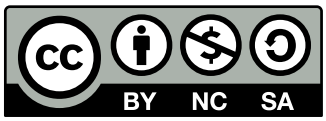
Aalto University
School of Science

Master's Programme in Master's Programme in Security and Cloud Computing

AI/ML specific Threat Modeling in Mobile Networks

Vipul Kumar

Master's Thesis
2024



Aalto University
School of Science

Author Vipul Kumar

Title AI/ML specific Threat Modeling in Mobile Networks

Degree programme Master's Programme in Security and Cloud Computing

Major Security and Cloud Computing

Supervisor Dr. Lachlan Gunn (Aalto University), Prof. Sokratis Katsikas (NTNU)

Advisor Esa Metsala (MSc)

Collaborative partner Nokia

Date 12 November 2024 **Number of pages** 53+1 **Language** English

Abstract

The incorporation of AI and ML into RAN is enhancing mobile networks with better optimizations and automation. However, using AI/ML introduces new security vulnerabilities unique to them and cannot be addressed by traditional security evaluation frameworks. This thesis addresses the gap by proposing a tailored threat modeling framework for security evaluation of ML based features in RAN. The threat modeling framework is built upon established guidelines from OWASP, STRIDE and LINDDUN and incorporates an attack library based on NIST but redefined in the context of RAN.

This framework is designed to evaluate features based on predictive ML algorithms and deployed in a physical base station environment. It provides a structured approach to identify, categorize and mitigate the security risks. Moreover, the framework's design allows for future expansion to include generative ML algorithms and cloud based RAN deployments.

This research fills a critical gap in the literature by extending the AI/ML security evaluation to the unique requirements of RAN, contributing as a valuable resource for security evaluation of AI/ML integration in the next generation mobile networks.

Keywords RAN, AI, ML, Security, Mobile Network, Threat Modeling

Preface

I would like to thank everyone who believed in me and supported me in staying mentally strong and determined throughout my studies, leading to the completion of this thesis, which is a significant part of it.
I would also like to extend my thanks to my supervisor, Dr. Lachlan J. Gunn, for the immense support and feedbacks that he has given me over the past few months. My special thanks also go to Prof. Sokratis Katsikas for supervising my thesis from NTNU. I am also grateful to my advisor at Nokia, Esa Metsala for the help in understanding the basic concepts and guiding the progress of this thesis.

Espoo, 12 November 2024

Vipul Kumar

With the support of the
Erasmus+ Programme
of the European Union

Contents

Acronyms

3GPP 3rd Generation Partnership Project. 8, 11, 12, 20, 24, 48

AI Artificial Intelligence. 3, 8–10

ATLAS Adversarial Threat Landscape for Artificial-Intelligence Systems. 18

ATT&CK Advantageous Tactics, Technologies, and Common Knowledge. 18

CAPEC Common Attack Pattern Enumeration. 18

DP Differential Privacy. 36

EPC Evolved Packet Core. 10, 11

IoT Internet of Things. 11

LTE Long-Term Evolution. 10, 11

ML Machine Learning. 3, 8–13, 17, 19–21, 26, 31, 34, 36, 37, 39, 47–49

NIST National Institute of Standards and Technology. 3, 17, 18, 21, 26, 28, 33, 47, 48

O-RAN Open RAN. 8, 47

OWASP Open Worldwide Application Security Project. 3, 9, 16, 18, 21, 47, 49

RAN Radio Access Networks. 3, 8, 9, 11, 13, 19–21, 24–27, 38, 39, 44–49

ReLU Rectified Linear Unit. 35

RONI Reject On Negative Impact. 35

SMT satisfability Modulo Theories. 35

Glossary

LINDDUN Threat modeling focusing on Privacy. 3, 15–17, 21, 49

STRIDE Threat modeling method developed by Microsoft. 3, 9, 15, 17, 18, 21, 47

1 Introduction

Artificial Intelligence (AI) and Machine Learning (ML) are rapidly advancing fields that aim to mimic human intelligence. They enable systems to learn from data and make decisions without explicit programming instructions. AI is the broader concept of creating machines which can perform tasks that would ordinarily require human intelligence and ML is the sub-field of AI that focuses on the development of the algorithms that enable the machines to learn from and make predictions based on the data. Over the last few years, AI/ML technologies have found widespread applications across various domains, including healthcare, transport and telecommunications. This widespread use leads to the necessity of a comprehensive threat assessment process for these applications which takes into consideration the new vulnerabilities and risks that emerge due to the use of AI/ML.

Radio Access Networks (RAN) or mobile networks are critical infrastructure for any country or organization. Today, everyday life is driven by internet use, and a major part of internet access is through mobile networks. Other critical infrastructure, even if they have their own IT systems, can be impacted by disruptions in mobile networks. This makes mobile networks even more important to be protected from any attacks.

The responsibility for securing this infrastructure is twofold: infrastructure providers must develop secure systems and products, while network operators must manage and enforce security policies. The design and architecture of RAN are largely guided by bodies such as the 3rd Generation Partnership Project (3GPP) and the Open RAN (O-RAN) Alliance, in collaboration with various research organizations.

1.1 Motivation

There exist many resources and measures to make the classical RAN secure. Extensive research investigation is also available about the security risks affecting AI/ML systems in general. With the advancement of mobile networks incorporating more ML based features in the RAN, there is a need to reassess the security measures. Despite the benefits, the integration of AI/ML into RAN introduces new security vulnerabilities. The traditional RAN security models are primarily focused on hardware and network-centric threats only and thus leave a gap in addressing the threats and risks associated specifically with the use of AI/ML systems. The motivation for this thesis stems from the critical gap in existing literature and security practices that fail to address the specific vulnerabilities of AI/ML in RAN. Developing a tailored threat modeling framework is essential to ensure the secure deployment of AI/ML features in current and future mobile network infrastructures.

1.2 Contribution

The primary goal of this thesis is to provide a threat modeling framework to perform a security assessment of the AI/ML-based features in RAN. It adopts a multi-faceted approach to threat modeling, incorporating different perspectives and useful guidelines

from widely used threat modeling frameworks such as Open Worldwide Application Security Project (OWASP) and STRIDE.

The proposed threat modeling framework will provide a detailed framework for evaluating current and future ML-based features in RAN. This new threat modeling framework will enable the development team and other stakeholders, not necessarily security experts, to easily perform a threat assessment for an ML based feature in RAN. The framework also maps the existing AI/ML attacks into the context of RAN to create a RAN specific attack library. The process will involve categorizing the assets and the attackers, identifying the capabilities of the attacker and evaluating the possibility of attacks in RAN context.

Studying how to implement threat modeling in practice for AI/ML use in RAN is also relevant from a research point of view. There have been many studies around the security of AI and ML models, many with a conclusive evaluation and some with an open question directing towards a need for more advanced research and study. Although there are numerous studies on the security of AI and ML models, few have focused specifically on their application in mobile networks. This research also fills the gap by not only addressing the current threat modeling need for AI/ML application in RAN but also by documenting a flexible and scalable method for conducting similar assessments in future.

2 Background

2.1 AI and ML

AI can be described as the ability of computer systems or machines to mimic intelligent human-like behavior that allows them to act and learn autonomously [1]. It is often used to refer different but interrelated sub-fields such as Machine Learning, Deep Learning, Natural Language Processing and Robotics. AI applications are categorized into two broad categories - generative and predictive. Generative AI is used to create new content such as text, images or videos. Predictive AI is a more traditional approach which focuses on prediction, classification and regression use cases [2].

ML can be described as a subset of AI which focuses on learning complex patterns from existing data and use these patterns to make predictions on unseen data [3]. Traditional algorithms take the input data, process them though a known or existing pattern and then produce an output. Whereas, a machine learning algorithm takes the existing input and output data, recognizes the pattern and then use this pattern to predict outputs on new data.

2.2 Mobile Networks

Mobile networks for consumers provide connectivity for smartphones, tablets, laptops and other communication devices. They have evolved through the past decades, transforming from just providing an infrastructure for phone calls to being the backbone of mobile broadband. The introduction of 5G has taken traditional mobile broadband to the extreme in terms of data rates, capacity and availability. This has opened up more application areas as well as industry interest in the advanced use of 5G networks. 5G being accepted as the new standard for radio technology, the use of cloud technology and virtualization in wireless networks and the rapid increase in the use of AI and ML as three major converging inflection points is a good recipe for hyper success [4].

2.2.1 5G and RAN Architecture

The next step in the evolution of mobile networks after the 4G system was Long-Term Evolution (LTE) Advanced. The LTE core network, came with Evolved Packet Core (EPC) which replaced the traditional circuit-switched core network used in earlier generations like 2G and 3G. EPC is a framework for providing converged voice and data services on a LTE network. The core network manages and routes data traffic between mobile devices and the broader internet, ensuring high-speed transmission and efficient, seamless mobility and connectivity. The core network is responsible for handling and routing data and control signals between the mobile device and external networks.

5G has been developed and deployed in two main forms. One of them uses LTE as the connection anchor, i.e., using the existing LTE core for the new network infrastructure and functionality. The second was the standalone 5G radio with the new

5G core [5] . This standalone architecture enables new functionality brought by the 5G core, such as network slicing, which creates multiple virtual networks on top of a shared physical network to provide greater flexibility in the use and allocation of resources, and service-based architecture, among others.

Figure 1 represents the difference between the LTE anchored 5G network and a standalone 5G network. 5G base station (5G BTS) (known as eNodeB in 4G), or base station is the network location which provides wireless signal to the user equipment and connects back to the core network.

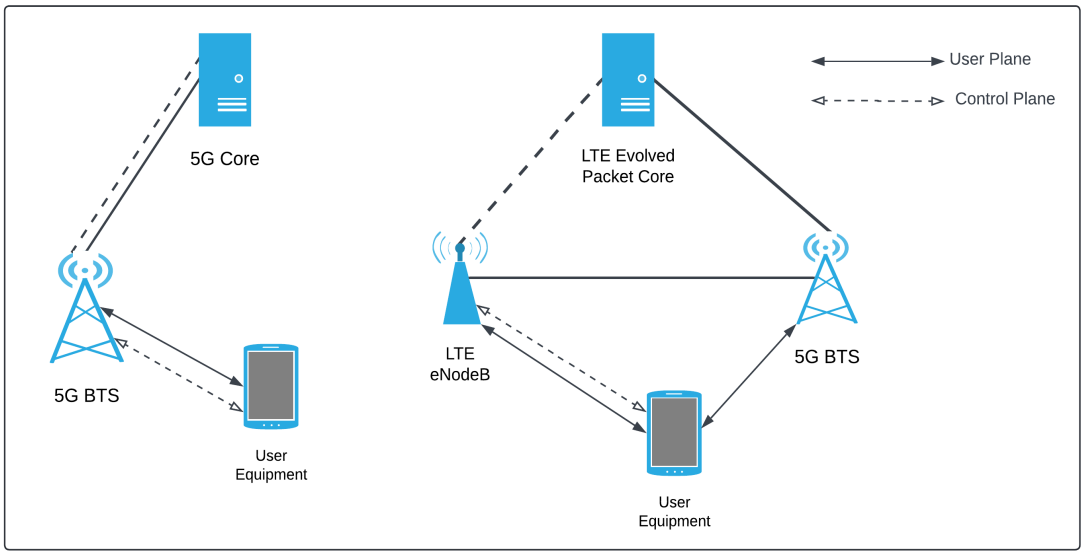


Figure 1: 5G network architectures

The concept of network virtualization is based on the software-based representation of hardware and software resources, considering both data-plane and control-plane functions [6]. The design, development, and management of network components leverage software programmability features, resulting in increased network flexibility and reduced development time and maintenance costs.

Virtualization of the core network allows the service providers to expand into new services such as cellular Internet of Things (IoT) and more enterprise-specific domains. This did not work well with EPC in 4G-LTE generation as it was not a part of EPC specifications. This was addressed in the standalone 5G core with support for Network Function Virtualization and Software Defined Networking, and the 5G core was designed to be deployed in a cloud-native way.

2.2.2 RAN Intelligence

ML based features can help in making mobile networks more efficient. The push towards virtualization and cloud-native deployments has opened up even more areas of ML integration as these can be deployed easily in cloud based environments compared to physical RAN environment. Two of the use cases proposed by 3GPP are [7]:

- **Network Energy Saving**: A cell activation/deactivation scheme where the cell may be switched off depending on the predicted traffic volume and required threshold. Then the severed users may be offloaded to a new target cell.

  This optimization is a complex task as there is a possibility of a negative impact on network performance due to energy-saving actions. Current energy-saving schemes face challenges like inaccurate load predictions, conflicting goals between performance and efficiency, rigid parameter settings, and actions that might improve the efficiency locally but worsen it overall.

  These issues can be addressed using ML techniques to help optimize energy-saving decisions. ML models can predict the future load and energy efficiency, enabling dynamic configuration of energy-saving strategies. This allows a balance between system performance and energy efficiency, ultimately reducing energy consumption.

- **Load Balancing**: This has been proposed to steer the traffic in a balanced distribution among available cells. With rapid traffic growth and multiple frequency bands utilized in commercial networks, the objective of load balancing is to distribute the load evenly among cells or to transfer a part of the traffic from congested cells.

  Automating network optimization can be very helpful in enhancing user experience, improving system capacity and minimizing human intervention. However, there are a few challenges to address. The current load balancing decisions based on real-time cell load status are not good enough due to rapidly changing network conditions, especially in areas with high mobility and numerous connections. This can lead to frequent handovers, cell overload and degraded user service quality.

  Guaranteeing overall network and service performance during load balancing is difficult. Offloading users from a congested cell to a target cell can eventually overload the target cell. Determining whether service performance after offloading meets the desired targets is also difficult. ML models can address these issues by leveraging data from various sources, including user feedback, network node measurements, and historical data to predict the load and improve load balancing performance. This will ultimately help in enhancing user experience and system capacity.

2.2.3 RAN Intelligence Framework

3GPP has defined the functional framework for RAN intelligence [7].

- **Data Collection:** A function that provides input to Model Training and Model inference functions. No data processing is carried out here. Examples include measurements from user equipment or different network entities and feedback from the actor among others. The collected data can be divided into two broad categories.

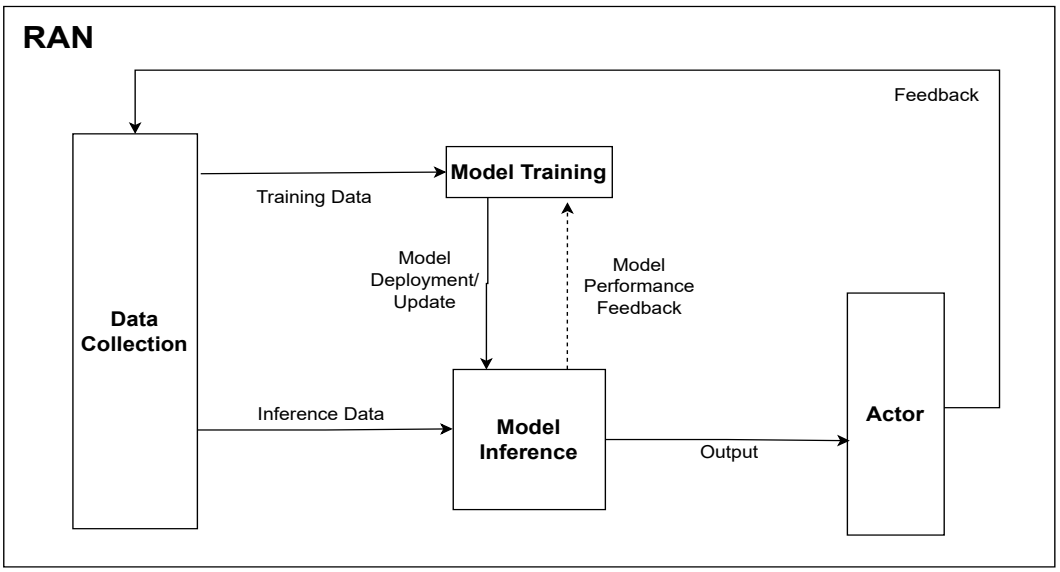


Figure 2: Functional Framework for RAN intelligence

  - Training Data: Input data for the ML Model Training function, which is used to train or retrain the model.
  - Inference Data: Input data for the ML Model Inference function during operation which is used to produce the prediction outputs

- **Model Training:** A function that performs ML model training, validation and testing. Model performance metrics can be a part of this procedure. Data pre-processing, if required, happens here. A trained, validated and tested ML model is then deployed to the Model Inference function.

- **Model Inference:** Provides ML model inference output (prediction and decisions). Data preparation happens here for the inference data delivered by the data collection function. The details of output produced by Model Inference are use-case specific.

- **Actor:** Receives the output from the model inference function and triggers or performs corresponding actions. Feedback information can be used to derive the training data, inference data or to monitor the performance of the ML model.

Depending on the deployment and use case, these ML functionalities can either all be located in the same RAN component or be distributed across multiple components of RAN architecture. For the sake of simplicity, and the available implementation for study, it is considered that all RAN Intelligence components are located in the base station.

2.3 Threat Modeling

Threat modeling can be defined as a systemic review of the architecture or design of a given system to identify and fix design-level security issues. Threat modeling can provide a clear view of a project that will justify security efforts. It helps the stakeholders to make rational security decisions with all the information in one place.

There are many different approaches to threat modeling and none of them are perfect on their own. Threat modeling can be summarized as a structured process using which potential security threats and vulnerabilities can be identified, their impact quantified and mitigation techniques prioritized to protect the resources.

Three major ways to approach threat modeling are:

- **Asset-centric threat modeling** focuses on the system resources or "assets" and the business impact if the targeted asset is compromised. This method makes it easier to prioritize security efforts and resource allocation as it has a clear prioritization for assets requiring protection. It aligns well with business objectives as well as the security measures can be directly tied to business-critical assets. However, it comes with shortcomings such as a requirement for continuous updates as the asset's value can change over time. There is also a risk of overlooking the less critical assets, which may be exploited to gain access to more valuable ones, and overall identifying and categorizing all the assets can be a complex and time-intensive exercise in large organizations.

- **Attack-centric threat modeling** focuses on attacks against the system which are more likely to occur. This method allows for the development of a proactive defense mechanism by focusing on potential attack vectors and techniques. It is more adaptive to emerging threats and provides a practical understanding of potential threats. On the other side, it might miss novel or unknown threats and the focus is mostly on known attack vectors. Enumerating all possible attacks can be a complex task which may require significant expertise. It can also lead to an imbalanced security evaluation as there is a risk of overemphasis on certain attack vectors while neglecting others.

- **System-centric threat modeling** focuses on the understanding of the system before analyzing the threats. This method provides a comprehensive view of the entire system and integrates well with system design and development processes making early identification and mitigation of threats easier. It can be quite complex in large systems and become overwhelming and difficult to manage. There is also a risk of some specific threats being overlooked which might not be apparent from a high-level system view.

Each approach has its own advantages and disadvantages, and the decision on which to employ depends on the specific context and needs of the organization. In practice, integrating multiple approaches can yield a more thorough and effective threat modeling strategy.

2.3.1 STRIDE

STRIDE threat modeling was developed by Microsoft and it is one of the most used threat modeling methods today [8]. Originally invented by Loren Kohnfelder and Praerit Garg, it was designed to help identify the different types of attacks that can be done on any software system. It is an abbreviation for "**S**poofing, **T**ampering, **R**epudiation, **I**nformation Disclosure, **D**enial of Service, **E**levation of Privileges".

- **Spoofing** can be understood as impersonating someone else in the digital world. This method is used by an attacker to bypass the authorization layer into a system and gain access while masquerading as a legitimate user.
- **Tampering** is used to make illegal changes to the data or the system itself, It can violate the integrity of the system.
- **Repudiation** refers to the ability of a given actor to perform an action and refuse the responsibility towards it. In an ideal and secure system, non-repudiability is desired.
- **Information disclosure** can occur when some confidential information is exposed to unauthorized entities.
- **Denial of Service** violates the availability of a service or the system. It is an attempt by the attacker to restrict the use of the system by legitimate users.
- **Elevation of Privilege** attacks occur when a limited user gains more access and rights, such as that of an administrator, either temporarily or permanently. Such a user can exploit these extra privileges to perform malicious actions.

It is one of the most mature frameworks available for threat modeling today. There are different approaches built on STRIDE such as STRIDE-per-element and STRIDE-per-interaction among others.

The threats identified are the opposite of the desired properties of the system being evaluated. One point to note is that while using STRIDE for threat modeling, the aim is to find and list down the things that can go wrong and the exact method of how it can go wrong or exploited can develop later [9].

STRIDE framework can be used to classify the threats for the system being evaluated and eventually aids in assessing them and mitigating the risks.

2.3.2 LINDDUN

LINDDUN, created by Deng in 2010 [10] explicitly mirrors STRIDE-per-element threat modeling [9]. It is a privacy-focused threat modeling framework, inspired by STRIDE and is complementary to it. It encourages analysts to evaluate privacy concerns in a systematic way.

LINDDUN is a mnemonic for the following properties:

- Linkability
- Identifiability
- Non-repudiation
- Detectability
- Disclosure of information
- Unawareness
- Non-compliance

Typically a LINDDUN based threat modeling consist of three steps [11]:

- Model the system: this can involve a data-flow-diagram representing the high-level components of the system.
- Elicit threats: Identify and document the threats while systematically analyzing each of the data flow diagram elements.
- Manage threats: this is a solution-oriented step where the threats are prioritized based on their risk and mitigation strategies are identified.

Industry feedback indicates that the content of LINDDUN is comprehensive but fairly complex at the same time, which in turn requires considerable privacy expertise.

2.3.3 OWASP Threat Modeling Project

OWASP is an open community dedicated to improving the security of software. They work towards developing standards, promoting awareness and providing resources to tackle the security problems.

They have described threat modeling as a four-question framework [12].

- **What are we working on?**
  The first step calls out for scoping the work. Is it just a part or the entire system that is being evaluated? This can involve drawing data flow diagrams and identifying entry points and access available to the system. This is the section where the "assets" and "attackers" are identified.
- **What can go wrong?**
  This can be a simple brainstorming exercise or a structured work using STRIDE, kill-chains, or other frameworks for classifying the threats. The goal of this step is to identify the potential attacks and the vulnerable targets from the attacker's perspective.

- **What are we going to do about it?**
  Here the counter-measures are identified to mitigate or manage the risk. This also means that the threats might have to be evaluated from the point of business impact they pose. Accepting a business impact from the threat, and then evaluating how to eliminate the vulnerable points, mitigate the risks by adding checks and controls, or transfer the risk to some other stakeholder.

- **Did we do a good job?**
  This involves assessing the work done i.e., finding out if the threat modeling is good enough for the system.

2.4 Attack Libraries

Frameworks such as STRIDE or LINDDUN may seem to be obscured high-level strategies only when one has a new system and not enough information about what can go wrong with it. A more detailed compilation of all the common problems can be very useful when trying to do threat modeling for a new system.

Similarly, no attack library is suitable for all applications and different libraries address different goals. The level of detail included for any particular attack depends on the target audience as well as the scope. It can be organized in a structured manner which will promote consistency between entities. Anyone referring to an attack library should remember that it is not an exhaustive list or a complete enumeration of everything that can go wrong.

2.4.1 NIST

The National Institute of Standards and Technology (NIST) is a federal agency in the U.S.A. and plays a crucial role in cybersecurity by developing and promoting best practices and guidelines.

NIST has published an extensive report on "Trustworthy and Responsible AI" [13] which is a significant effort towards a standard taxonomy for ML security evaluation. It provides a good structure to classify the attacks as well as the attackers based on different criteria.

For predictive AI, it focuses on attacks specifically against the machine learning model and algorithms, rather than attacks against the broader system surrounding it. Three major categories were defined: Availability, Integrity and Privacy. It further goes on to define the possible goals, capabilities and knowledge of an attacker, and then the threats concerning a system based on predictive ML.

The taxonomy defined here is also a base for this thesis study.

2.4.2 MITRE

MITRE is a not-for-profit American organization involved in multiple federally funded research supporting U.S. government agencies. It provides guidelines, researches, frameworks among many other things which are helpful in protecting against cyber-threats.

Common Attack Pattern Enumeration (CAPEC) is a structured set of more than 400 attack patterns organized in about 15 groups. This can be used for threat modeling in several ways, such as reviewing a system against each CAPEC category, or each CAPEC entry if a more detailed analysis is required.

Advantageous Tactics, Technologies, and Common Knowledge (ATT&CK) is a framework created by MITRE in 2013 [14]. This framework outlines the components of tactics, techniques and procedures that an attacker can use and their interconnections. It aims to improve system detection capabilities and enhance threat intelligence analysis while strengthening security risk assessments.

While MITRE CAPEC focuses on attack patterns, MITRE ATT&CK focuses on techniques that an attacker can use. The ATT&CK framework describes specific techniques in a very detailed way and is commonly used for threat intelligence and incident response. These two frameworks can be used together to provide a better understanding and complete picture of the cyber threats concerning a given system.

MITRE has developed an adversarial ML threat matrix called Adversarial Threat Landscape for Artificial-Intelligence Systems (ATLAS) [15]. It aims to help security analysts by positioning the attacks on ML systems in an ATT&CK-style framework. ATLAS can be used for threat assessment and to understand real-world adversary behaviors and mitigation pathways. It also helps to report unique real-world attacks on AI-enabled systems.

2.4.3 OWASP Top 10

OWASP offers a top ten risks list every year. OWASP ML Top 10 is one of their projects which identifies the top 10 risks concerning a machine learning system. While NIST focused mainly on adversarial attacks, OWASP also considered non-adversarial scenarios.

Each of the ten attacks have bees assigned a risk factor which helps to understand the impact and possibility of the given attack. A few possible prevention strategies are also mentioned for each attack, though many of the prevention strategies are common for some of the attacks [16].

Depending on the extent of threat modeling, OWASP's list of top 10 risks area good supplement to STRIDE. However, over time, this list as an attack library may be more or less valuable as it is based on the input of volunteering members and gets updated every year.

3 Problem Statement

Many features in mobile networks which are in the planning phase and a few in the development and trial phases are leveraging ML models, offering significant benefits such as resource and mobility performance optimization and predictive maintenance among others. This adoption of ML in RANs brings in the need to study and find the new vulnerabilities and risks ML integration brings and how they can be managed.

There is a push in all sectors to take advantage of ML at a rapid pace, and mobile networks are not untouched either. Despite the rapid incorporation often focused on functionality, security must not be sidelined. Not taking care of the security around new technology can lead to significant risks, as ML models become susceptible to various attacks, potentially compromising the reliability and safety of mobile networks.

Some attacks against ML systems may not be possible when it is used in a RAN because of additional layers of security available in RANs. If it is possible to create the same type of attack for the ML system inside a RAN, the way to initiate it can be different compared to those used for a similar ML system in some other environment. The potential attackers and their capabilities will also be very different in the context of RANs. Moreover, the mobile networks, or RAN are part of the critical infrastructures which makes it important to have a dedicated evaluation of the security landscape with the integration of new technologies like ML. Existing research has extensively investigated security threats to ML models and applications of ML systems, yet there is a critical gap when it comes to examining these threats in the specific context of mobile networks. This thesis aims to bridge that gap by assessing and understanding the security risks associated with ML features in RANs.

3.1 Core Issues

1. **Security Vulnerabilities in ML for RANs:** The rapid incorporation of ML based features in RANs without sufficient attention to security may introduce new vulnerabilities which need to be addressed.

2. **Lack of a tailored security framework:** Any generic security framework doesn't addresses all concerns related to RAN and integration of ML with RAN. Thus, there is a need for a dedicated framework to systematically evaluate and address the security risks posed by incorporating ML in mobile networks.

3.2 Research Goal

The main goal of this research is design a flexible threat modeling framework tailored specifically towards ML features within RAN, making it easier for stakeholders to assess potential vulnerabilities and adjust the model based on specific features or working groups.

3.3 Requirements

To achieve a comprehensive threat modeling framework, the following requirements must be met:

1. **Structured Classification:** The framework must provide a structured guideline to categorize attacks, assets and attackers based on their attributes such as capabilities and knowledge, making it easier to identify different asset and attacker groups to which the potential attacks can be mapped.

2. **Flexibility and Adaptability:** The framework must be flexible enough to be used for assessing any of the existing and future ML based features in RANs. The framework must be flexible and allow adjustments based on specific features or operational contexts.

3. **Simplicity and Scalability:** The framework must be easy to understand and use by all stakeholders including those who are not security specialists, such that it can be adopted as a part of the development process. It should also provide a clear path to accommodate new attacks and vulnerability considerations as and when they are discovered.

3.4 Scope

In this thesis, we restrict ourself to the following setting:

1. **Predictive AI:** This study only considers the threat modeling for features based on Predictive ML as this has more applications in RANs and was also used in the example discussed by 3GPP.

2. **Classical Base Station:** This study has been conducted to evaluate the ML based features being deployed in a classical, or physical base station environment while cloud base station would be considered for further analysis in future.

3. **Classical Base Station as a secure environment:** A classical base station can be considered a secure environment for this study. Vulnerability or attack on the environment itself is not specific to ML features and is addressed by existing security measures for the environment.

The defined scope provides a clear focus for this thesis work without limiting the potential for future expandability.

4 Design

The focus of this thesis is towards mapping the existing threats for ML to the RAN context. Thus, the first and foremost step is to understand the attacks that threaten ML systems before they are assessed in the RAN context. These threats depend on the underlying algorithm and the use case, but can also depend on the way it is implemented. Then the next step will involve understanding their application in RANs, which can be an iterative process required for each new type of feature that will be using some form of ML. This will help to narrow down the threats specific to a given feature, and will also eventually help in selecting the counter-measures for the same.

4.1 Threat Modeling Process

No single threat modeling framework is perfect in practical situations, and most often threat modeling as implemented in practice involves aspects from multiple frameworks to suit the context. While we look for vulnerabilities and threats in a system, identification of the assets and potential attackers at the beginning of the process makes it easier to map the threats and eventually helps in deciding and implementing the counter strategies. Thus, the threat modeling process described by OWASP as discussed in Section 2.3.3 is selected as the base for designing this framework. The threat modeling guidelines by OWASP is a more suitable choice over other strategies as it makes it easy to identify the unknowns and then connect them to get the whole security landscape picture. One of the end goals is also to map the existing threats for ML to RAN which acts as an attack library and is combined with the threat modeling process along with STRIDE and LINDDUN guidelines to give the final shape to the threat modeling framework in this work.

STRIDE is one of the most common strategies to classify threats for any given system. A mapping of the identified attacks to STRIDE can work as a bridge for the understanding of their relevance and impact. It also provides an easy reference to stakeholders already familiar with the threat modeling of conventional software systems.

Karaçay et. al in their work focused on threat modeling a specific use case of ML in 6G and defined the STRIDE properties in the context of ML [17], an adaptation of which is presented in the table 1.

The threat modeling process developed for this thesis involves five major components as visualized in figure 3.

Though this threat modeling process is not aiming to be purely asset-centric or attacker-centric in nature, defining assets and attackers complements the understanding of how a threat maps to the context of mobile networks.

NIST's publication on AI/ML threats is considered the main attack library for this framework as it is one of the most extensive ones available in the public domain currently. The attacks' definition and possibility of it affecting the AI/ML system in RAN is also bridged to conform OWASP ML top 10.

Terminology	Property	Definition
Spoofing	Authenticity	Unauthorized access to AI/ML System
Tampering	Integrity	Model or the data used for the model is altered
Repudiation	Non - Repudiability	Denied responsibility of malicious activity
Information Disclosure	Confidentiality	Information leak from AI/ML system by the attacker
Denial of Service	Availability	bombarding AI/ML model with large inputs, resources exhausted by the attacker
Elevation of Privilege	Authorization	Malicious actions performed by gaining higher access than intended by the attacker

Table 1: STRIDE mapping for AI/ML use cases

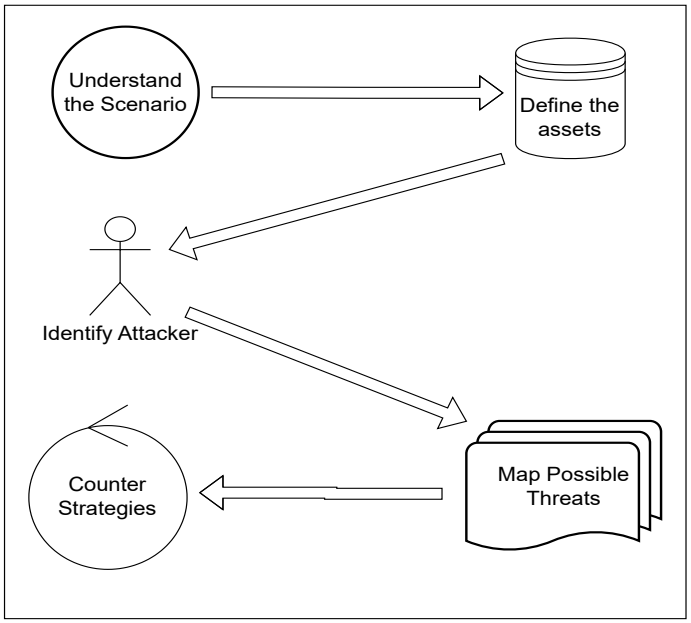


Figure 3: Threat Modeling Process

The first step in threat modeling of a feature is to gain an understanding of the feature. A good knowledge of the scenario where the feature is going to be used, and how builds the base for the evaluation. Some information helpful in the evaluation includes the deployment environment, the underlying ML algorithm type, the input data format, the training and input data source and the output destination.

A full overview of all information flows in the system also helps in identifying

vulnerable points. Ideally threat-modeling should be part of the feature development plan and responsibility of the development team. There should be involvement of all the stakeholders for a given feature to get a proper overview and insights about different aspects.

Another thing to consider is whether the new ML feature is replacing an existing algorithm or is a novel feature. Even if a threat model already exists for an existing non-ML feature, with the introduction of ML, there can be areas requiring re-evaluation. Thus, a new threat model is required that considers the implications of using ML. Though, in such case, a fallback option to the older version might be helpful as a contingency measure.

The next step in the process is to define the assets that need to be protected. These include the datasets used for training, or the ML model itself. A good understanding of the feature and different components with which it interacts with is essential to identify the assets, and many times these first two steps can be performed in parallel. The identified assets can also be categorized in a simplified way for easier understanding.

Identifying the potential attacker groups and their attributes makes the further steps of evaluating a possible attack and its counter strategy easier. The attacker attributes can include the knowledge extent of the attacks, its capabilities and motivation to compromise the system.

The next step involves identifying which threats or attacks can be initiated by the threat actors, and the danger posed by that attack in the context of the given feature being evaluated. The findings can be organized in a concise table for easy reference and additionally supplied with references if required.

Once the threats are identified, the next step entails finding out how to manage them. The actual business and revenue impact of the attack dictates the severity and the mitigation measures to a great extent. Risk management can involve measures to prevent an attack from happening, detection techniques to identify an attack, and in case of an attack measures to minimize the impact on the system.

5 Taxonomies and Classifications

We choose to tailor our process to analyzing features entirely deployed in a physical base station to better focus on the threats to the ML system and not the deployment environment. This also follows the model of RAN Intelligence proposed by 3GPP closely and thus can be easily reused for any feature following the same pattern.

5.1 Defining Assets

To understand the complete impact of any exploitation of a compromised system, we first need to understand all the components involved. Assets in a threat modeling can refer to those components that needs to be protected.

For an ML system, the main assets are the model itself, the Training Data and the Inference Data. But in the context of RAN, along with these we also consider other involved assets which can help us get a better overall picture of the feature for threat modeling. Identification of Assets for any given threat can also help to understand the vulnerable and impact points and to determine the severity of the threat.

Apart from the model and the data, we also need to consider the environment where the ML model is used i.e., source and output components, and while evaluating threats, the effects on them. Another major thing to consider for an ML-based feature is the Computation Resources, as they can be choked or compromised resulting in availability breakdown.

Asset Type	**Asset**
Data	Training Data
	Inference Data
	Training and Inference Data
Model	Model Parameters
	Model
Environment	Source data Component
	Output Destination
	Computation Resources
Framework and Processes	Deployment Processes
	Data Collection Function
	Model Training Function
	External Framework and dependencies
	Internal Framework and dependencies
Other	Other

Table 2: Classification of assets

The frameworks and processes involved must also be protected. Any undesirable behavior of such processes will result in a sub-optimal or compromised model. The most prominent processes for an AI/ML-based feature can be the deployment processes, which is used to deploy the model after training/retraining. The data collection function

and the model training function can be abstracted to do one specific job of preparing the inputs/ data sets from live data and training/retraining the model. On the other hand, the frameworks can be internal or external. In this context, frameworks refer to any in-house or 3rd part dependencies or tools, such as TensorFlow, used in any phase of the ML model's life cycle.

5.2 Classifying the Attacker

The next step is to identify and classify the attackers. In the context of ML features in RAN, one of our assumptions is that the physical components such as the base station and data transfer channels are secure and cannot be accessed by unauthorized persons.

When the mobile network is in operation, the operator is responsible for running, maintaining and protecting the mobile network. Thus for RAN applications, we try to identify threat actors (attackers) from the Operator's point of view.

To have a malicious intent the threat actor need not have any knowledge of the feature; but to be able to affect it adversely, it should be aware of the existence of the AI/ML features. These threat actors can be put into two broad categories - Internal and External.

From the Operator's Point of view, the only internal threat actors are their employees. All of the remaining groups of threat actors are external to the Operator.

The external group may contain the following groups of attackers:

1. Employees in the Feature development team at the manufacturing organization
2. Employees in Troubleshooting/Maintenance team
3. Remaining employees at the feature development Organization
4. Outsourced teams/workers by the Operator
5. Other 3rd Party actors

5.2.1 Classification

Once the threat actors are identified, we classify them based on three properties.

Goals The goal a threat actor or attacker might want to achieve by compromising the AI/ML system.

Capabilities What the attacker is capable of doing.

Knowledge How much knowledge the attacker has about the system.

5.2.2 Goals

NIST has mentioned three major goals for an attacker with respect to AI/ML systems [13].

AG-01 Availability Breakdown The attacker attempts to degrade the performance of the model at deployment time

AG-02 Integrity Violation Integrity of the model's output is the target, resulting in incorrect predictions

AG-03 Privacy Compromise extracting information about the training data or the ML model itself

To determine if an attacker might have one of these goals, we should consider if that particular group will have any motive (e.g., economic, emotional, etc) to compromise the ML system.

5.2.3 Capabilities

The report by NIST mentioned six types of capabilities, which is also considered a standard taxonomy for this thesis [13]. These capabilities are also viewed from the perspective of RAN system and adjusted accordingly. One major deviation from the NIST report is that the capability called "Label Limit", where the attacker does not control the label of the training sample. We consider "Label Limit" to be a part of "Training Data Control" instead of a distinct capability. This change is done because "Label Limit" is essentially one particular type of data manipulation among several other types that can be done by someone with "Training Data Control". The "Query Access" capability also considers the passive aspect as it can enable the attacker to create similar attacks in RAN environment which are possible using an active query access in other systems.

AC-01 Training Data Control: The attacker controls a subset of training data by adding or modifying training samples

AC-02 Model Control: The attacker controls model parameters such as By generating and inserting a trojan trigger or By sending malicious local model updates in federated learning

AC-03 Testing Data Control: The attacker adds perturbations to testing samples at model deployment time

AC-04 Source Code Control: The attacker modifies the source code of the ML algorithm, including dependencies and libraries

AC-05 Query Access: The attacker with active query access submits queries to receive prediction and uses it for privacy attacks, e.g., in cloud MLaaS. In the case of passive query access, the attacker doesn't interact with the ML model directly but can access both the input and the output.

5.2.4 Knowledge

The knowledge an attacker may possess is categorized into three broad categories, normally used as a standard across domains.

AK-01 White-box full knowledge about the ML system

AK-02 Black-box minimal knowledge about the ML system

AK-03 Grey-box somewhere between white-box and black-box

Any adversarial actor, or attacker with full knowledge of the model algorithm, model parameters and how the model is deployed can be said to have white-box knowledge. This can include groups such as members of the team who were involved in designing the ML system. A total outsider with no knowledge of these three things and its working can be said to have black-box knowledge and grey-box knowledge would be a broad spectrum which includes different extents of partial knowledge.

5.2.5 Threat Actor Scenario

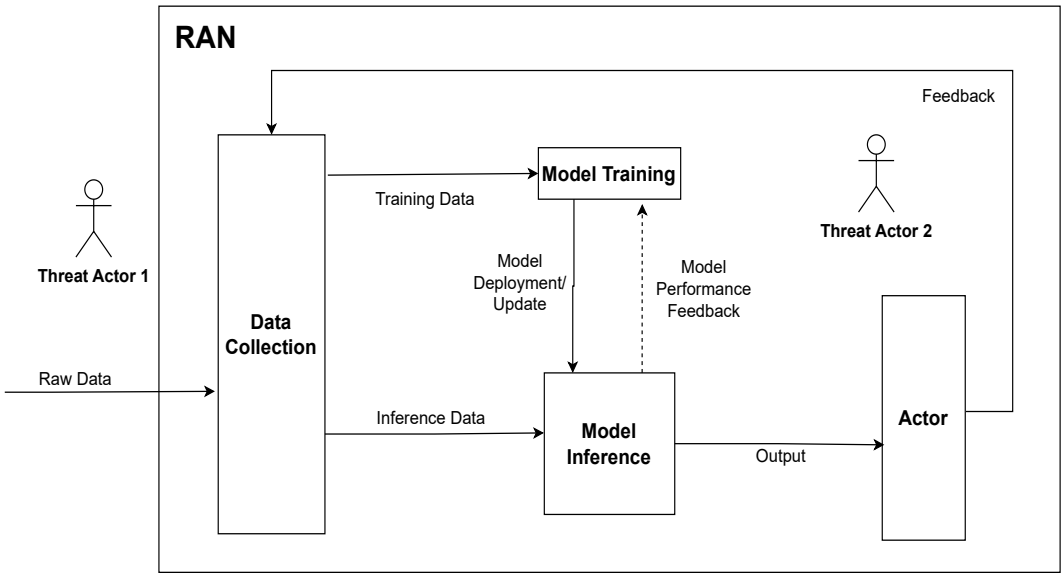


Figure 4: Threat Actors in RAN Intelligence

Figure 4 represents an example attacker scenario to understand the capabilities and knowledge of the attackers. The Threat Actor 1 (TA-01) is present outside the RAN environment and has no access to the elements inside the RAN. The Threat Actor 2 (TA-02) represents an attacker that can assess some elements inside the RAN. The capabilities discussed here may be available in different combinations to different groups of a threat actor, depending on the attack and method used by the attacker.

TA-01 can add perturbations or new data or manipulate the Data/information in some way before it is received by the 'Data Collection Function' and thus can get Training data control (AC-01).

There is also a possibility that TA-01 can induce Model Control (AC-02) by adding some trojan triggers (which don't affect the working of the model until an input which can activate them is received) in the Data that is received by the data collection function.

TA-01 can also attempt Source Code Control (AC-04) by modifying one of the OpenSource frameworks/dependencies that might be used in the ML model.

TA-02 can add perturbations / add new data or manipulate the Data/information in some way before it is received by the 'ML Management Processes' and thus can get Training data control (AC-01) as well as Testing Data Control (AC-03).

There is also a possibility that TA-02 can induce Model Control (AC-02) by adding some Trojan triggers or controlling the model parameter in some other way if TA-02 was involved in the development or maintenance of the ML model/feature.

TA-02 can also attempt Source Code Control (AC-04) by modifying one of the internal frameworks/dependencies that might be used in the ML model.

Since TA-02 can see both the input and output of inference data, it can attempt a passive Query Access (AC-05) and use the information to train a clone model.

5.3 Attack Library

A consolidated list of threats, largely based on the NIST definition of attacks for predictive AI, was considered. Further, their applicability in the context of RAN was analyzed.

5.3.1 Evasion Attacks

A testing sample whose classification can be altered to an arbitrary class as chosen by the attacker with minimal perturbation, when the model is deployed, is referred to as an adversarial example. In evasion attacks, the threat actor generates adversarial examples [18]. Evasion attacks take place after the AI/ML system is deployed. The adversary attempts to alter an input, i.e., generate an adversarial example, to change the system's response. One example can be altering some road sign which is then misinterpreted by an autonomous vehicle and it is not able to function properly.

- **T-01 Optimization-based methods**
  Biggio et al. proposed the use of optimization techniques to generate adversarial samples [19]. Two types of adversaries were considered. First, the adversary with white-box knowledge, where the adversary knows the feature space, classifier type and model parameters. Second, adversaries with grey-box knowledge, in which the attacker knows the classifier type and the feature representation but not the parameters of the model or the training data. The adversary can modify the input data and feature vectors. The attacker aims to find an adversarial example x within a maximum distance d_{max} from an initial malicious sample x^0

which will evade the classifier with maximum confidence. This method was applied to several models such as linear classifiers, Kernel SVM and Multi-layer perceptrons.

- **T-02 Universal Evasion attacks**
  Universal evasion attacks focus on finding a single perturbation that, if introduced in the input data, results in the misclassification of all the inputs fed to the ML algorithm. Such perturbations are termed as universal as they are data-point agnostic. Moosavi-Dezfooli et al. demonstrated a technique for image classification models, based on successive optimization of the universal perturbation using a set of points sampled from data distribution [20]. One key finding of their research was that these universal perturbations generalize well across different classification models.

- **T-03 Physically realizable attacks**
  The attacks that involve some form of physical interaction can be categorized as physically realizable attacks. Sharif et al. demonstrated how a facial biometrics system can be compromised by an attacker to avoid recognition by just printing a pair of eyeglass frames [21]. Though their research was focused on a white-box recognition system, they also mentioned how similar techniques can be used in a black-box setting.

- **T-04 Score-based attacks**
  The output of predictive ML algorithms are associated with a numerical value that indicates the level of certainty or accuracy of the prediction, known as confidence score. In this attack, the attacker uses various optimization techniques, with a reference to the model's confidence score or logits, to create the adversarial example. Some of these optimization methods such as zeroth order optimization involve estimating the gradient signal based on input queries as demonstrated in [22]. Seungyong Moon et al. discussed how a black-box adversarial attack can be crafted without the need for estimation of the gradient using an efficient combinatorial optimization [23].

- **T-05 Decision-based attacks**
  In this attack, the attacker only obtains the final predicted labels. Brendel et al. introduced the first decision-based attack called boundary attack, which started by introducing a large adversarial perturbation and then focused on reducing the perturbation while staying adversarial [24]. Such attacks rely solely on the model's decision and are closer to real-world scenarios with less knowledge about the system.

- **T-06 Transferability of Attacks**
  Any malicious input which affects one model can often affect other models as well if they are trained to perform the same task, irrespective of the algorithm or training data used. An attacker can craft a white-box attack on a substitute model, which performs a similar or the same task as the target model, and then

transfers it to the target model. The substitute model can be trained in different ways. One of the ways to do that would be actively querying the target model to generate the input/output data, or if such dataset is readily available to the attacker.

5.3.2 Poisoning Attacks

Poisoning attacks involve inserting maliciously chosen data into the datasets or the algorithm code base to achieve some effect or disrupt the proper functioning of the system. They are initiated during the training phase of the ML life cycle. Poisoning attacks have a history of over a decade in cybersecurity since the first known attack in 2006 for worm signature generation [25]. A paper by Kumar et al. published in 2020, which involved interviewing 28 organizations, revealed poisoning attacks to be the most critical vulnerability of machine learning systems deployed in production [26].

Poisoning attacks can target both the availability and integrity of a system. For example, inflating the training data set with multiple unrelated inputs would lead to a differently trained model than expected, and eventually wrong and harmful predictions for whichever system it is going to be used in.

- **T-07 Availability Poisoning**
  Availability Poisoning aims to disrupt the ML model's normal functioning, eventually resulting in a denial-of-service scenario for users. Availability poisoning can be achieved in both white-box and grey-box settings. The initial poisoning attacks identified in cybersecurity applications were availability attacks against worm signature generation and spam classifiers. These attacks had an indiscriminate effect on the entire machine learning model, essentially resulting in a denial-of-service attack against users of the AI system. Predisci et al. demonstrated that the deliberate introduction of noise, even at a relatively low level such as 50%, can significantly compromise the reliability of a worm signature generator, hindering its ability to produce effective worm signatures [25].

- **T-08 Backdoor Poisoning**
  Gu et. al. in 2017 observed that image classifiers can be compromised by introducing a small, inconspicuous trigger pattern into a portion of the training images and then reassigning their labels to a specific target class. The classifier subsequently learns to link this trigger with the target class, leading to the misclassification of any image containing the trigger or backdoor pattern into the target class during testing [27].

  In the last few years, backdoor attacks have become more sophisticated and stealthy, making them harder to detect and mitigate. Wenger et al. poisoned facial recognition systems by using physical objects as triggers, such as sunglasses and earrings [28].

- **T-09 Targeted Poisoning**
  Unlike availability attacks, targeted poisoning attacks induce a change in the

ML model's prediction for a specific, small number of samples. Label flipping, where the attacker inserts samples with the target label, is effective if they control the labeling function. However, targeted poisoning attacks are mainly studied in clean-label settings where the attacker cannot alter labels.

- **T-10 Model Poisoning**
  Model poisoning attacks aim to directly manipulate a trained ML model to insert harmful functions. In centralized learning, Liu et al. identified triggers within a trained neural network and then retrained the model by embedding the trigger in external data, thus poisoning it [29]. Many model poisoning attacks focus on federated learning where clients send local model updates to a server. Compromised clients sending malicious updates can poison the global model, causing availability and integrity issues.

5.3.3 Privacy Attacks

Privacy attacks are also deployment time attacks. In this, the attacker attempts to learn some sensitive information about the AI/ML system or the data that was used to train it for malicious activities. One example can be reverse engineering an ML algorithm by repeatedly querying it with legitimate inputs and then exploiting the vulnerabilities.

- **T-11 Data Reconstruction**
  Data reconstruction attacks extract individual data from aggregated information that has been released. Nissim et al. introduced reconstruction attacks and discovered that smaller perturbation always results in a strong violation of privacy [30]. The U.S. Census Bureau conducted an extensive study to evaluate the risk of data reconstruction attacks on census data. This study was a key factor in the decision to implement differential privacy in the official release of the 2020 U.S. Census data [31]. Neural networks' tendency to memorize their training data partly explains their ability to reconstruct training samples. Zhang et al. examined how neural networks can memorize datasets composed of randomly selected data [32].

- **T-12 Membership Inference**
  Membership inference attacks, which reveal private details about individuals similar to reconstruction or memorization attacks, remain a significant concern when releasing aggregated information or machine learning models trained on user data. In some cases, simply establishing that an individual was included in the training dataset can have privacy implications, such as in the medical study of patients with a rare disease. Moreover, membership inference can serve as an initial step for mounting data extraction attacks. Here, the attacker's objective is to find whether a specific record or data sample was included in the training dataset utilized for the statistical or ML algorithm.

- **T-13 Model Extraction**
  In a model extraction attack, the attacker aims to gain knowledge about the

model's architecture and parameters by sending queries to the ML model hosted by an MLaaS (Machine Learning as a service) provider. Tramer et al. initially demonstrated such attacks on various online ML services, targeting different ML models, including logistic regression, decision trees, and neural networks [33]. Jagielski et al. have demonstrated that it is not possible to perfectly replicate an ML model. Instead, a functionally equivalent model, while different from the original, can be created that achieves similar performance on the prediction task [34]. Moreover, they have shown that even this less demanding task of extracting functionally equivalent models is computationally challenging.

It's important to understand that model extraction often serves as a stepping stone rather than the ultimate goal. Once the model's architecture and weights are exposed, attackers can execute more potent attacks typically associated with white-box or grey-box scenarios. Thus, preventing model extraction can effectively mitigate subsequent attacks that rely on the attacker's knowledge of the model's internal workings.

- **T-14 Property Inference**
  In property inference attacks, the attacker aims to extract general information about the distribution of the training data by interacting with an ML model. For example, they might try to determine the proportion of the training set possessing a particular sensitive attribute, like demographic information. This can reveal potentially confidential details about the training data that were not meant to be disclosed.

6 Implementation

The threat model puts together all the different pieces of the threat modeling process in one single place. This chapter further discusses the template which was created using the threat modeling process and how the modularity makes it easy to connect different parts with each other. Further, the chapter delves into the mitigation strategies for the attacks that are in the attack library for this threat modeling framework.

6.1 Threat Model

A threat modeling template is created with the information presented in Figure 5. The template can have several tables, depending on the need, in a spreadsheet in a concise way and can be made available to all stakeholders as a tool for further review and as a summary of security evaluation.

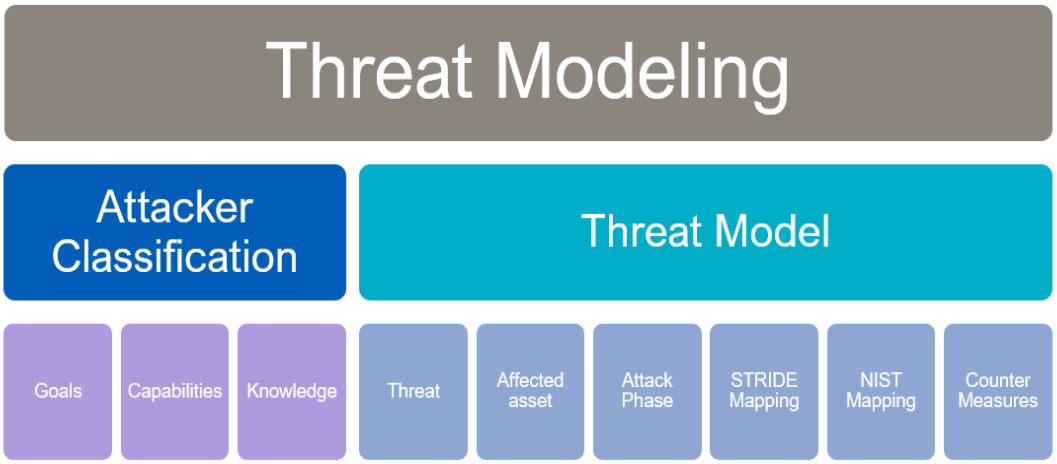


Figure 5: Simplified Threat Model Contents

The template created during this thesis comprises three main tables. First table included the basic information of the feature being evaluated, such as its name, description, base algorithm, deployment environment, input data format and a link to the feature documentation. The second table lists the attacker groups and summarizes their knowledge and capabilities at one place. This is demonstrated in Table 3 and Table 4 separately but they are included in the same table in the spreadsheet template.

The third table summarized the possible attacks on the feature, where each possible attack was mapped against the affected asset as demonstrated in Table 8. It also had columns mapping the attacks to the attack phase - inference or training, STRIDE properties, NIST categories of Integrity, Availability or Privacy attacks, and a column for the suggested counter measures. An impact severity can be assigned to each of the identified threats based on the effort required to initiate the attack and the harm that it can cause to the system.

Attacker	AK-01	AK-02	AK-03
Feature Development team			
Feature Maintenance team			
Remaining Employees at feature org.			
3rd party malicious actor			
Outsourced teams by operator			
Operator's internal teams			

Table 3: Attacker Knowledge

Attacker	AC-01	AC-02	AC-03	AC-04	AC-05
Feature Development team					
Feature Maintenance team					
Remaining Employees at feature org.					
3rd party malicious actor					
Outsourced teams by operator					
Operator's internal teams					

Table 4: Attacker Capabilities

In some situations, it may be necessary to provide additional details along with the summary by filling in the template. These details can include detailed information on how the identified attacks can be crafted by the attacker, the pros and cons of the probable preventive measures and any possible contingency plan for the immediate implementation in case of an attack.

6.2 Mitigation Strategies

Mitigating attacks on ML systems is challenging because adversarial examples are widespread in a variety of ML model architectures and a strong mitigation strategy may not be feasible to implement. Any mitigation strategy is not perfect and may come with its own issues, and thus the selection of such strategies would depend on other factors such as implementation environment, ML algorithm base and cost-benefit analysis among others.

6.2.1 Adversarial training

Adversarial training involves training the model on a data set that has been perturbed intentionally. This makes the ML model safe from attack examples where input data is altered slightly to confuse it. Adversarial training may incorporate a method to penalize the model for being overly sensitive to small changes in the input data. This results in a reduced chance of a successful evasion attack as the model's prediction will not change drastically with minor perturbations.

Adversarial training was introduced by Goodfellow et al. in 2014 [35] and expanded on by Madry et al. in 2017 with a focus on making deep learning models resistant

to adversarial attacks [36]. Tsipras et al. in 2018 demonstrated that the additional robustness achieved through adversarial training of an ML model comes at the cost of accuracy [37]. Apart from consuming more resources, the disadvantage of a decreased model accuracy on clean data needs to be considered while opting for this mitigation or prevention technique. It was also found to have some unexpected benefits such as more semantic meaning, i.e., capturing meaning in a way which is more clearer, than standard models.

6.2.2 Formal verification

Techniques from Formal Methods can be used to certify the adversarial robustness of a neural network. Reluplex introduced by Katz et al. uses satisfability Modulo Theories (SMT) solvers to verify the robustness of deep neural networks, without making any simplifying assumptions [38]. This technique extends the simplex method to accommodate the non-convex Rectified Linear Unit (ReLU) activation functions, an important component of many modern neural networks. Along with ReLU, ma-pooling layers are also a key layer in modern neural networks, and it was considered in AI^2 introduced by Gehr et al. for analyzing realistic neural networks [39]. Along with being precise, AI^2 has other benefits such as being significantly faster and the ability to handle deep convolutional networks.

These research demonstrate that though a very robust method, formal verification is difficult to implement and needs to be modified to be practical. The practical application of formal verification, despite its potential for certifying neural network robustness, is hindered by limitations such as limited scalability, high computation cost, and restriction in supported operation types.

6.2.3 Training Data Sanitization

Training data sanitization involves removing or repairing parts of training data suspected of poisoning before the model training is performed. It spans across multiple techniques such as anomaly removal, noise reduction and imputation of missing values among others. Nelson et al. introduced a Reject On Negative Impact (RONI) defence which measured the incremental effect of each sample on the model accuracy and excluded it from the training if the accuracy would decrease [40].

Though the data sanitization techniques provide improved data quality and enhanced security over poisoning attacks, they are cost and resource-intensive to implement and may have consistency issues. Data sanitization is more effective when the poisoning attack influences a significant portion of the training data. However, it is less effective against subtle, stealthy poisoning attacks, thus creating a trade-off between the success of the attack and the detectability of the malicious samples.

6.2.4 Robust Training

A different strategy to counter availability poisoning attacks involves altering the machine learning training algorithm to conduct robust training rather than standard

training. An ensemble of multiple models can be trained and the predictions are produced through model voting. Creating multiple versions of a model using different subsets of the training data and averaging their predictions also reduces overfitting [41]. In a setting which consists of subsequently built models, with each one of them focusing on correcting the errors made by the previous model, combines the strength of weak models to create a strong overall model.

Some of the disadvantages of such robust training using ensemble models can include high computational costs and the need for continuous training.

6.2.5 Differential Privacy

Differential Privacy (DP) has been rigorously defined led by the discovery of reconstruction attacks against aggregate information [42]. DP is a robust privacy definition that ensures a bound on the amount of information accessible to an attacker with the algorithm's output about a particular record in the original dataset. DP has gained widespread adoption due to several valuable properties: group privacy (the extension of the definition to two datasets differing in k records), post-processing (privacy remains protected even after the output has been processed), and composition (privacy is is preserved across multiple computations performed on the dataset).

By definition, DP provides protection against data reconstruction and membership inference attacks. The definition of DP inherently indicates an upper limit on how successful an adversary can be when attempting a membership inference attack. Thudi et al. [43] derived strict limits on the success of membership inference attacks. DP does not provide protection against model extraction attacks since it is designed to secure the training data and not the model. Various papers have reported adverse outcomes of using differential privacy to defend against property inference attacks [44].

One of the limitations of using DP in real world applications is configuring the privacy parameters to balance the balance between privacy levels and utility, often assessed in terms of accuracy for ML models.

6.2.6 Trigger Reconstruction

This class of mitigation focuses on reconstructing the backdoor trigger, assuming it remains in a fixed position in the poisoned training data samples. NeuralCleanse by Wang et al. introduced the first trigger reconstruction method, leveraging optimization techniques to identify the most likely backdoor pattern that consistently leads to misclassification of test samples [45]. Subsequent improvements have enhanced performance across multiple classes and enabled support for multiple triggers within the model [46]. Artificial Brain Simulation (ABS) by Liu et al. activates multiple neurons and measures their responses to reconstruct trigger patterns [47].

6.2.7 Model inspection and Sanitization

The model inspection involves analyzing the trained ML model prior to deployment to determine if it has been poisoned. An early contribution to this field is NeuronInspect, which applies explainability methods to determine distinguishing features between clean and backdoored models, subsequently employing these features for outlier detection [48]. DeepInspect employs a conditional generative model to determine the probability distribution of trigger patterns and implements model patching to eliminate the trigger. [49]. Xu et al. introduced the Meta Neural Trojan Detection (MNTD) framework, which trains a meta-classifier to predict whether an ML model is compromised with a backdoor (or Trojaned, as termed by the authors) [50]. This technique is versatile and applicable to various data modalities, including vision, speech, tabular data, and natural language processing (NLP). Once a backdoor is detected, model sanitization can be achieved through pruning, retraining, or fine-tuning to recover the model's accuracy.

7 Evaluation

The evaluation of this framework assesses the effectiveness and robustness of the proposed threat modeling template to address the security risks posed by AI/ML applications in RAN. The evaluation focuses on validating the framework's effectiveness in translating the potential vulnerabilities in RAN context, mapping the attacks to assets and attackers, and its adaptability for evaluation of different AI/ML-based features in RAN.

7.1 Approach

We first describe the approach that was used to test the usability of the proposed threat modeling framework.

- Example Feature considered: An example feature deployed within the classical base station setup is analyzed using the framework. The feature leveraged predictive AI outputs based on different measurements for resource optimization.
- Information gathering: Various stakeholders involved with the development and progress of the feature were involved in the information gathering phase to successfully understand the feature before commencing the security analysis.
- Threat modeling: The threat modeling process was applied to the selected features independently by three team members utilizing the system, asset and attacker-centric approaches involved.
- Evaluation and review: The threat modeling was then reviewed together, helping to identify corner cases with proper justifications.
- Feedback: The final threat modeling framework and the threat model for the specific example feature were reviewed by a wider group involving non-security and non-ML stakeholders as well. Their feedback was incorporated into the framework to make the documentation and explanations easy to understand by everyone.

7.2 Case Study

A simple example was considered for the initial threat modeling process where the AI/ML system is deployed in the base station and data is collected from user equipment measurements. This is a basic scenario which can be relevant in different use cases such as load distribution and cell allocation among others.

The proposed threat modeling framework is applied to this scenario, similar to the load-balancing scenario discussed in section 2.2.2. The threats identified will apply to other scenarios as well but the method to initiate or apply them need not be the same.

7.2.1 Defining Assets

Since major assets in any ML system are the data and the model itself. The data was further divided into "inference data" and "training data" while defining assets. An option of "training and inference data" was included to make it easier to map to the attacks which involve or affect both of them.

The model asset type was also divided into two parts, viz., "Model Parameters" and "Model" itself.

In this example feature, the source data is coming from user equipments, and the output would be used by some other process inside the RAN for load-balancing or other applications. The ML model is deployed inside the physical RAN environment and is using some computation power as well. All these things are considered to be part of the asset type "Environment".

Other dependencies like the deployment or data collection process and external or internal frameworks are considered to be part of "Frameworks and Processes".

Another asset type "Other" was kept to include anything which might be identified later and cannot be part of any of the above categories.

This is summarized in the Table 5.

Asset Type	**Asset**
Data	Training Data
	Inference Data
	Training and Inference Data
Model	Model Parameters
	Model
Environment	Source data Component
	Output Destination
	Computation Resources
Framework and Processes	Deployment Processes
	Data Collection Process
	Model Training Process
	External Framework and dependencies
	Internal Framework and dependencies
Other	Other

Table 5: Classification of assets

7.2.2 Classifying Attackers

We start with the assumption that everyone can be an attacker given the correct combination of motive, effort and capability. It can be an outsider with no previous interaction with the system or someone who is or was associated with the system at some point in time.

From the operator's point of view, this leads us to the following six groups of threat actors:

1. **Operator's internal team:** The employees at the operator's organization who may or may not interact with the ML-based feature.

2. **Feature development team:** The team that is involved at different stages of the development of the ML-based feature.

3. **Feature maintenance team:** The members of the team that are involved with the maintenance or troubleshooting of the ML-based feature. They may or may not belong to the team or organization that initially developed the feature.

4. **Remaining employees at the feature development organization:** The employees at the feature development organization may not have interacted or been involved with the ML-based feature but may have some knowledge or motive to harm it.

5. **Outsourced teams by operator:** Sometimes, the operators outsource some of the tasks to an external team or organization, which can eventually gain some knowledge about the involved ML-based features.

6. **3rd Party malicious actors:** This group of threat actors refers to all other external threat actors who may have some motive to harm the system.

These attackers may have different motives to harm the ML-based feature. It can be for economic benefits or just for pleasure. To compromise or disrupt the feature, they will aim to achieve one of the goals discussed in section 5.2.2, i.e., AG-01 Availability Breakdown, AG-02 Integrity Violation or AG-03 Privacy compromise.

In the next step, we try to identify what level of knowledge these different groups of attackers have. An attacker with white-box knowledge can craft some attacks which will be impossible for an attacker with only black-box knowledge. This is summarized in the table 6 according to the three knowledge levels discussed in section 5.2.4, viz., AK-01 White-box, AK-02 Black-box and AK-03 Grey-box. This can vary depending on the feature and its implementation.

Attacker	AK-01	AK-02	AK-03
Feature Development team	●	○	●
Feature Maintenance team	○	○	●
Remaining Employees at feature org.	○	○	●
3rd party malicious actor	○	●	○
Outsourced teams by operator	○	●	○
Operator's internal teams	○	●	○

Table 6: Attacker Knowledge

We also need to analyze the capabilities that these attackers possess. These capabilities enable the attackers to craft a specific attack. For example, an attacker with no access to data at any stage but with access to the source code will not be able to craft

any attack involving data but it can perform an attack which involves manipulating the source code or algorithm. This is summarized in the table 7 according to the five different capabilities discussed in section 5.2.3, viz., AC-01 Training Data Control, AC-02 Model control, AC-03 Testing Data Control, AC-04 Source Code Control AC-05 Query Access.

Attacker	AC-01	AC-02	AC-03	AC-04	AC-05
Feature Development team	○	●	○	●	○
Feature Maintenance team	●	●	●	●	●
Remaining Employees at feature org.	○	○	○	●	○
3rd party malicious actor	○	○	○	●	○
Outsourced teams by operator	●	○	●	○	●
Operator's internal teams	●	○	●	○	●

Table 7: Attacker Capabilities

Some of the attacker groups here have more than one capability and this is done keeping in mind different possibilities. In the real world, one individual attacker from that particular attacker group will have one or a combination of possible capabilities for that attacker group.

7.2.3 Identifying Threats

Here, we evaluate each attack from the attack library against their possibility and the knowledge and capabilities required to craft them. These attacks can be further backtracked to the potential attacker group based on the required knowledge and capability.

- **T-01 Optimization based methods**
  Adversarial samples are at a small distance from the original testing samples, the attack can be targeted or untargeted. The perturbations are done to a small part of the original data set. This can be done by an attacker, who has access to the data collected by the data collection engine inside the base station. The attacker can attempt to introduce an adversarial example initially and work towards optimizing it to the desired state based on the feedback from the output generated.
  Attacker group: TA-02 (Access inside the base station)
  Attacker Knowledge: AK-01,AK-03 (White-box or Grey-box)

- **T-02 Universal Evasion attacks**
  This differs from optimization based attacks as it induces small universal perturbations, i.e., changes to the entire data set, to induce a misclassification. The attacker with access to the collected data inside the base station may attempt to modify all the collected data points with the same changes.
  Attacker group: TA-02 (Access inside the base station)

Attacker Knowledge: AK-01, AK-03 (White-box or Grey-box)

- **T-06 Transferability of Attacks** The attacker creates the attack on a similarly working model and deploys it on the target model hoping for a desired result. For this, the attacker needs to have some knowledge about the target model, such as the data it is working with or the underlying base algorithm. Since even limited knowledge can enable the attacker to craft an attack, this attack can be categorized as a grey-box attack.
  Attacker group: TA-02 (Access inside the base station)
  Attacker Knowledge: AK-03 (Grey-box)

- **T-07 Availability Poisoning** The goal of the attacker here is to make the ML system unavailable for normal use. It can resemble a DDOS attack when the attacker sends a big influx of unrelated inputs to the ML system. It can also be done by exhausting the underlying resources. If such resources as CPU or other parts are shared between various services, the ML system can be made unavailable by compromising some other service which uses up all the available resources. For this type of attack, knowledge of the ML system is not necessary, and neither the access inside the base station required. An influx of input can be created by getting a large number of fake user equipment.
  Attacker group: TA-01 (No access inside the base station)
  Attacker Knowledge: AK-02 (Black-box)

- **T-08 Backdoor Poisoning**
  The attacker needs access to the training and input data during different stages of the ML life cycle. A trigger or a backdoor pattern is hidden or blended in the training data, which is eventually exploited once the model is deployed by sending an input which will activate the trigger. Since access to both the training and input data is required for this attack, it can be crafted by an attacker group which has access inside the base station and white-box knowledge. Though, comparatively much more difficult, an attacker with no access inside the base station or the collected data can also attempt to craft this attack as the data source such as user-equipments can be compromised and controlled by the attacker.
  Attacker group: TA-02 (Access inside the base station)
  Attacker Knowledge: AK-01 (White-box)

- **T-09 Targeted Poisoning**
  The goal of the attacker here is to induce a change in a small number of targeted samples. This type of attack may need multiple iterations to be effective, as the attacker will tune the changes to the input data by observing the effect of the previous change in the output. Thus, the access to the output from the ML system is essential.

Attacker group: TA-02 (Access inside the base station)
Attacker Knowledge: AK-01 (White-box)

- **T-10 Model Poisoning**
  This attack doesn't involve the access or manipulation of the data, rather than the model itself. The attacker with access to the model attempts to modify the trained model or inject some malicious functionality into it. Attacker group: TA-02 (Access inside the base station) Attacker Knowledge: AK-01 (White-box)

- **T-13 Model Extraction**
  The attacker attempts to train a surrogate model to work in the same way as the target model. It can train the surrogate model using the same or similar data used to train the original model, and thus the attacker group here will be a person who has access to both input and output data. This setup can resemble passive query access since the attacker doesn't directly interact with the model as opposed to active query access where the attacker queries the original model and uses that input and output information to train the surrogate model. This can also be a stepping stone to craft other attacks.
  Attacker group: TA-02 (Access inside the base station)
  Attacker Knowledge: AK-03 (Grey-box)

- **T-14 Property Inference**
  The attacker tries to infer some global information, such as demographic information, which can affect the working of the ML model. For this, the attacker would need to observe the output. Though not easy for an attacker group with no access inside the base station, it can still be possible if the output from the ML model results in some configuration changes in the attacker's user equipment.
  Attacker group: TA-01 (No access inside the base station)
  Attacker Knowledge: AK-02 (Black-box)

This can be summarised in a tabular format for easy reference. One important point to note is that the methods to initiate the attacks mentioned here are not the only method but a relevant one for the example feature in consideration.

7.2.4 Threats not considered

- **T-03 Physically realizable attacks**
  Physical factors to induce a misclassification, are not relevant to RAN use cases.

- **T-04 Score-based attacks**
  Adversarial examples are created based on the model's confidence score/logits.

Attack	Affected Asset	Possible Attacker Group	Attacker Knowledge
T-01	Data	TA-02	AK-01,AK-03
T-02	Data	TA-02	AK-01,AK-03
T-03	NA	NA	NA
T-04	NA	NA	NA
T-05	NA	NA	NA
T-06	Model	TA-02	AK-03
T-07	Environment	TA-01	AK-02
T-08	Data, Model	TA-01,TA-02	AK-01,AK-03
T-09	Data, Model	TA-02	AK-01
T-10	Model	TA-02	AK-01
T-11	NA	NA	NA
T-12	NA	NA	NA
T-13	Model	TA-02	AK-03
T-14	Data	TA-01	AK-02

Table 8: Threats Mapped to affected assets and attacker

- **T-05 Decision-based attacks**
  Attackers obtain only the final predicted labels of the model

  For threat actors with black-box knowledge of the ML model, in the RAN environment their access to required resources is restricted, hence creating a black-box evasion attack for them is not possible. Thus, T-04 and T-05 were not considered to be relevant.

- **T-11 Data Reconstruction**
  Recover an individual's data from released aggregate information. This is not relevant for RAN use cases as it requires active query access.

- **T-12 Membership Inference**
  Determine if a particular record or data sample was part of the training set. This is not possible for TA-01 as it doesn't have access to direct outputs from the ML model, and TA-02 already has access to the data, thus no intention for this attack.

7.3 Result and Analysis

We evaluate our method using the requirements set out in Section 3.3.

7.3.1 Classification Structure

The asset categorization will not be the same for every feature in consideration and hence, it cannot be overlooked. The attacker groups were also classified according to

their goals, knowledge and capabilities based on all the possibilities. In a practical scenario, the attacker can have these capabilities and knowledge in various combinations. The goals of the attacker were not considered to have a significant impact while identifying the attacks as it doesn't impact the possibility of an attack, but the probability of the attack. The effort required to deploy any attack by a specific attacker group against the benefit the attacker can get from it can be analyzed further to determine the probability of the attack. This wouldn't change the threat model but it can help in determining the severity and probability of any attack and also the selection of countermeasures.

7.3.2 Flexibility and Adaptability

The framework's adaptability was demonstrated by its ability to assess the feature without requiring any extensive reconfiguration. However, if the need arises, it is flexible enough to allow adjustments without changing the underlying structure for different features. The attack library can be easily expanded by adding emerging threats or more categories besides the evasion, poisoning and privacy attacks. The attacker attributes can also be expanded to include more capabilities as and when they emerge. The indexing of these attributes would help to maintain the structure and any future expansion of the framework would not affect the existing threat model.

7.3.3 Simplicity and Scalability

The framework by itself was easy to understand for all the stakeholders. It went through a review process where stakeholders from multiple domains and non-specialists in security or ML were able to understand the framework as well as the specialists. Some of the attacks needed detailed explanations for the stakeholders not very well-versed in the AI/ML domain. The Scalability could not be evaluated at this point, but new features in future can be analyzed in a similar way where the threats affecting them can be mapped to the affected asset and potential attacker subgroup and their attributes.

7.4 Limitation Analysis

1. The framework in its current state is only applicable to the features using predictive ML algorithms. Expanding this framework to cover features based on generative ML algorithms will need further work around analyzing and mapping the attacks that affect generative ML in the context of RAN. Thus this framework doesn't cover all the ML use cases.

2. The example feature analyzed using the framework was deployed in a physical base station environment. Though this framework was good for evaluating such a feature, it may not be complete enough if a similar feature is deployed in a cloud-based environment. The use of a cloud environment will introduce new vulnerable points which don't exist in the classical RAN and the method to craft an attack can also be different than those discussed in this thesis.

3. The counter strategies are just a suggestion as a part of this framework. Their selection and implementation would need further cost-benefit analysis of the attacks from which the system has to be protected as well as pros and cons of the counter strategies.

7.5 Summary

The proposed threat modeling framework was successfully used in identifying and mapping the security risks associated with AI/ML based RAN feature. The counter strategies to mitigate such risks could only be suggested and not selected due to the further need for cost-benefit analysis. The framework was found to be easily adaptable, as well as flexible while also being user-friendly and easy to understand for most of the stakeholders. Despite some limitations in conducting a complete threat assessment, the framework offered a solid foundation for securing AI/ML-based features in RAN.

8 Related Works

AI/ML-based systems are being developed and widely deployed across the globe, leading to the integration of these services into our everyday life. Research about securing AI/ML systems is an ever-evolving area of interest. As a new and evolving technology with widespread use cases, it also comes with new vulnerabilities and never seen before attacks.

Recent research by Koball et al. explored the security vulnerabilities specific to ML systems and proposed a novel threat modeling and classification framework focused on the evasion, extraction and poisoning attacks [51]. Mcgraw et al. focused on building a taxonomy of known ML attacks and building a threat model for the ML systems [52]. They categorized the attacks on ML system into two broad categories - manipulation attacks and extraction attacks, affecting three asset groups i.e., input, training data and model leading to six attack categories.

While classifying the risks on ML systems, works by NIST also made a significant attempt towards using a standard taxonomy for the attacks [13]. Marshall et al. at Microsoft published a comprehensive work on threat modeling of ML systems along with suggested mitigation strategy for each of the identified attacks and the expected impact severity of the same [53]. The ML Top 10 list published by OWASP tracks the current trending attacks associated with the ML [16]. These research only focus on evaluating the security around ML systems, which is not directly applicable to the scenarios where ML is integrated into RAN.

Along with addressing security risks for ML systems in general, there has been some research analyzing specific use cases of ML in mobile networks. Alatwi and Morisset worked on threat modeling for ML based Network Intrusion Detection System using an attack tree and STRIDE [54]. Soltani et al. examines the role of ML in the O-RAN architecture within the context of 6G networks and highlights the security risks introduced by AI-enabled functionalities [55]. Karaçay et al. explores the AI/ML-enabled "All-Senses Meeting" use case in 6G, identifying unique threats introduced by AI/ML components in holographic telepresence meetings, such as tampering with haptic signals and adversarial attacks [17]. These works focus on very specific use cases which are not very useful to address the security of ML use in RAN.

Despite the active research in this domain, there has not been any significant work for studying the security aspect of AI/ML use in RAN in general. The available research addresses the security of ML systems or some very specific use case in the context of mobile networks. Works by Abdalla et al. demonstrate that the existing RAN has been evaluated in depth for its security [56] but such work has not been done for the ML-integrated RAN. With the advancement to 6G and increasing use of ML in RAN, there is a need to have a more comprehensive security evaluation of such systems. This thesis work is a stepping stone to bridge the gap between the security evaluation of ML systems and the security evaluation of their use in the context of RAN.

9 Discussion

The threat modeling framework proposed by this thesis study considers the differences when the ML system is used in RAN context. This makes it more suitable to address the security vulnerabilities of ML used in RAN compared to a threat modeling for a generic ML system and serves as a bridge between them.

The threats defined by NIST and others had to be re-assessed in the context of RAN. They were redefined in terms of how they can be crafted while keeping the overall meaning and impact similar to the original definition for generic ML systems. One such example was interpreting an attacker's capability of query access to be of two types - active and passive, where the passive query access is more likely to be available to the attacker in RAN.

The proposed framework, while being complete in itself to perform threat modeling of a ML based features in RAN, is not rigid. It is flexible and can be expanded to include more threats or attacker groups when they are identified or depending on the need of the feature being evaluated. The use of this framework can help different stakeholders involved in the development and implementation of such features to properly assess and understand the security landscape around it and select the countermeasures. It will enable them to integrate ML features more securely, ensuring that new capabilities do not introduce vulnerabilities in critical network infrastructures.

The framework in its current form was applied to only one example feature for threat modeling. The current availability of features that could be fully or partially evaluated during this study also influenced some restrictions as a part of the design itself. The scope was limited to features being implemented in a classical base station environment and focused only on the use of predictive AI/ML-based features. This also leads to future research that will assess the applicability of the threat modeling framework in cloud-based RAN and the use of generative AI/ML models.

The non-availability of more diverse features for evaluation during this study doesn't necessarily limit the framework due to its comprehensive and flexible nature. Evaluation of the features implemented according to the RAN Intelligence design proposed by 3GPP [7] would be similar to the evaluation of the feature that was considered in this study.

This study is an attempt to fill a significant gap in the literature by providing a comprehensive threat modeling framework tailored for the ML based features in mobile networks, which will enable more secure development and deployment of future ML based features in RAN. As mobile networks move towards 6G, which is supposed to heavily adopt ML based features, the proposed framework and its future iterations will enable the stakeholders involved to ensure that the integration of such features is secure and protected.

10 Conclusion

In this thesis, a threat modeling framework for ML based features in RAN was proposed to assess the security challenges introduced by the integration of AI/ML in mobile networks. The framework is based on the threat modeling guidelines by OWASP while incorporating additional guidelines from LINDDUN and an attack library. The attack library is based on the existing attacks for AI/ML systems but their applicability has been redefined in the context of RAN.

This framework provides a practical and structured tool for evaluating a ML based feature in RAN which is deployed in a classical base station and uses a predictive algorithm. This fills a critical gap in the existing literature about the security of ML integration in mobile networks, where the traditional threat model for generic ML systems or RAN proves to be insufficient.

While the proposed framework focused on predictive algorithms and a classical base station deployment, it is designed with future adaptability in mind. The scope can be expanded to include the aspects of generative algorithms and cloud-based RAN environments in future research, as these technologies will become more relevant with the next generation of mobile networks. The flexibility of the framework allows it to be expanded without the need to redo the threat assessments already done using the current version.

This thesis contributes to both the academic and practical fields by offering a robust threat modeling framework for AI/ML features in mobile networks. The proposed framework can be easily used by mobile network operators and developers to assess and understand the security risks, paving a foundation for a safer integration of AI/ML-based technologies in RAN.

However, as with any research, this thesis is also bound by some limitations. The framework was applied to only an example feature based on a predictive ML algorithm, and deployed in the physical, i.e., non virtualized base station. Future research should explore the implications of deployments in cloud-based RANs and the use of generative ML algorithms. Additionally, real-world use and testing will further validate the framework's efficacy and highlight the areas for refinement.

To conclude, as AI/ML continues to play an important role in the advancement of mobile networks, the need for a robust security assessment of these technologies becomes increasingly critical. This thesis is a small contribution towards safe and resilient development and integration of AI/ML in RAN. Future research and development will build on this foundation to ensure the security of the evolving landscape of continuously evolving mobile networks.

References

[1] B. Marr and M. Ward, *Artificial Intelligence in Practice: How 50 Successful Companies Used AI and Machine Learning to Solve Problems*. Wiley, 2019.

[2] G. Rani, J. Singh, and A. Khanna, "Comparative analysis of generative ai models," in *International Conference on Advances in Computation, Communication and Information Technology (ICAICCIT)*, pp. 760–765, 2023.

[3] C. Huyen, *Designing Machine Learning Systems*. USA: O'Reilly Media, 2022.

[4] H. Holma, A. Toskala, T. Nakamura, and T. Uitto, "Introduction," in *5G Technology: 3GPP Evolution to 5G-Advanced*, pp. 1–11, Wiley, 2024.

[5] A. Toskala and M. Poikselkä, "5g architecture," in *5G Technology: 3GPP Evolution to 5G-Advanced*, pp. 67–86, Wiley, 2024.

[6] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network slicing and softwarization: A survey on principles, enabling technologies, and solutions," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2429–2453, 2018.

[7] 3GPP, "3GPP TR 37.817 Study on enhancement for data collection for NR and ENDC." https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3817, 2022. [Accessed: 2024-07-16].

[8] S. Hernan, S. Lambert, T. Ostwald, and A. Shostack, "Threat modeling-uncover security design flaws using the stride approach," *MSDN Magazine*, pp. 68–75, 01 2006.

[9] A. Shostack, *Threat Modeling: Designing for Security*. Wiley, 2014.

[10] M. Deng, K. Wuyts, R. Scandariato, B. Preneel, and W. Joosen, "A privacy threat analysis framework: supporting the elicitation and fulfillment of privacy requirements," *Requirements Engineering*, vol. 16, no. 1, pp. 3–32, 2011.

[11] K. Wuyts, L. Sion, and W. Joosen, "Linddun go: A lightweight approach to privacy threat modeling," in *IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pp. 302–309, 2020.

[12] "Threat Modeling | OWASP Foundation — owasp.org." https://owasp.org/www-community/Threat_Modeling. [Accessed 04-09-2024].

[13] A. Vassilev, A. Oprea, A. Fordyce, and H. Andersen, "Adversarial machine learning: A taxonomy and terminology of attacks and mitigations." https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=957080, 2024.

[14] B. E. Strom, A. Applebaum, D. Miller, K. C. Nickels, A. G. Pennington, and C. Thomas, "Mitre att&ck ® : Design and philosophy," in *MITRE ATT&CK ® : Design and Philosophy*, 2020.

[15] MITRE, "MITRE ATLAS." https://atlas.mitre.org/. [Accessed 05-09-2024].

[16] OWASP Foundation, "OWASP Machine Learning Top 10." https://mltop10.info/, 2023. [Accessed: 2024-07-16].

[17] L. Karaçay, Z. Laaroussi, S. Ujjwal, and E. U. Soykan, "On the security of 6g use cases: Ai/ml-specific threat modeling of all-senses meeting," in *International Conference on 6G Networking (6GNet)*, pp. 1–8, 2023.

[18] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *International Conference on Learning Representations*, 2014.

[19] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli, *Evasion Attacks against Machine Learning at Test Time*, pp. 387–402. Springer Berlin Heidelberg, 2013.

[20] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 86–94, 2017.

[21] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, "Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, (New York, NY, USA), pp. 1528–1540, Association for Computing Machinery, 2016.

[22] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, "Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models," in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, CCS '17, ACM, Nov. 2017.

[23] S. Moon, G. An, and H. O. Song, "Parsimonious black-box adversarial attacks via efficient combinatorial optimization," in *Proceedings of the 36th International Conference on Machine Learning, ICML* (K. Chaudhuri and R. Salakhutdinov, eds.), vol. 97 of *Proceedings of Machine Learning Research*, pp. 4636–4645, PMLR, 2019.

[24] W. Brendel, J. Rauber, and M. Bethge, "Decision-based adversarial attacks: Reliable attacks against black-box machine learning models," *CoRR*, vol. abs/1712.04248, 2017.

[25] R. Perdisci, D. Dagon, W. Lee, P. Fogla, and M. Sharif, "Misleading worm signature generators using deliberate noise injection," in *IEEE Symposium on Security and Privacy (S&P'06)*, pp. 15–31, 2006.

[26] R. S. Siva Kumar, M. Nyström, J. Lambert, A. Marshall, M. Goertzel, A. Comissoneru, M. Swann, and S. Xia, "Adversarial machine learning-industry perspectives," in *IEEE Security and Privacy Workshops (SPW)*, pp. 69–75, 2020.

[27] T. Gu, K. Liu, B. Dolan-Gavitt, and S. Garg, "Badnets: Evaluating backdooring attacks on deep neural networks," *IEEE Access*, vol. 7, pp. 47230–47244, 2019.

[28] E. Wenger, J. Passananti, A. N. Bhagoji, Y. Yao, H. Zheng, and B. Y. Zhao, "Backdoor attacks against deep learning systems in the physical world," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6202–6211, 2021.

[29] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, J. Zhai, W. Wang, and X. Zhang, "Trojaning attack on neural networks," in *Network and Distributed System Security Symposium*, 2018.

[30] I. Dinur and K. Nissim, "Revealing information while preserving privacy," in *Proceedings of the 22nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '03, (New York, NY, USA), pp. 202–210, Association for Computing Machinery, 2003.

[31] S. Garfinkel, J. M. Abowd, and C. Martindale, "Understanding database reconstruction attacks on public data," *Commun. ACM*, vol. 62, pp. 46–53, feb 2019.

[32] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning (still) requires rethinking generalization," *Commun. ACM*, vol. 64, pp. 107–115, feb 2021.

[33] F. Tramer, F. Zhang, A. Juels, M. Reiter, and T. Ristenpart, "Stealing machine learning models via prediction apis," in *Stealing Machine Learning Models via Prediction APIs*, 08 2016.

[34] M. Jagielski, N. Carlini, D. Berthelot, A. Kurakin, and N. Papernot, "High accuracy and high fidelity extraction of neural networks," in *Proceedings of the 29th USENIX Conference on Security Symposium*, SEC'20, (USA), USENIX Association, 2020.

[35] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *International Conference on Learning Representations* (Y. Bengio and Y. LeCun, eds.), 2015.

[36] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *CoRR*, vol. abs/1706.06083, 2017.

[37] D. Tsipras, S. Santurkar, L. Engstrom, A. Turner, and A. Madry, "Robustness may be at odds with accuracy," in *International Conference on Learning Representations, publisher = OpenReview.net, year = 2019, url = https://openreview.net/forum?id=SyxAb30cY7, timestamp = Thu, 25 Jul 2019 14:26:02 +0200, biburl = https://dblp.org/rec/conf/iclr/TsiprasSETM19.bib, bibsource = dblp computer science bibliography, https://dblp.org*.

[38] G. Katz, C. W. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer, "Reluplex: An efficient SMT solver for verifying deep neural networks," in *Computer Aided Verification - 29th International Conference* (R. Majumdar and V. Kuncak, eds.), vol. 10426 of *Lecture Notes in Computer Science*, pp. 97–117, Springer, 2017.

[39] T. Gehr, M. Mirman, D. Drachsler-Cohen, P. Tsankov, S. Chaudhuri, and M. Vechev, "Ai2: Safety and robustness certification of neural networks with abstract interpretation," in *IEEE Symposium on Security and Privacy (SP)*, pp. 3–18, 2018.

[40] B. Nelson, M. Barreno, F. J. Chi, A. D. Joseph, B. I. Rubinstein, U. Saini, C. Sutton, and K. Xia, "Exploiting machine learning to subvert your spam filter," in *First USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET 08)*, (San Francisco, CA), USENIX Association, Apr. 2008.

[41] B. Biggio, I. Corona, G. Fumera, G. Giacinto, and F. Roli, "Bagging classifiers for fighting poisoning attacks in adversarial classification tasks," in *Proceedings of the 10th International Conference on Multiple Classifier Systems*, MCS'11, (Berlin, Heidelberg), pp. 350–359, Springer-Verlag, 2011.

[42] C. Dwork, "Differential privacy," in *Proceedings of the 33rd International Conference on Automata, Languages and Programming - Volume Part II*, ICALP'06, (Berlin, Heidelberg), pp. 1–12, Springer-Verlag, 2006.

[43] A. Thudi, I. Shumailov, F. Boenisch, and N. Papernot, "Bounding membership inference," *CoRR*, vol. abs/2202.12232, 2022.

[44] S. Mahloujifar, E. Ghosh, and M. Chase, "Property inference from poisoning," in *IEEE Symposium on Security and Privacy SP*, pp. 1120–1137, IEEE, 2022.

[45] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao, "Neural cleanse: Identifying and mitigating backdoor attacks in neural networks," in *IEEE Symposium on Security and Privacy (SP)*, pp. 707–723, 2019.

[46] Z. Xiang, D. J. Miller, and G. Kesidis, "Post-training detection of backdoor attacks for two-class and multi-attack scenarios," *CoRR*, vol. abs/2201.08474, 2022.

[47] G. Shen, Y. Liu, G. Tao, S. An, Q. Xu, S. Cheng, S. Ma, and X. Zhang, "Backdoor scanning for deep neural networks through k-arm optimization," in *Proceedings of the 38th International Conference on Machine Learning* (M. Meila and T. Zhang,

eds.), vol. 139 of *Proceedings of Machine Learning Research*, pp. 9525–9536, PMLR, 2021.

[48] X. Huang, M. Alzantot, and M. B. Srivastava, "Neuroninspect: Detecting backdoors in neural networks via output explanations," *CoRR*, vol. abs/1911.07399, 2019.

[49] H. Chen, C. Fu, J. Zhao, and F. Koushanfar, "Deepinspect: a black-box trojan detection and mitigation framework for deep neural networks," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, IJCAI'19, pp. 4658–4664, AAAI Press, 2019.

[50] X. Xu, Q. Wang, H. Li, N. Borisov, C. A. Gunter, and B. Li, "Detecting ai trojans using meta neural analysis," in *IEEE Symposium on Security and Privacy (SP)*, pp. 103–120, 2021.

[51] C. Koball, Y. Wang, B. P. Rimal, and V. Vaidyan, "Machine learning security: Threat model, attacks, and challenges," *Computer*, vol. 57, no. 10, pp. 26–35, 2024.

[52] G. Mcgraw, R. Bonett, H. Figueroa, and V. Shepardson, "Security engineering for machine learning," *Computer*, vol. 52, no. 8, pp. 54–57, 2019.

[53] A. Marshall, J. Parikh, E. Kiciman, and R. S. S. Kumar, "Threat Modeling AI/ML Systems and Dependencies." https://learn.microsoft.com/en-us/security/engineering/threat-modeling-aiml. [Accessed 08-10-2024].

[54] H. Ali Alatwi and C. Morisset, "Threat modeling for machine learning-based network intrusion detection systems," in *IEEE International Conference on Big Data*, pp. 4226–4235, 2022.

[55] S. Soltani, M. Shojafar, R. Taheri, and R. Tafazolli, "Can open and ai-enabled 6g ran be secured?," *IEEE Consumer Electronics Magazine*, vol. 11, no. 6, pp. 11–12, 2022.

[56] A. S. Abdalla and V. Marojevic, "End-to-end o-ran security architecture, threat surface, coverage, and the case of the open fronthaul," *IEEE Communications Standards Magazine*, vol. 8, no. 1, pp. 36–43, 2024.