially into IS^2 -DriftGuard, allowing us to analyze its response to evolving data conditions and compare its performance against that of [Ayala Molina et al. \(2025\)](#).

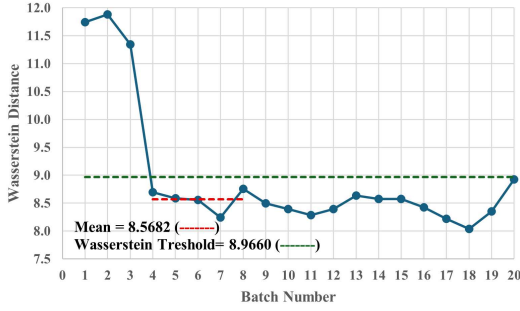
Table 5.5: Comparison of PUL-Inter-Slice Defender [Ayala Molina et al. \(2025\)](#) and IS^2 -DriftGuard under incremental RSA and TSA drift.

Drifted Distribution	PUL-ISD F1-Score	IS^2 -DriftGuard						
		F1-Score B.A.	W.D. B.A.	W.D. Threshold B.A.	C.D. Adaptation	W.D. A.A.	F1-Score A.A.	W.D. Threshold A.A.
1	96.29%	96.29%	7.4424	10.5437	————	————	————	————
2	95.39%	95.39%	7.7521	10.5437	————	————	————	————
3	91.66%	91.66%	9.0481	10.5437	————	————	————	————
4	90.43%	90.43%	9.9059	10.5437	————	————	————	————
5	89.10%	89.10%	11.5365	10.5437	X	8.5682	95.10%	8.9660
6	88.92%	88.92%	11.9363	————	————	6.5658	94.32%	8.9660
7	87.39%	87.39%	12.3884	————	————	7.1232	93.67%	8.9660
8	86.19%	86.19%	12.6762	————	————	7.8171	92.64%	8.9660
9	85.03%	85.03%	13.1338	————	X	9.6170	93.62%	10.2820
10	84.69%	84.69%	13.4629	————	————	9.9411	94.17%	10.2820
11	83.50%	83.50%	13.8972	————	————	10.1769	93.15%	10.2820
12	82.95%	82.95%	14.3081	————	X	10.4640	93.28%	10.7423
13	81.77%	81.77%	14.7259	————	————	8.1682	91.78%	10.7423
14	80.71%	80.71%	15.0628	————	————	9.0738	92.41%	10.7423
15	80.11%	80.11%	15.5204	————	————	9.2244	90.11%	10.7423

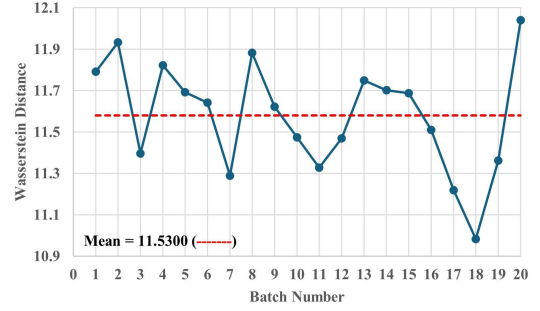
A.A. = After concept drift adaptation; B.A. = Before concept drift adaptation; C.D. = Concept drift; W.D. = Wasserstein distance.

PUL-ISD = PUL-Inter-Slice Defender; W.D. A.A. = Wasserstein distance after concept drift adaptation;

W.D. B.A. = Wasserstein distance before concept drift adaptation.



(a) Drifted distribution during concept drift adaptation.



(b) Drifted distribution without concept drift adaptation.

Figure 5.7: Analysis of drifted distribution number 5.

Table 5.5 (i.e., PUL-ISD column) highlights the progressive degradation observed in the PUL-Inter-Slice Defender performance as it is sequentially evaluated on drifted distributions. This degradation arises from its architecture, which lacks mechanisms to accommodate the gradual evolution of attack behaviors. As a result, it suffers from a decline in its performance as drift severity increases. Thus, we notice a degradation in its performance from an F1-score of 96.29% for the drifted distribution #1 to reach 80.11% for the drifted distribution #15 as the drift level increases.

On the contrary, IS^2 -DriftGuard integrates a dynamically adaptive Wasserstein threshold mechanism, initially set to 10.5437 as described in Section 5.5.6. This threshold enables the model to detect and incorporate knowledge from the first drifted distribution for which the Wasserstein distance exceeds this value. As shown in Table 5.5, the first four drifted distributions yield F1-scores ranging from 96.29% to 90.43% (F1-Score B.A. column) and Wasserstein distances ranging from 7.4424 to 9.9059 (i.e., W.D. B.A. column), both below the initial threshold (i.e., W.D. Threshold column), indicating minimal impact on the model’s predictive capability. However, the fifth drifted distribution surpasses this threshold with a measured Wasserstein distance of 11.5365, signaling the beginning of an incremental distributional shift that degrades the model’s performance.

Figure 5.7a illustrates how our approach processes the fifth drifted distribution. It shows that when three consecutive batches exceed the initial Wasserstein threshold (i.e., W.D. Threshold column in Table 5.5), the system detects that incremental drift is degrading the model performance and triggers adaptation using knowledge extracted from the affected batches (i.e., C.D. Adaptation column in Table 5.5). Specifically, the current reference window is updated using a decay factor

of 0.7 (Section 5.5.7), retaining 70% of the current window and incorporating 30% of the drifted samples. The model is then retrained through a continual learning process, allowing for adaptation to the evolving distribution without requiring a full retraining from scratch. This adaptation process leads to a performance improvement from 89.10% to 95.10% in terms of F1-score (i.e., F1-Score A.A. column in Table 5.5), demonstrating the model’s ability to recover predictive accuracy following a drift. In addition, the Wasserstein threshold is recalculated (Section 5.5.6) based on the new reference window and updated to 8.9660 (i.e., W.D. Threshold A.A. column in Table 5.5), reflecting the refined latent distribution boundaries resulting from the integration of drifted samples. This behavior is contrasted in Figure 5.7b, which shows the same drifted distribution processed without activating the retraining mechanism. While Figure 5.7b exhibits persistent distributional divergence, Figure 5.7a demonstrates that, following retraining, the Wasserstein distances of subsequent batches steadily decrease. This reduction underscores the effectiveness of our custom loss function in shaping a stable and adaptable latent space that enables seamless integration of new distributions without compromising prior knowledge.

Furthermore, the retraining process restores the model’s performance for the initially drifted distribution and enhances its generalization to subsequent ones. As shown in Table 5.5, distributions #6, #7, and #8 before concept drift adaptation yielded F1-scores of 88.92%, 87.39%, and 86.17%, respectively (i.e., F1-Score B.A. column). After the adaptation that is triggered by distribution #5, the model’s performance improved significantly, reaching F1-scores of 94.32%, 93.67%, and 92.64% on these distributions (i.e., F1-Score A.A. column). This improvement is attributed to the updated reference window, the recalculated Wasserstein threshold, and the refined latent space. These results illustrate that CL progressively enhances the generalization of the model, allowing it to assimilate future distributional changes without triggering additional drift detection or requiring complete retraining.

This process continues with drifted distribution #9, for which the Wasserstein distance reaches 9.6170, surpassing the updated threshold of 8.9660 established after the retraining on distribution #5. Consequently, the model undergoes another retraining using the drifted samples from distribution #9, which results in further refinement of the latent space and the calculation of a new Wasserstein threshold, now updated to 10.2820. This new threshold becomes the new baseline for

evaluating subsequent distributions. With each retraining cycle, the model’s adaptability improves. Following this retraining, it accurately predicts distribution #10 with an F1-score of 94.17% and distribution #11 with 93.15% (i.e., F1-Score A.A. column), both of which previously achieved only 84.69% and 83.50% (i.e., F1-Score B.A. column), respectively. This performance gain reflects the successful integration of new knowledge from distribution #9. The cycle of detection and adaptation is repeated whenever the model encounters a drifted distribution exceeding the current threshold, as further demonstrated by distributions #12 in Table 5.5.

5.5.9 Computational Overhead

While IS^2 -DriftGuard demonstrates strong adaptability and resiliency under incremental concept drift, its responsiveness introduces computational costs that may impact real-time deployment. Specifically, once incremental drift is detected based on three consecutive batches exceeding the Wasserstein distance threshold, retraining of the adaptive PUL-Inter-Slice defender is initiated using a CL strategy. This process involves fine-tuning the LSTM-Autoencoder with a custom loss function that integrates MSE, MMD, SIR, and EWC. Additionally, the Fisher information matrix is recalculated to preserve previously learned knowledge. Further, MiniBatch K-means is also retrained. In the current experimental setup, the initial offline adaptive PUL-Inter-Slice defender training requires 309 seconds. However, its retraining requires approximately 137 seconds, and the Fisher matrix computation adds another 75 seconds, resulting in a total retraining time of 212 seconds. Although this retraining overhead is manageable in offline or near real-time scenarios, it may be limiting in high-throughput environments where rapid adaptation is essential. Nonetheless, the concept drift detection time and the time required to classify the data into benign or attack are negligible (i.e., order of microseconds).

5.6 Discussion

The drifted RSA and TSA samples are assumed to represent realistic variants of the original attack classes, simulating how adversaries might adapt DSM strategies in 5G networks. Although

synthetically produced through controlled perturbations in the latent space, these drifted distributions are treated as credible manifestations of evolving threats. This assumption is grounded in the adversarial learning capability of CWGAN-GP, which enables the generator to explore a wide spectrum of class-conditioned variations. Importantly, these drifted samples also model incremental concept drift, as their generation involves small, systematic modifications to the noise parameters, specifically, slight changes in the mean and standard deviation. This simulates gradual, targeted shifts in behavior designed to evade static detection systems. Consequently, the generated samples are not considered hypothetical artifacts, but are integrated into the detection pipeline as valid representations of adversarial drift. This assumption forms the basis of our evaluation methodology, ensuring that the proposed framework is tested against realistic, adaptive attack patterns and enabling a robust assessment of its inference and adaptation capabilities.

Chapter 6

Real-World Deployment Considerations and Limitations

This chapter discusses the practical considerations and limitations associated with deploying the proposed frameworks in real-world 5G environments. It addresses implementation challenges, scalability aspects, and model limitations to provide a comprehensive perspective on their applicability beyond controlled experimental settings.

6.1 Deployment Aspects in Realistic 5G Environments

Inter-slice Defender, PUL-Inter-slice Defender, and DriftGuard utilize 3GPP KPIs and PM counters, collected through the NWDAF [Garcia-Martin et al. \(2023\)](#), to enable data-driven detection of DSM attacks and enhance the resilience of 5G networks . These frameworks can also support automated mitigation actions, such as rate limiting or slice isolation, when integrated with existing security systems. However, real-world deployment is challenged by the distributed nature of NSs across multiple administrative domains, which limits data sharing and global visibility due to policy and privacy constraints. To address these issues, a distributed monitoring approach is required, where each domain contributes locally while preserving autonomy. In this context, federated learning [Lu et al. \(2024\)](#); [Wen et al. \(2023\)](#) represents a promising solution, enabling collaborative and privacy-preserving model training through the sharing of aggregated insights rather than raw data,

thereby ensuring scalability and adaptability in heterogeneous 5G environments.

6.2 Experimental Setup and Deployment Limitations

It is important to note that the deployment analysis presented in this work is based on a single experimental testbed using free5GC [Free5GC \(2023\)](#) and UERANSIM [Aligungr \(2023\)](#). While this setup enables controlled evaluation of the proposed frameworks, it represents a simplified environment compared to real-world 5G deployments. In operational networks, significantly larger numbers of user equipments, heterogeneous infrastructures, and diverse traffic conditions may introduce additional challenges that are not fully captured in this study. This limitation may affect the generalizability of the results and highlights the need for future validation across multiple testbeds and large-scale production environments to ensure the robustness of the proposed approaches.

6.3 Impact of Feature Quality on Model Performance

One important aspect to consider is that the effectiveness of the proposed models depends on the quality and representativeness of the input features, namely the 3GPP KPIs [3GPP TS 28.554 \(2023\)](#) and PM counters [3GPP TS 28.552 \(2023\)](#). In scenarios where these features are noisy, incomplete, or affected by measurement inconsistencies, the learned latent representations may become less discriminative, potentially impacting the model’s ability to accurately distinguish between benign and malicious behavior. Furthermore, as 5G networks evolve, new 3GPP-defined KPIs and PM counters may be introduced, while existing ones may be modified or deprecated. Such changes can alter the relevance and expressiveness of the feature set used in this work, potentially reducing its effectiveness over time. Consequently, maintaining optimal performance may require periodic updates to the selected features and adaptation of the models to incorporate newly defined or updated network metrics.

6.4 Scalability Considerations

Our frameworks training and evaluation was performed on a dataset generated using 92 UEs distributed across four NSs (Section 4.4). While the number of UEs can be considered minimal compared to that of real-world deployment, it is important to note that the proposed frameworks rely on 3GPP KPIs and PM counters that can be collected from the NWDAF Garcia-Martin et al. (2023). The values of these KPIs and PM counters are not necessarily tied to the number of UEs, but are rather impacted by the number of 5G procedures that these UEs trigger. In other words, Inter-Slice Defender, PUL-Inter-Slice Defender, and DriftGuard rely on procedure-level behavioral patterns rather than device-specific traffic, which suggests their potential to scale to a larger number of UEs and more diverse network environments. However, further evaluation under large-scale deployment scenarios is required to fully validate this scalability

Chapter 7

Conclusion and Future Directions

This thesis presented a comprehensive study on enhancing the security and resilience of 5G network slices through the development of advanced anomaly detection frameworks designed to counteract DSM attacks. The research addressed critical challenges associated with the detection of control-plane anomalies in dynamic 5G environments, where data contamination, distributional shifts, and adversarial adaptations can significantly hinder detection reliability.

The work began with the design of Inter-Slice Defender, a deep learning-based solution employing Long Short-Term Memory Autoencoders to detect DSM-induced anomalies using temporal dependencies in 3GPP KPIs and PM counters. This solution demonstrated strong detection capabilities in a controlled 5G testbed, establishing the foundation for further exploration into more realistic and challenging network conditions.

Building upon this foundation, PUL-Inter-Slice Defender was proposed to address scenarios where training datasets are contaminated with mixed benign and malicious samples. By incorporating PUL and clustering-based decision boundaries in the latent space, the model effectively differentiated between normal and attack traffic without requiring exhaustive labeling. Experimental results confirmed its robustness and data efficiency, making it suitable for practical deployment in dynamic and large-scale 5G infrastructures.

Recognizing that 5G networks are subject to continuous evolution and adversarial drift, the thesis introduced IS^2 -DriftGuard, an adaptive detection framework capable of learning from incremental concept drift in DSM attack behaviors. Leveraging a CWGAN-GP to simulate drifted traffic distributions, IS^2 -DriftGuard integrated continual learning with advanced regularization techniques, including MMD, EWC, SIR and MSE, to preserve prior knowledge while adapting to evolving data. The use of Wasserstein distance as a drift metric provided a reliable mechanism for detecting distributional changes and triggering retraining when necessary.

The collective results of these studies validated the robustness, scalability, and adaptability of the proposed solutions across varying attack intensities, contamination levels, and drift scenarios. Through extensive testing on a Free5GC-based testbed, the models consistently achieved high detection performance and maintained stability under evolving traffic conditions. Together, the three proposed frameworks constitute an integrated defense pipeline that advances the state of the art in 5G network slice security, bridging the gap between static detection and adaptive resilience.

Future research could focus on developing a UE location-aware FL framework to strengthen the detection of DSM attacks in 5G networks. Given that DSM attacks are closely linked to the mobility and slice-switching behavior of UE, integrating spatial and temporal context into anomaly detection models can significantly enhance their ability to capture evolving attack dynamics.

A promising approach involves leveraging FL to train distributed detection models across multiple edge nodes or base stations without sharing raw UE data. This decentralized paradigm not only ensures privacy preservation but also reduces the communication overhead associated with centralized training. By incorporating location-dependent features such as UE trajectory, handover frequency, and slice association patterns, the system can learn localized behaviors that collectively contribute to a global understanding of DSM attack propagation.

Moreover, this line of research can enable context-aware and adaptive security mechanisms, where each participating node dynamically refines its local model based on regional traffic characteristics and collaboratively contributes to global model updates. Such an architecture would allow for real-time, privacy-preserving, and scalable anomaly detection, effectively correlating UE movement, signaling behavior, and slice interactions.

Ultimately, integrating UE location awareness with FL represents a crucial step toward developing intelligent, distributed, and resilient defense mechanisms capable of mitigating DSM attacks across large-scale 5G and B5G infrastructures.

The research carried out throughout this Ph.D. program has resulted in multiple contributions, which have been either published/submitted to top-tier venues, as presented in Table 7.1

Table 7.1: Publications and Contributions during the Ph.D. Program

Publication Title	Citation
<i>Inter-slice Defender: An Anomaly Detection Solution for Distributed Slice Mobility Attacks</i>	R. M. A. Molina, N. Wehbe, H. A. Alameddine, M. Pourzandi, and C. Assi, “Inter-slice Defender: An Anomaly Detection Solution for Distributed Slice Mobility Attacks,” in <i>2024 IFIP Networking Conference (IFIP Networking)</i> . IEEE, 2024, pp. 432–440.
<i>PUL-Inter-slice Defender: An Anomaly Detection Solution for Distributed Slice Mobility Attacks</i>	R. Misael Ayala Molina, H. Assem Alameddine, M. Pourzandi, and C. Assi, “PUL-Inter-slice Defender: An Anomaly Detection Solution for Distributed Slice Mobility Attacks,” <i>IEEE Transactions on Information Forensics & Security</i> , accepted with mandatory minor revision to publication in 2025.
<i>IS2-DriftGuard: Concept Drift-Based Adaptive Detection of Distributed Slice Mobility Attacks</i>	R. Misael Ayala Molina, H. Assem Alameddine, M. Pourzandi, and C. Assi, “IS2-DriftGuard: Concept Drift-Based Adaptive Detection of Distributed Slice Mobility Attacks,” <i>IEEE/ACM Transactions on Networking</i> , submitted, 2025.

Additional contributions derived from this Ph.D. research, published in leading venues focusing on ransomware detection, are summarized in Table 7.2.

Table 7.2: Additional Contributions

Publication Title	Citation
<i>On Ransomware Family Attribution Using Pre-Attack Paranoia Activities</i>	R. M. A. Molina, S. Torabi, K. Sardedine, E. Bou-Harb, N. Bouguila, and C. Assi, “On Ransomware Family Attribution Using Pre-Attack Paranoia Activities,” <i>IEEE Transactions on Network and Service Management</i> , vol. 19, no. 1, pp. 19–36, 2021.
<i>RPM: Ransomware Prevention and Mitigation Using Operating Systems’ Sensing Tactics</i>	R. M. A. Molina, E. Bou-Harb, S. Torabi, and C. Assi, “RPM: Ransomware Prevention and Mitigation Using Operating Systems’ Sensing Tactics,” in <i>ICC 2023–IEEE International Conference on Communications</i> . IEEE, 2023, pp. 1–6.

References

- 3GPP. (2023). *The 5g standard*. <https://www.3gpp.org/>. ([Online; accessed Aug-2023])
- 3GPP TS 23.501. (2023). *System architecture for the 5G System(5GS): TS 23.501 V18.4.0*.
- 3GPP TS 23.502. (2023). *Procedures for the 5G System (5GS): TS 23.502 V18.4.0*.
- 3GPP TS 28.552. (2023). *Management and orchestration; 5G performance measurements: TS 28.552 V18.3.0*.
- 3GPP TS 28.554. (2023). *Management and orchestration; 5G end to end Key Performance Indicators (KPI): TS 28.554 V18.2.0*.
- Abbas, K., Khan, T. A., Afaq, M., & Song, W.-C. (2021). Network slice lifecycle management for 5g mobile networks: An intent-based networking approach. *IEEE Access*, 9, 80128–80146.
- Agyemang, E. F. (2024). Anomaly detection using unsupervised machine learning algorithms: A simulation study. *Scientific African*, 26, e02386.
- Aich, A. (2021). Elastic weight consolidation (ewc): Nuts and bolts. *arXiv preprint arXiv:2105.04093*.
- Akhtarshenas, A., Vahedifar, M. A., Ayoobi, N., Maham, B., Alizadeh, T., Ebrahimi, S., & López-Pérez, D. (2024). Federated learning: A cutting-edge survey of the latest advancements and applications. *Computer Communications*, 228, 107964.
- Aleixo, E. L., Colonna, J. G., Cristo, M., & Fernandes, E. (2023). Catastrophic forgetting in deep learning: a comprehensive taxonomy. *arXiv preprint arXiv:2312.10549*.
- Ali, A. F., & Siyad, A. A. (2024). Comprehensive survey on network slicing in 5g. In *International conference on artificial intelligence of things* (pp. 14–29).
- Aligungr. (2023). *Ueransim*. <https://github.com/aligungr/UERANSIM>. ([Online;

accessed Jul-2025])

- Al-Shareeda, M. A., Manickam, S., & Ali, M. (2023). Ddos attacks detection using machine learning and deep learning techniques: Analysis and comparison. *Bulletin of Electrical Engineering and Informatics*, 12(2), 930–939.
- Altowaijri, S. M., & Ayari, M. (2025). The synergistic impact of 5g on cloud-to-edge computing and the evolution of digital applications. *Mathematics*, 13(16), 2634.
- Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein generative adversarial networks. In *International conference on machine learning* (pp. 214–223).
- Awotunde, J. B., Ayo, F. E., Panigrahi, R., Garg, A., Bhoi, A. K., & Barsocchi, P. (2023). A multi-level random forest model-based intrusion detection using fuzzy inference system for internet of things networks. *International Journal of Computational Intelligence Systems*, 16(1), 31.
- Ayala Molina, R. M., Assem Alameddine, H., Pourzandi, M., & Assi, C. (2025). Pul-inter-slice defender: An anomaly detection solution for distributed slice mobility attacks. *arXiv e-prints*, arXiv–2510.
- Ayala Molina, R. M., Wehbe, N., Alameddine, H. A., Pourzandi, M., & Assi, C. (2024). Inter-slice defender: An anomaly detection solution for distributed slice mobility attacks. In *2024 ifip networking conference (ifip networking)* (pp. 432–440).
- Bekker, J., & Davis, J. (2020). Learning from positive and unlabeled data: A survey. *Machine Learning*, 109(4), 719–760.
- Beshah, Y. K., Abebe, S. L., & Melaku, H. M. (2024). Drift adaptive online ddos attack detection framework for iot system. *Electronics*, 13(6), 1004.
- Bisht, H., Patra, M., & Kumar, S. (2023). Detection and localization of ddos attack during inter-slice handover in 5g network slicing. In *2023 ieee 20th consumer communications & networking conference (ccnc)* (pp. 798–803).
- Chahbar, M., Diaz, G., Dandoush, A., Cérin, C., & Ghoumid, K. (2020). A comprehensive survey on the e2e 5g network slicing model. *IEEE Transactions on Network and Service Management*, 18(1), 49–62.
- Chaouki, H., Iqdour, R., & Boulouird, M. (2024). Enhancing 5g networks with edge computing: An overview study. In *Itm web of conferences* (Vol. 69, p. 04010).

- Chu, R., Jin, P., Qiao, H., & Feng, Q. (2024). Intrusion detection in the iot data streams using concept drift localization. *AIMS mathematics*, 9(1), 1535–1561.
- Cybersecurity and Infrastructure Security Agency (CISA). (2024, August). *5g network slicing: Security considerations for design, deployment, and maintenance* (Tech. Rep.). Washington, D.C.: U.S. Department of Homeland Security. Retrieved 2025-10-28, from https://www.cisa.gov/sites/default/files/2024-08/ESF_5G_NETWORK_SLICING-SECURITY_CONSIDERATIONS_FOR_DESIGN%2CDEPLOYMENT%2CAND_MAINTENANCE_FINAL_508.pdf
- Dangi, R., Jadhav, A., Choudhary, G., Dragoni, N., Mishra, M. K., & Lalwani, P. (2022). MI-based 5g network slicing security: A comprehensive survey. *Future Internet*, 14(4), 116.
- De Alwis, C., Porambage, P., Dev, K., Gadekallu, T. R., & Liyanage, M. (2023). A survey on network slicing security: Attacks, challenges, solutions and research directions. *IEEE Communications Surveys & Tutorials*.
- De Alwis, C., Porambage, P., Dev, K., Gadekallu, T. R., & Liyanage, M. (2024). A survey on network slicing security: Attacks, challenges, solutions and research directions. *IEEE Communications Surveys & Tutorials*, 26(1), 534–557.
- Dias, J., Pinto, P., Santos, R., & Malta, S. (2025). 5g network slicing: Security challenges, attack vectors, and mitigation approaches. *Sensors*, 25(13), 3940.
- Dilworth, R., & Gudla, C. (2024). Harnessing pu learning for enhanced cloud-based ddos detection: A comparative analysis. *arXiv preprint arXiv:2410.18380*.
- Dunmore, A., Jang-Jaccard, J., Sabrina, F., & Kwak, J. (2023). A comprehensive survey of generative adversarial networks (gans) in cybersecurity intrusion detection. *IEEE Access*, 11, 76071–76094.
- Fan, Z., Yao, Y., Du, Y., & Du, X. (2023). Self-paced and reweighting pu learning for imbalanced malicious traffic detection. In *Globecom 2023-2023 ieee global communications conference* (pp. 6018–6023).
- Farzaneh, B., Shahriar, N., Al Muktedir, A. H., Towhid, M. S., & Khosravani, M. S. (2024). Dtl-5g: Deep transfer learning-based ddos attack detection in 5g and beyond networks. *Computer Communications*, 228, 107927.

- Franceschi, L., Donini, M., Perrone, V., Klein, A., Archambeau, C., Seeger, M., ... Frasconi, P. (2024). Hyperparameter optimization in machine learning. *arXiv preprint arXiv:2410.22854*.
- Free5GC. (2023). *Free5gc*. <https://free5gc.org/>. ([Online; accessed Jul-2025])
- Gao, R., Liu, F., Zhang, J., Han, B., Liu, T., Niu, G., & Sugiyama, M. (2021). Maximum mean discrepancy test is aware of adversarial attacks. In *International conference on machine learning* (pp. 3564–3575).
- Gao, S., Lin, R., Fu, Y., Li, H., & Cao, J. (2024). Security threats, requirements and recommendations on creating 5g network slicing system: A survey. *Electronics*, 13(10), 1860.
- Garcia-Martin, M. A., Gramaglia, M., & Serrano, P. (2023). Network automation and data analytics in 3gpp 5g systems. *IEEE Network*, 1-1. doi: 10.1109/MNET.2023.3321524
- Greenacre, M., Groenen, P. J., Hastie, T., d’Enza, A. I., Markos, A., & Tuzhilina, E. (2022). Principal component analysis. *Nature Reviews Methods Primers*, 2(1), 100.
- Gudepu, V., Chirumamilla, B., Chintapalli, V. R., Castoldi, P., Valcarengi, L., Tamma, B. R., ... Kondepu, K. (2024). Gan-based drift and anomaly detection for open radio access networks. In *2024 ieee 25th international conference on high performance switching and routing (hpsr)* (pp. 124–129).
- Gudepu, V., Chirumamilla, B., Chintapalli, V. R., Castoldi, P., Valcarengi, L., Tamma, B. R., & Kondepu, K. (2025). Gen-drift: Generative ai-driven drift handling for beyond 5g networks. *Computer Networks*, 263, 111237.
- Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. C. (2017). Improved training of wasserstein gans. *Advances in neural information processing systems*, 30.
- Hinder, F., Vaquet, V., & Hammer, B. (2024). One or two things we know about concept drift—a survey on monitoring in evolving environments. part a: detecting concept drift. *Frontiers in Artificial Intelligence*, 7, 1330257.
- Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning* (pp. 448–456).
- Jain, M., Kaur, G., & Saxena, V. (2022). A k-means clustering and svm based hybrid concept drift detection technique for network anomaly detection. *Expert Systems with Applications*, 193,

116510.

- Khan, M. S., Farzaneh, B., Shahriar, N., Saha, N., & Boutaba, R. (2022). Slicesecond: Impact and detection of dos/ddos attacks on 5g network slices. In *2022 ieee future networks world forum (fnwf)* (pp. 639–642).
- Kim, J., El-Khamy, M., & Lee, J. (2017). Residual lstm: Design of a deep recurrent architecture for distant speech recognition. *arXiv preprint arXiv:1701.03360*.
- Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., ... others (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13), 3521–3526.
- Kuadey, N. A. E., Maale, G. T., Kwantwi, T., Sun, G., & Liu, G. (2021). Deepsecure: Detection of distributed denial of service attacks on 5g network slicing—deep learning approach. *IEEE Wireless Communications Letters*, 11(3), 488–492.
- learn developers, S. (2024). *sklearn.cluster.minibatchkmeans*. <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.MinibatchKMeans.html>. (Accessed: 2025-06-23)
- Long, G., & Zhang, Z. (2024). Punet: A semi-supervised anomaly detection model for network anomaly detection based on positive unlabeled data. *Computers, Materials & Continua*, 81(1).
- Lu, Z., Pan, H., Dai, Y., Si, X., & Zhang, Y. (2024). Federated learning with non-iid data: A survey. *IEEE Internet of Things Journal*, 11(11), 19188–19209.
- Lv, S., Liu, Y., Liu, Z., Chao, W., Wu, C., & Wang, B. (2020). Intrusion detection based on non-negative positive-unlabeled learning. In *2020 ieee 9th data driven control and learning systems conference (ddcls)* (pp. 1015–1020).
- Manias, D. M., Chouman, A., & Shami, A. (2022). A model drift detection and adaptation framework for 5g core networks. In *2022 ieee international mediterranean conference on communications and networking (meditcom)* (pp. 197–202).
- Min-Gyu, K., & Kim, H. (2025). Ensemble encoder-based attack traffic classification for secure 5g slicing networks. *Computer Modeling in Engineering & Sciences*, 143(2), 2391.
- Mirza, M., & Osindero, S. (2014). Conditional generative adversarial nets. *arXiv preprint*

arXiv:1411.1784.

- Molina, R. M. A., Wehbe, N., Alameddine, H. A., Pourzandi, M., & Assi, C. (2024). Inter-slice defender: An anomaly detection solution for distributed slice mobility attacks. In *2024 ifip networking conference (ifip networking)* (pp. 432–440).
- Müller, R., Abdelaal, M., & Stjelja, D. (2024). Open-source drift detection tools in action: Insights from two use cases. In *International conference on big data analytics and knowledge discovery* (pp. 346–352).
- Nawara, D., & Kashef, R. (2025). Penalized gans with latent perturbation for robust shilling attack generation in recommender systems. *Discover Computing*, 28(1), 1–24.
- Neshenko, N., Bou-Harb, E., Crichigno, J., Kaddoum, G., & Ghani, N. (2019). Demystifying iot security: An exhaustive survey on iot vulnerabilities and a first empirical look on internet-scale iot exploitations. *IEEE Communications Surveys & Tutorials*, 21(3), 2702–2733.
- Olimid, R. F., & Nencioni, G. (2020). 5g network slicing: A security overview. *IEEE Access*, 8, 99999–100009.
- OpenStack. (2023). *An openinfra foundation project*. <https://www.openstack.org/>. ([Online; accessed Apr-2023])
- Papič, A., Kononenko, I., & Bosnić, Z. (2023). Conditional generative positive and unlabeled learning. *Expert Systems with Applications*, 224, 120046.
- Said Elsayed, M., Le-Khac, N.-A., Dev, S., & Jurcut, A. D. (2020). Network anomaly detection using lstm based autoencoder. In *Proceedings of the 16th acm symposium on qos and security for wireless and mobile networks* (pp. 37–45).
- Sajjad, M. M., Bernardos, C. J., Jayalath, D., & Tian, Y.-C. (2022). Inter-slice mobility management in 5g: motivations, standard principles, challenges, and research directions. *IEEE Communications Standards Magazine*, 6(1), 93–100.
- Salahdine, F., Han, T., & Zhang, N. (2022). Security in 5g and beyond: Recent advances and future challenges. *Security and Privacy*, 6(4), e271.
- Sathi, V. N., & Murthy, C. S. R. (2020). Distributed slice mobility attack: A novel targeted attack against network slices of 5g networks. *IEEE Networking Letters*, 3(1), 5–9.
- Sathi, V. N., & Murthy, C. S. R. (2021). Dsm attack resistant slice selection in 5g. *IEEE Wireless*

- Communications Letters*, 10(7), 1469–1473.
- Sathi, V. N., Srinivasan, M., Thiruvassagam, P. K., & Chebiyyam, S. R. M. (2018). A novel protocol for securing network slice component association and slice isolation in 5g networks. In *Proceedings of the 21st acm international conference on modeling, analysis and simulation of wireless and mobile systems* (pp. 249–253).
- Sattar, D., & Matrawy, A. (2019). Towards secure slicing: Using slice isolation to mitigate ddos attacks on 5g core network slices. In *2019 ieee conference on communications and network security (cns)* (pp. 82–90).
- scikit-learn developers. (2024). *Kmeans*. <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html>. ([Online; accessed Aug-2024])
- scikit-learn developers. (2025). *StandardScaler — scikit-learn 1.7.0 documentation*. Retrieved from <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html> (Accessed: 2025-06-22)
- Sethi, T. S., & Kantardzic, M. (2018). Handling adversarial concept drift in streaming data. *Expert systems with applications*, 97, 18–40.
- Shen, J., Qu, Y., Zhang, W., & Yu, Y. (2018). Wasserstein distance guided representation learning for domain adaptation. In *Proceedings of the aaai conference on artificial intelligence* (Vol. 32).
- Singh, V. P., Singh, M. P., Hegde, S., & Gupta, M. (2024). Security in 5g network slices: Concerns and opportunities. *IEEE Access*, 12, 52727–52744.
- Thantharate, A., Paropkari, R., Walunj, V., Beard, C., & Kankariya, P. (2020). Secure5g: A deep learning framework towards a secure network slicing in 5g and beyond. In *2020 10th annual computing and communication workshop and conference (ccwc)* (pp. 0852–0857).
- TSHARK.DEV. (2023). *Tshark.dev*. <https://tshark.dev/>. ([Online; accessed Jul-2023])
- van de Ven, G. M., Soures, N., & Kudithipudi, D. (2024). Continual learning and catastrophic forgetting. *arXiv preprint arXiv:2403.05175*.
- Venkatapathy, S., Srinivasan, T., Lee, O.-S., Jayaraman, R., Jo, H.-G., & Ra, I.-H. (2024). Slice-aware 5g network orchestration framework based on dual-slice isolation and management

- strategy (d-sims). *Scientific Reports*, 14(1), 18623.
- Wahyuningrum, T., Khomsah, S., Suyanto, S., Meliana, S., Yunanto, P. E., & Al Maki, W. F. (2021). Improving clustering method performance using k-means, mini batch k-means, birch and spectral. In *2021 4th international seminar on research of information technology and intelligent systems (isriti)* (pp. 206–210).
- Wang, L., Zhang, X., Su, H., & Zhu, J. (2024). A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- Wen, J., Zhang, Z., Lan, Y., Cui, Z., Cai, J., & Zhang, W. (2023). A survey on federated learning: challenges and applications. *International Journal of Machine Learning and Cybernetics*, 14(2), 513–535.
- Wijethilaka, S., & Liyanage, M. (2021). Survey on network slicing for internet of things realization in 5g networks. *IEEE Communications Surveys & Tutorials*, 23(2), 957–987.
- Wong, T.-T., & Yeh, P.-Y. (2020). Reliable accuracy estimates from k-fold cross validation. *IEEE Transactions on Knowledge and Data Engineering*, 32(8), 1586-1594. doi: 10.1109/TKDE.2019.2912815
- Wu, X., Yu, W., Zhai, X., & Liu, N. (2025). Self-regularization with latent space explanations for controllable llm-based classification. *arXiv preprint arXiv:2502.14133*.
- Zhang, S. (2019). An overview of network slicing for 5g. *IEEE Wireless Communications*, 26(3), 111–117.