TIMESAFE: Timing Interruption Monitoring and Security Assessment for Fronthaul Environments.

JOSHUA GROEN, Electrical and Computer Engineering, Northeastern University, Boston, United States and Electrical Engineering and Computer Science, United States Military Academy, West Point, USA
SIMONE DIVALERIO, University of Rome La Sapienza, Rome, Italy
IMTIAZ KARIM, Purdue University, West Lafayette, United States
DAVIDE VILLA, Northeastern University, Boston, United States
YIWEI ZHANG, Purdue University, West Lafayette, United States
LEONARDO BONATI, Northeastern University, Boston, United States
MICHELE POLESE, Northeastern University, Boston, United States
SALVO D'ORO, Northeastern University, Boston, United States
TOMMASO MELODIA, Northeastern University, Boston, United States
ELISA BERTINO, Department of Computer Sciences, Purdue University, West Lafayette, United States
FRANCESCA CUOMO, University of Rome La Sapienza, Rome, Italy
KAUSHIK CHOWDHURY, The University of Texas at Austin, Austin, United States

5G and beyond cellular systems embrace the disaggregation of Radio Access Network (RAN) components, exemplified by the evolution of the fronthaul (FH) connection between cellular baseband and radio unit equipment. Crucially, synchronization over the FH is pivotal for reliable 5G services. In recent years, there has been a push to move these links to an Ethernet-based packet network topology, leveraging existing standards and ongoing research for Time-Sensitive Networking (TSN). However, TSN standards, such as Precision Time Protocol (PTP), focus on performance with little to no concern for security. This increases the exposure of the open FH to security risks. Attacks targeting synchronization mechanisms pose significant threats, potentially disrupting 5G networks and impairing connectivity.

This article is based upon work partially supported by the U.S. National Science Foundation under grants CNS-1925601 and CNS-2112471 and by SERICS (PE00000014) 5GSec project under the MUR National Recovery and Resilience Plan funded by the European Union - NextGenerationEU. The opinions in the work are solely of the authors and do not reflect those of the the U.S. Military Academy, U.S. Army, or the Department of Defense.
Authors' Contact Information: Joshua Groen (corresponding author), Electrical and Computer Engineering, Northeastern University, Boston, MA, USA, Electrical Engineering and Computer Science and United States Military Academy, West Point, NY, USA; e-mail: groen.j@northeastern.edu; Simone DiValerio, University of Rome La Sapienza, Rome, Lazio, Italy; e-mail: divalerio.1835412@studenti.uniroma1.it; Imtiaz Karim, Purdue University, West Lafayette, IN, USA; e-mail: karim7@purdue.edu; Davide Villa, Northeastern University, Boston, MA, USA; e-mail: villa.d@northeastern.edu; Yiwei Zhang, Purdue University,West Lafayette, IN, USA; e-mail: zhan4630@purdue.edu; Leonardo Bonati, Northeastern University, Boston, MA, USA; e-mail: l.bonati@northeastern.edu; Michele Polese, Northeastern University, Boston, MA, USA; e-mail: m.polese@northeastern.edu; Salvo D'Oro, Northeastern University, Boston, MA, USA; e-mail: s.doro@northeastern.edu; Tommaso Melodia, Northeastern University, Boston, MA, USA; e-mail: t.melodia@northeastern.edu; Elisa Bertino, Department of Computer Sciences, Purdue University, West Lafayette, IN, USA; e-mail: bertino@purdue.edu; Francesca Cuomo, University of Rome La Sapienza, Rome, Lazio, Italy; e-mail: francesca.cuomo@uniroma1.it; Kaushik Chowdhury, The University of Texas at Austin, Austin, TX, USA; e-mail: kaushik@utexas.edu.

ACM 2471-2566/2025/12-ART7
https://doi.org/10.1145/3775060

In this article, we demonstrate the impact of successful spoofing and replay attacks against PTP synchronization. We show how a spoofing attack is able to cause a production-ready O-RAN and 5G-compliant private cellular base station to catastrophically fail within 2 seconds of the attack, necessitating manual intervention to restore full network operations. To counter this, we design a Machine Learning (ML)-based monitoring solution capable of detecting various malicious attacks with over 97.5% accuracy.

CCS Concepts: • **Security and privacy → Mobile and wireless security**; **Intrusion detection systems**; **Distributed systems security**

Additional Key Words and Phrases: ORAN, PTP, ML, open fronthaul S-plane, security

ACM Reference Format:
Joshua Groen, Simone DiValerio, Imtiaz Karim, Davide Villa, Yiwei Zhang, Leonardo Bonati, Michele Polese, Salvo D'Oro, Tommaso Melodia, Elisa Bertino, Francesca Cuomo, and Kaushik Chowdhury. 2025. TIMESAFE: Timing Interruption Monitoring and Security Assessment for Fronthaul Environments.. *ACM Trans. Priv. Sec.* 29, 1, Article 7 (December 2025), 30 pages. https://doi.org/10.1145/3775060

1 Introduction

Recent advancements in 5G and beyond have focused on disaggregating the **Next Generation Node Base (gNB)** and core networks, leading to the evolution of more open and modular systems. As a result, the **fronthaul (FH)** has emerged as a critical component, enabling the separation of radio elements from baseband processing. Disaggregated FH deployments were originally based on the **Common Public Radio Interface (CPRI)**. While CPRI facilitated high-speed, centralized processing of radio signals, it lacked built-in security measures due to its design for single-vendor, co-located deployments [53]. The subsequent enhanced CPRI (eCPRI) introduced some security improvements, but significant gaps remain, particularly in synchronization [13]. The eCPRI-based O-RAN ALLIANCE's open FH now connects disaggregated components from multiple vendors, often using switched Ethernet topologies [19, 41]. While Ethernet's flexibility supports diverse traffic types, it also exposes the FH to threats, as demonstrated by potential attacks on interconnected devices [8, 10, 15, 23, 50]. These attacks can disrupt synchronization and cause outages. For example, as shown in Figure 1, an attacker that gains access to the FH network can carry out spoofing attacks that cause synchronization drift, resulting in a reduction in throughput and eventually a complete crash of the base station. Although there are ongoing efforts by industry, government, and academia to address these security gaps [8, 11, 21, 38, 42, 53], no comprehensive standards for securing the FH have yet been established.

The open FH relies on precise timing through the **Synchronization Plane (S-plane)** to perform critical operations like OFDM synchronization, carrier aggregation, and handovers, which require time accuracy within tight margins [30, 35]. This synchronization depends on the **Precision Time Protocol (PTP)**, which, like the FH, was initially developed without any security mechanisms [49]. While recent updates to the PTP standard include additional security features [2, Annex P], these have not been widely implemented in **Open Radio Access Network (Open RAN)** environments, leaving vulnerabilities that attackers could exploit.

Motivation and Scope. Security is critical for the success and widespread adoption of any system, and Open RAN has been criticized for lacking robust security mechanisms. In response, the O-RAN ALLIANCE established WG11 [38] to define a security framework, including zero-trust architectures [3], procedures, and defense mechanisms essential for Open RAN's success [21]. While WG11 acknowledges that attacks on the S-plane can have a high impact, they consider the likelihood low due to the perceived sophistication required by an attacker [40]. However, a significant gap remains in analyzing security risks and proposing solutions for the open FH S-plane

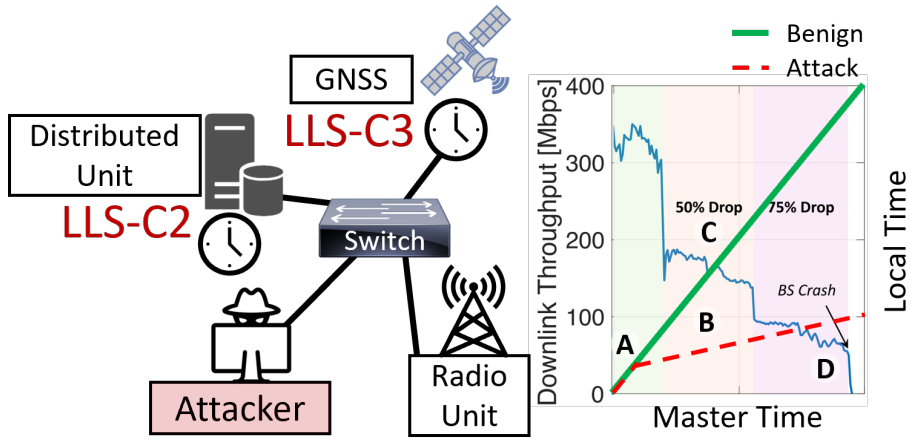


Fig. 1. The open fronthaul employs PTP to synchronize the master clock with distributed base station components over a switched network. In this scenario, an attacker compromises a device on the network and, at time **A**, initiates a Spoofing Attack. This leads to gradual synchronization drift (**B**), first degrading performance (**C**), and then causing the base station to crash at time **D**.

in production-grade Open RAN environments. Many researchers lack access to realistic over-the-air systems for testing and validating both attacks and countermeasures. To address this, we developed a Digital Twin system that replicates our production-ready FH network environment, enabling rapid experimentation and analysis.

In this article, we aim to: *(i)* confirm the high impact of PTP attacks in a production-ready, 5G and O-RAN-compliant private cellular network; *(ii)* demonstrate that these attacks require relatively low sophistication to execute successfully; *(iii)* provide a Digital Twin framework for further analysis of attacks and development of solutions; and *(iv)* develop and evaluate detection mechanisms, showing that our **Machine Learning (ML)** model can effectively identify malicious PTP attacks with over 97.5% accuracy across diverse environments.

Contributions. Based on this motivation, we propose TIMESAFE: Timing Interruption Monitoring and Security Assessment for Fronthaul Environments, a comprehensive framework for assessing the impact and likelihood of attacks against PTP in the FH of Open RAN and 5G networks. TIMESAFE utilizes machine learning to enhance monitoring and provide highly accurate attack detection. The main contributions of our work include:

- **Experimental Analysis:** We conduct the first experimental analysis of the impact of timing attacks in a production-ready private cellular network (see Table 2), following O-RAN ALLIANCE procedures [39]. Our findings reveal that these attacks can lead to catastrophic outages of the gNB, highlighting critical vulnerabilities in PTP synchronization mechanisms.
- **Attack Demonstration:** We show that timing attacks on PTP in the FH S-plane are straightforward and require minimal sophistication, countering the belief that such attacks are highly complex.
- **Machine Learning Detection:** We develop a state-of-the-art transformer-based ML detection mechanism, achieving over 97.5% accuracy in detecting attacks on a deployed **Distributed Unit (DU)**.
- **Open Source and Dataset Contributions:** We open source our framework, including our Digital Twin design, the attacks, ML models, automation scripts, and a `.pcap` traffic trace dataset to support further research and advancement in O-RAN security [5].

Table 1. Glossary of Commonly Used Acronyms

Acronym	Definition
PTP	Precision Time Protocol
FH	Fronthaul
DU	Distributed Unit
RU	Radio Unit
ML	Machine Learning
O-RAN	Open Radio Access Network
S-plane	Synchronization Plane
UE	User Equipment (e.g., mobile phone)
gNB	Next Generation Node B (5G base station)
BMCA	Best Master Clock Algorithm
LLS-C	Lower Layer Split Control Plane

Responsible Disclosure. We have disclosed the identified attacks to the O-RAN ALLIANCE Working Group 11 (O-RAN-CVD-002, O-RAN-CVD-003). Our commitment extends beyond this; we are dedicated to collaborating with regulatory organizations and product vendors to enhance security. Through this proactive approach, we aim to ensure the robustness and resilience of the FH against future threats.

The rest of the article is organized as follows. In Section 2, we provide background on the S-plane and PTP, while Section 3 surveys prior work. Next, we describe the threat model in Section 4 and our test beds in Section 5. We show the impact of the attacks in Section 6. Then, we demonstrate the need for ML based PTP monitoring tool in Section 7, describe our model in Section 8 and show state of the art results in Section 9. Finally, we provide additional insights and directions for research in Section 10, and draw our conclusions in Section 11.

2 Background

Traditional **Radio Access Network (RAN)** architectures are monolithic, leading to vendor lock-in and limited flexibility. The Open RAN paradigm addresses these issues with disaggregated, software-based components and open interfaces [45], based on the 3GPP 7.2 split between **Radio Unit (RU)** and DU. Current O-RAN/ETSI standards [19, 41] do not mandate encryption on the FH interface due to bandwidth and latency concerns [21], though some studies suggest using **Media Access Control Security (MACsec)** for added security [12, 15, 16]. We start by defining key acronyms in Table 1, followed by background on the S-plane, PTP, and security.

2.1 Fronthaul S-plane

The FH S-plane standards define four clock models and synchronization topologies based on the **Lower Layer Split Control Plane (LLS-C)**: LLS-C1, LLS-C2, LLS-C3, and LLS-C4 [41], as illustrated in Figure 2. Each of these profiles reflects a different method for distributing synchronization between the DU and RU, with varying complexity and exposure to attack.

In LLS-C1, the DU, and RU are directly connected, typically via a point-to-point Ethernet link. This straightforward setup uses standard PTP signaling between the two endpoints, often with the DU serving as the master and the RU as the slave. With no intermediate devices involved, there are fewer components to misconfigure or compromise, resulting in a relatively low risk of timing manipulation.

LLS-C4 similarly avoids reliance on the transport network by providing each device–both the DU and one or more RUs–with a local GNSS receiver. These receivers discipline the internal clocks

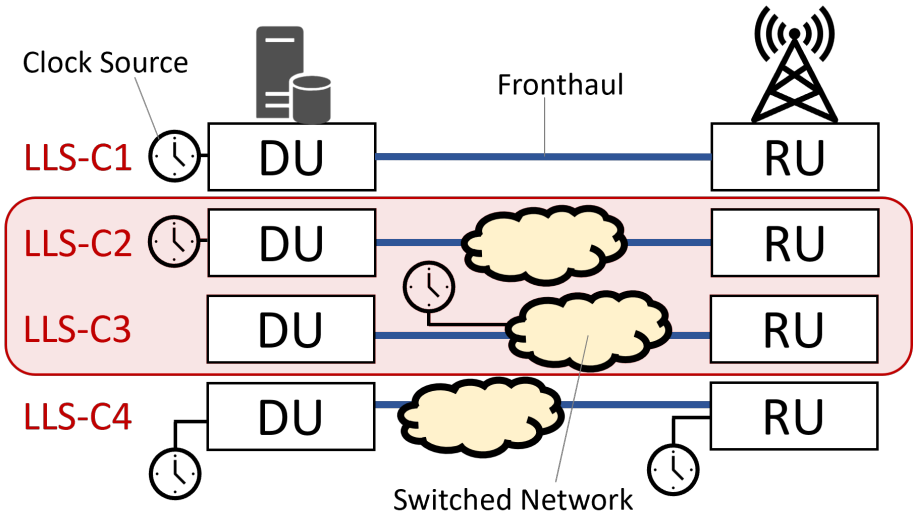


Fig. 2. The fronthaul connects the **Distributed Unit (DU)** to the **Radio Unit (RU)**. There are four synchronization plane topologies based on the location of the clock source and switched network. Any topology where the timing information of a single clock has to be synchronized via a packet switch network (e.g. LLS-C2 and LLS-C3) is especially vulnerable to PTP attacks because the switched network is used to transport the timing messages.

using GNSS signals and holdover algorithms, reducing the need for any in-band synchronization messages. This setup removes network-based attack vectors but introduces potential vulnerabilities to GNSS jamming and spoofing, which must be mitigated with resilient oscillator design and fallback mechanisms.

In contrast, LLS-C2 and LLS-C3 rely heavily on the transport network to carry synchronization information, making them far more vulnerable to attack. LLS-C2 places one or more Ethernet switches between the DU and RU, forming a chain in which each switch must function as a **Boundary Clock (BC)** or **Transparent Clock (TC)**. These intermediate devices forward or adjust PTP messages in real time, and their correct behavior is critical for accurate timing. Any misconfiguration, software bug, or compromise of these switches can introduce subtle timing errors or deliberate tampering–both of which may be difficult to detect. LLS-C3 is even more exposed: the DU and RU both receive timing from a central Grandmaster, such as a **Primary Reference Time Clocks (PRTC)**, via a shared switched network. Because neither endpoint provides timing to the other, the entire synchronization path runs through multiple switches and routers, each of which becomes a potential point of failure or compromise. Moreover, both C2 and C3 require careful configuration of PTP parameters, including delay mechanisms, priority levels, and clock class settings, and they depend on hardware timestamping and accurate **Best Master Clock (BMC)** selection algorithms to maintain time alignment. This increased complexity not only makes these profiles harder to secure and debug but also expands the attack surface significantly. In practice, any adversary capable of accessing the synchronization path—whether through compromised network hardware or malicious software—can degrade or hijack timing across the system, impacting radio coordination, scheduling, and service integrity [22, 40].

2.2 PTP Overview

The PTP standard (IEEE 1588 [2]) ensures precise network clock synchronization in the nanosecond range. It achieves this through hardware time stamping, which minimizes delays from the

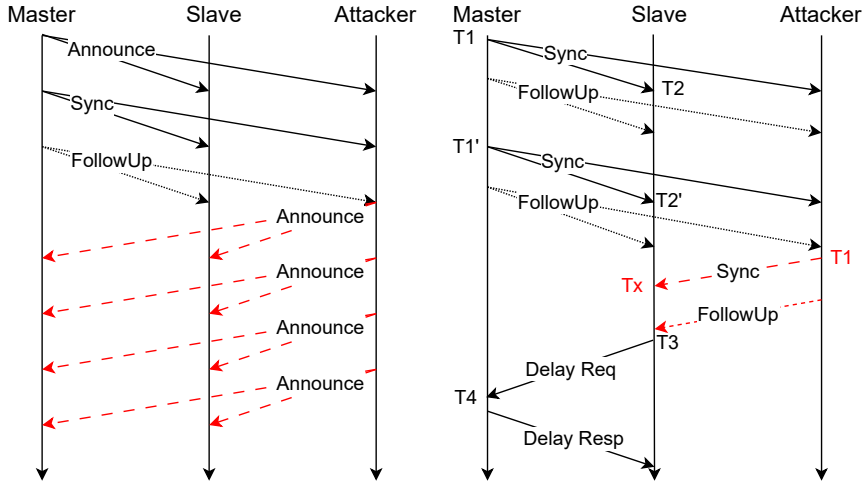


Fig. 3. The black (solid/dotted) lines represent normal traffic, while the red (dashed) on the left side illustrates the flow of messages during a spoofing attack, where an attacker manipulates the BMCA to become the master. The red (dashed) on the right side illustrates the flow of messages during a replay attack, showing the re-transmission of sync and FollowUp messages.

networking stack [47]. The process involves electing a master clock that synchronizes all slave clocks in the network through a series of exchanges and messages.

Best Master Clock Algorithm (BMCA). The initial step in PTP involves leader election through the BMCA. During this process, each clock periodically broadcasts `Announce` messages that contain up to nine attributes, which help determine the best clock for master selection. Key attributes include:

— *Priority1:* A configurable priority setting used as the first comparison feature in the selection process.
— *ClockClass:* Indicates the clock's traceability and suitability as a time source.
— *ClockAccuracy:* Reflects the clock's precision.
— *ClockIdentity:* A unique identifier for the clock, used to resolve ties.

Each clock evaluates the `Announce` messages based on the available attributes, selecting the clock with the highest quality as the master. This process is dynamic, allowing for continuous re-evaluation and re-selection if higher-quality clocks are introduced or current attributes change.

Time Synchronization Process. After being elected, the master clock begins the synchronization process with the slave clocks. *Frequency*, *Phase* and *Delay* are critical aspects that influence synchronization.

Frequency refers to the rate at which a clock oscillates. To adjust the frequency, the master clock sends a `Sync` message and takes a timestamp (T_1). There are two types of clocks: *OneStep* clocks, which insert the timestamp (T_1) directly into the `Sync` message, and *TwoStep* clocks, which send a `FollowUp` message containing the timestamp (T_1). In the *TwoStep* case, the initial `Sync` message has an estimated timestamp or a value of 0. When the slave node receives the `Sync` message, it takes a timestamp (T_2). No timestamp is taken when the `FollowUp` message is received.

This process is repeated periodically and new timestamps (T_1' and T_2') are taken again as shown in Figure 3. After receiving two `Sync` messages, the Slave can calculate the frequency difference relative to the Master clock, which is called $Drift = \frac{(T_2'-T_2)-(T_1'-T_1)}{T_1'-T_1}$. By calculating this drift, the Slave can adjust its own frequency to align with the Master clock, ensuring precise synchronization. To meet

all FH timing requirements, additional **Physical Layer Frequency Support (PLFS)** support, such as SyncE, is required. See Appendix (Section A.2) for more background on PLFS.

Delay refers to the time it takes for a message or signal to travel from one clock to another through the network. In general, delay can be computed via two mechanisms: (i) the *E2E* mechanism which measures the delay from the master to the slave; and (ii) the *P2P* mechanism which measures the delay between any two nodes irrespective of their role. E2E does not require intermediate nodes to be PTP aware, while P2P requires all intermediate switches to be PTP aware. In our experiments, we use the E2E mechanism, which is the default configuration. In this case, the master sends a `Sync` (and optional `FollowUp`) message and takes the timestamp (T_1). The slave receives the `Sync` message, takes the timestamp (T_2), sends a `DelayReq` and takes the timestamp (T_3). The master receives the `DelayReq` message and takes the timestamp (T_4) and stores it. Lastly, the master sends a `DelayResp` message with the timestamp (T_4). After receiving the `DelayResp` message from the master, the slave can measure the delay between the two nodes: $Delay = \frac{(T_4-T_1)-(T_3-T_2)}{2}$. This process, also shown in Figure 3, is repeated at defined intervals.

Phase refers to the relative alignment of the waveform or signal between two clocks. In PTP, phase synchronization ensures that the phase difference between the master clock and the slave clock is minimized, allowing them to maintain consistent timing. After calculating the delay, the slave can also measure the phase, also called $Offset = (T_2 - T_1) - Delay$.

2.3 PTP Security

The original PTP standard, first published over twenty years ago, did not include any security mechanisms [49]. Subsequent revisions added a security annex, updated in 2019, detailing a flexible, multipronged security approach [2].

- **Prong A** focuses on authentication through an optional **Type, Length, Value (TLV)** field, with key management occurring outside of PTP. Adding cryptographic functions increases computational cost and processing latency, requiring synchronization adjustments, particularly in resource-constrained environments like FPGA-based RUs.
- **Prong B** addresses external security mechanisms, including MACsec (Layer 2) and IPsec (Layer 3). However, O-RAN Alliance specifications [41] and ITU-T G.8275.1 [4] generally do not permit IP transport. Using MACsec poses challenges; software implementations create virtual interfaces that disrupt hardware timestamping, reducing accuracy significantly. Although hardware-based MACsec solutions have been proposed [17, 18], they increase costs, delay, and complexity, reducing the virtualization central to Open RAN.
- **Prong C** focuses on redundancy (e.g., multiple grandmasters and alternate paths) to mitigate attacks and enhance system resilience. While beneficial, these methods add cost and complexity, conflicting with the push for increased virtualization and decreased costs in Open RAN [45].
- **Prong D** emphasizes monitoring and management. Monitoring PTP performance can help detect potential security attacks [2]. The annex suggests parameters to monitor, but does not specify actions to take upon detecting threats. Given the costs associated with the other prongs, it is logical to implement these security measures only when an attack is detected.

3 Related Work

There is a growing awareness of the need to secure PTP. Several attacks are demonstrated in a data center context by DeCusatis et al. [14]. Itkin and Wool developed a detailed analysis of threats against PTP and proposed using an efficient elliptic-curve public-key signature for Prong A [26]. Shereen et al. [48] experimentally evaluated implementing Prong A, finding that software-based authentication adds at least an additional 70μs of delay. Similarly, Rezabek et al. [47] evaluated

Table 2. Overview of Prior Work on PTP Security Highlighting the Application Domain Studied, the Attacks (see Section 4.1) and Mitigations Implemented, and the Framework used for Validation

Paper	Domain	Attacks Implemented	Mitigation Implemented	Type of Validation
[14]	Data center	Spoofing, Comp. Master Clock	None	Experimental testbed
[26]	None	Spoofing, Comp. Master Clock	Prong A, Modifications to PTP	Experimental testbed
[48]	None	None	Prong A	Experimental testbed
[47]	None	None	Prong A	Experimental testbed
[32–34]	Smart Grid	Spoofing, Replay	Modifications to PTP	Formal model verification
[49]	IoT	Spoofing	Prong C, Modifications to PTP	IoT testbed
[20]	None	Spoofing	Prong C, Modifications to PTP	Experimental testbed
[30]	O-RAN	None	None	None
[15–18]	O-RAN	None	Prong B	Experimental testbed
TIMESAFE	O-RAN	Spoofing, Replay	Prong D	Production-ready network

Our work is the first to demonstrate the impact of attacks on an O-RAN production-ready environment as well as the first to implement a Prong D (monitoring and management) solution.

software-based authentication and concluded that there is visible degradation of clock synchronization for each hop in the network with standard deviations between 118 and 571ns.

There are several works that investigate security threats and propose related defense mechanisms for PTP in smart grids. Moussa et al. [32] proposed adding a new type of PTP clock and modifying the PTP slave functionality. Then in subsequent work, Moussa et al. proposed adding further message types to aid in detection and mitigation [33], and adding feedback from both slaves and master clocks to a reference entity [34].

A method targeting Prong C, where clock and network path redundancy are added, is proposed by Shi et al. [49]. Likewise, the approach by Finkenzeller et al. [20] uses redundant, or cyclic paths, to detect and mitigate time delay attacks.

Maamary et al. give an overview of several threats to the FH S-plane and discuss possible countermeasures [30], though they do not implement either attacks or countermeasures. FH security considerations are further described and MACsec is proposed as a solution by Dik et al. [15–18]. While these works are the most similar to ours, they primarily address Prong B and could be implemented in parallel to our work.

In contrast to the above works, our paper is the first to demonstrate successful attacks against PTP used for the S-plane causing the gNB to fail and demonstrating how O-RAN-connected devices can be leveraged as attack vectors. We also are the first to propose a ML-based method to monitor PTP in accordance with Prong D, successfully discriminating between benign and malicious traffic with over 97.5% accuracy.

4 Threat model

The attack scenario we consider for both the production-ready network and digital twin environment is shown in Figure 1. In this setup, a switched network connects a software-based DU, with robust

computational and storage capabilities (e.g., an edge server running an open-source network stack), to a more limited RU, such as an FPGA-based device. The complexity of the Open RAN architecture, with its many interconnected components, heightens security risks[36, 37]. Although the fronthaul network is protected by IEEE 802.1X, adversaries may bypass this security in a variety of ways.

- **Increased Remote Access.** With its open and multi-vendor architecture, O-RAN relies on a diverse range of suppliers, developers, and service providers, each with varying levels of access and control over networked components. As a result, the number of personnel who can potentially access sensitive infrastructure grows, increasing the risk of misconfigured or compromised authentication processes [37]. IBM's X-Force Threat Intelligence 2023 report identified misuse of valid credentials (36%) and plaintext credentials on endpoints (33%) as common initial access vectors in cloud environments [25]. Such vulnerabilities are particularly concerning in O-RAN, where distributed network functions and the involvement of multiple parties create more opportunities for configuration errors and unauthorized access to the DUs [8, 24].
- **Increased Physical Access.** The distributed nature of gNB edge servers, often located in accessible public spaces such as sidewalks, rooftops, or building basements, makes on-site physical access an increasingly viable entry point for attacks on the FH [55]. Physical proximity to fronthaul infrastructure lowers the barrier for potential attackers compared to centralized cloud data centers. Once on-site, an adversary can bypass 802.1X by inserting a rogue device (e.g., a mini PC) into the network to intercept, modify, or replay traffic over already authenticated connections.
- **Dependency and Supply Chain Vulnerabilities.** Software-based network functions in 5G, including those in Open RAN, are susceptible to supply chain attacks, particularly with the use of unvetted and globally sourced components [37]. A recent government report highlighted that supply chain risks from untrusted hardware and software vendors are significant threats, as adversarial actors could embed backdoors in either the DU or RU to manipulate fronthaul traffic without being detected by 802.1X [55]. Recent studies of several O-RAN software stacks (including those provided by the open-source projects in the O-RAN Software Community) have uncovered numerous high-risk dependencies, misconfigurations, and weak security practices, such as inadequate access control, lack of encryption, and poor secret management across multiple components [10, 23, 27, 50, 51].

The combination of increased remote access to DUs and physical access to the FH, along with dependency and supply chain vulnerabilities, makes gaining access to a device connected to the FH network not just plausible but increasingly likely [37]. Once an attacker gains access to a device on the switched network, they can observe, replay, or inject PTP broadcast packets sent through the switch.

4.1 Attack Methods

Here we highlight a few of the possible attacks that can compromise PTP synchronization.

- **Denial-of-Service (DoS) Attack.** A DoS attack floods the network with a high density of PTP messages or other types of traffic, overwhelming PTP nodes and preventing them from correctly processing timing messages.
- **Spoofing Attacks.** In a spoofing attack, an adversary sends malicious PTP messages to deceive slave clocks into accepting fake timing information. This method involves impersonating the master clock by sending fake Announce packets or sending false Sync, FollowUp, DelayReq, or DelayResp messages to disrupt legitimate synchronization among nodes in the network.

— **Compromised Master Clock.** A compromised master clock poses a severe threat to the PTP service, granting attackers control over the network's synchronization source. Attackers can take control of the master clock by exploiting supply-chain vulnerabilities or gaining unauthorized access, enabling manipulation of timing messages and disruption of network operations.
— **Replay Attack.** A replay attack in PTP involves capturing and retransmitting timing messages, causing devices to synchronize with outdated or modified timestamps. This scenario can disrupt the accuracy of time synchronization and compromise the integrity of time-sensitive operations.

Within the TIMESAFE framework, we focus on two specific attacks targeting PTP synchronization: *Spoofing Attack* and *Replay Attack*. These attacks are chosen for their direct impact on PTP synchronization mechanisms. The spoofing attack manipulates the BMCA to influence master clock election, affecting network-wide synchronization. The replay attack disrupts time synchronization by altering drift, offset, and delay calculations on slave clocks. Details of these attacks are illustrated in Figure 3 and discussed in the following sections. Other attack types, such as DoS and compromised master clock attacks, involve vulnerabilities beyond PTP itself.

4.1.1 Spoofing Attack. The TIMESAFE framework spoofing attack targets PTP nodes by sending counterfeit `Announce` messages to manipulate the BMCA leader election and assume the role of the master clock. Initially, the attacker monitors benign traffic to gather critical information about the protocol configuration and node attributes. With this data, the attacker crafts `Announce` packets designed to have superior attributes crucial for master clock selection. The BMCA relies on up to nine features to determine the master clock, starting with *Priority1* (Section 2.2). The attacker manipulates this value by setting it to 1. Similarly, the attacker sets the values for *ClockClass* to 1. The *ClockIdentity* field is created by inserting priority values between the manufacturer ID and device ID portions of the MAC address. The attacker inserts `ffff` ensuring it has higher priority values than the legitimate Master. Other values are copied from the legitimate master.

These crafted messages are sent periodically to maintain the attacker's role as master. Apart from `Announce` messages, no synchronization packets are sent, and `DelayReq` messages from slaves remain unanswered. By disrupting the synchronization process, all clocks operate independently, leading to a gradual drift. Consequently, the attacker introduces synchronization errors among all nodes in the network, causing a complete outage for the gNB, as we will show in Section 6.1 and illustrated in Figure 1.

4.1.2 Replay Attack. The Replay Attack involves sniffing time synchronization messages (`Sync` and `FollowUp`) from the Master clock, storing, and retransmitting them after a delay. When the Slave clock receives the retransmitted messages, it uses the outdated timestamp T_1 and a new timestamp T_x to compute drift, delay, and offset (see Figure 3). The formulas in Section 2.2 become: $Drift = \frac{(T_2'-T_x)-(T_1'-T_1)}{T_1'-T_1}$, $Delay = \frac{(T_4-T_1)-(T_3-T_x)}{2}$, and $Offset = (T_x-T_1)-Delay$. The actual difference between when T_1 is generated and received in the malicious PTP message significantly exceeds the calculated difference, causing offset to increase by orders of magnitude and severely disrupting network synchronization, as we will detail in Section 6.2.

5 Experimental Test Beds

For TIMESAFE, we use two experimental test beds to assess network timing interruption attacks, analyze PTP security measures in Open RAN networks, and evaluate malicious PTP detection systems on the DU. The first test bed is a production-ready 5G network, while the second is a dedicated Open RAN digital twin environment for rapid testing, prototyping, and developing attacks and solutions.

5.1 Production-ready Network

Our first experimental analysis is performed on a private 5G network, deployed at Northeastern University [52] comprising 8 NVIDIA **Aerial RAN CoLab (ARC)** nodes, with dedicated **Core Network (CN)** and FH infrastructure. We utilize this production-ready platform to capture traffic and evaluate attacks on an O-RAN and 5G-compliant system based on a 3GPP 7.2 split, with a combination of open-source components for the higher layers of the stack, as well as commercial devices for the fronthaul infrastructure and the radios [41].

NVIDIA ARC combines the open-source project **OpenAirInterface (OAI)** [43] for the higher layers of the protocol stack with NVIDIA Aerial, a physical layer implementation accelerated on **Graphics Processing Unit (GPU)** (NVIDIA A100) and with a 7.2 implementation that combines NVIDIA Mellanox smart **Network Interface Cards (NICs)**. In an ARC server, the GPU is coupled with the programmable NIC (a Mellanox ConnectX-6 Dx) via **Remote Direct Memory Access (RDMA)**, bypassing the CPU for direct packet transfer from the NIC to the GPU. This architecture enables high-speed DU-side termination of the Open FH interface.

The FH infrastructure uses a Dell S5248F-ON switch and a Qulsar QG-2 as the grandmaster clock, distributing PTP and SyncE synchronization to the DU and RU with an O-RAN LLS-C3 configuration. For PTP synchronization, we use *ptp4l*, an IEEE 1,588 compliant implementation for Linux that offers extensive configuration options and enables monitoring and logging of synchronization. The RU is a Foxconn 4T4R unit operating in the $3.7 - 3.8$ GHz band, with **Commercial Off-the-Shelf (COTS)** 5G smartphones from OnePlus (AC Nord 2003) as **User Equipments (UEs)** [7].

While the DU codebase is open and potentially extensible to secure the S-Plane, the RU incorporates a closed-source FPGA-based termination for the FH interface, precluding additional security mechanisms in the FH environment from being directly enabled. Thus, we also adopt a trace-based approach similar to [21], configuring a port of the FH switch to mirror the FH traffic to a server running packet capture. Specifically, we capture all traffic traversing the FH for over 20 minutes under several different traffic loads.

5.2 Digital Twin Environment

To leverage these traces we built a digital twin environment using two desktop computers with Intel i9-13900K CPUs and NVIDIA Mellanox ConnectX-4 Lx NICs to represent the DU and RU, while a third desktop computer with an Intel Core i7-10700 CPU serves as the attacker. All machines are connected over a private, 10 Gbps capable switched network. Digital twins provide a highly accurate representation of the network and its devices, making them invaluable for designing and testing ML-based algorithms to detect malicious traffic and ongoing attacks [31]. Once trained, ML-based control solutions need to be validated and tested in controlled environments to avoid disruptions in production networks [44]. Our setup allows for rapid testing, prototyping, and the development of both attacks and solutions, ensuring robust security measures before deployment in real-world scenarios, meeting the need to test and evaluate any proposed S-plane security measures as discussed in [30]. We use `.pcap` files captured on the production-ready Open FH to accurately emulate the original FH and observe the impact the C and U plane traffic has on the PTP protocol operating in the S-Plane [21].

On top of the background traffic, we start PTP traffic between the DU and RU with *ptp4l*. This mirrors the setup in our production-ready network, which also uses *ptp4l* for PTP synchronization. We use the LLS-C2 profile in our digital twin environment, where the DU is the Master Clock and the RU is our Slave Clock. The attacker does not actively participate in the PTP protocol. Instead, it is capable of observing, replaying, or inserting PTP traffic. We discussed the details of our attacks in Section 4.1.

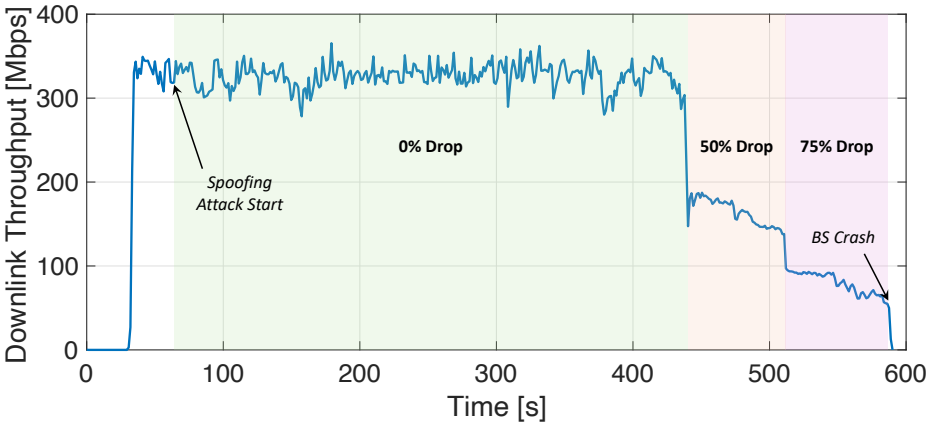


Fig. 4. Downlink throughput to a UE during a spoofing attack with the switch port set to PTP master. Approximately 60 seconds into the experiment, the spoofing attack is initiated. Around second 440 there is a 50% drop in throughput, progressing to a 75% drop by second 510, and ultimately causing the base station to crash at around second 580.

6 Impact of the Attacks

Implementing security measures incurs performance costs [21, 22], so balancing attack risks with the cost of securing the system is crucial. We use the TIMESAFE framework to evaluate the impact of attacks on the S-plane within our test environments, following the O-RAN interfaces S-plane specification [39]. Our assessment focuses on the gNB's availability. According to NIST's **National Vulnerability Database (NVD)** [6], impact levels are categorized as none, low, or high. Low impact attacks cause performance degradation or intermittent availability without fully denying service, affecting functionalities like Carrier Aggregation and handovers[35]. High impact attacks, however, result in complete outages, denying service to all User Equipments (UEs).

6.1 Production-ready Network Impact

We test our attacks in the production-ready private 5G network to assess each attack's impact following the recently published guidelines in the O-RAN ALLIANCE testing specifications [39]. Based on our threat model in Section 4, we assume that one of the eight DUs is the compromised machine.

In our initial tests, we found that switch configurations can provide protection against attacks. Malicious packets are not always received by other nodes, as verified through Wireshark captures on both the malicious machine and other nodes. This protection is tied to the switch port's PTP role settings. On the Dell S5248F-ON switch, ports set to master only send PTP packets, slave mode only receives, and dynamic mode allows both. Additionally, switch configurations can block duplicate MAC addresses. While a properly configured switch prevented our replay attack, assuming correct configurations is risky, especially when DU and RU vendors lack control over network settings, and the complexity and diversity of equipment and configurations in the FH path exacerbate this risk [24, 25, 46].

In our first successful attack, we configure the port of our malicious machine with a PTP master role. We start our private 5G network, connect the UE, and begin sending around 300 Mbps downlink iPerf traffic. After approximately 60 seconds, we launch the spoofing attack, as shown in Figure 4. Monitoring the PTP service logs at the DU reveals that the attack causes the DU to lose synchronization with the grandmaster clock, leading to a drift in network synchronization as the

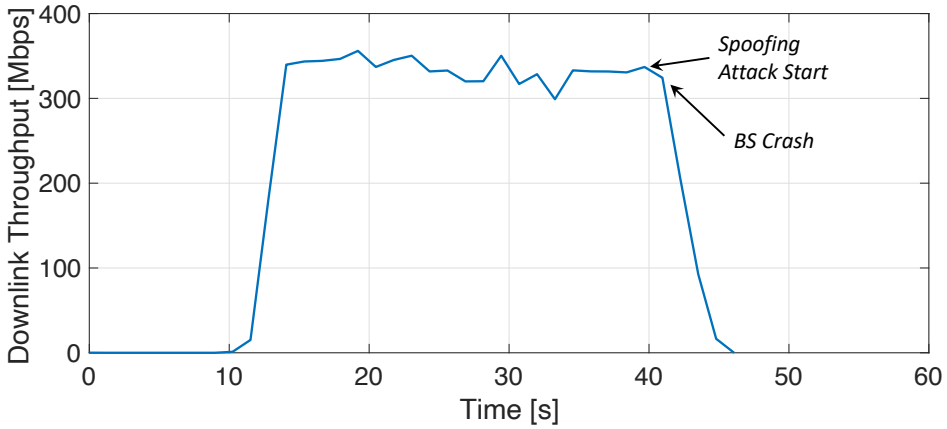


Fig. 5. Downlink throughput to a UE during a spoofing attack with switch ports set to PTP dynamic role. Throughput remains stable at around 350 Mbps initially. About 40 seconds in, the spoofing attack begins, causing the 5G cell to drop and the UE to lose connectivity.

DU uses its internal clock. This drift degrades UE performance, eventually causing disconnection from the network. About 380 seconds after the beginning of the attack, there is a sudden 50% drop in throughput. From this point, the throughput continues to drop roughly linearly as the clock drift increases. Finally, the DU loses synchronization with the RU, the throughput drops to 0, and the UE disconnects. When this attack stops, the DU resynchronizes with the grandmaster clock, and the network returns to a healthy state.

Next, we configure all ports as PTP dynamic so that malicious packets can be received by the other nodes and repeat the experiment. This time, the malicious PTP packets reach all the machines in the network. Approximately 2 seconds after the attack begins, the RU crashes, stopping its operations, causing the 5G cell to drop and the UE to lose connection, as shown in Figure 5. While the DU was able to recover on its own after the first attack, the RU required a manual reboot to regain functionality after the second attack. This behavior might be device specific, in our case the Foxconn RU. Regardless of any potential vendor-specific variations, these attacks proved to be very powerful, causing significant disruption and necessitating manual intervention to restore full network operations. These tests demonstrate the impact to the Open RAN gNB is *high*. Failing to secure the S-plane against spoofing attacks results in a complete outage of the gNB.

6.2 Digital Twin Network Impact

Next we tested the impact of our attacks on PTP functionalities in the Digital Twin environment, focusing on disruptions to synchronization and leader election rather than effects on the UE. We employed cyclic attack/recovery patterns with durations of 30/30, 40/20, 50/10 seconds, and continuous attacks for both of the attacks discussed in Section 4.1. The attack duration is the period (in seconds) the attack is carried out, while during the recovery time no attack occurs allowing PTP synchronization to recover. Each experiment involved four background traffic traces, resulting in 32 tests with durations ranging from 120 to 450 seconds, depending on the traffic. Results from the RU confirm significant performance degradation in PTP, with varying severity of impact.

To generate labeled training data, we used logs to identify the timing of attacks and the specific machines involved (based on MAC addresses). A script was written to label each packet in the dataset as either benign (0) or malicious (1) based on whether it came from an attacking machine during the attack window.

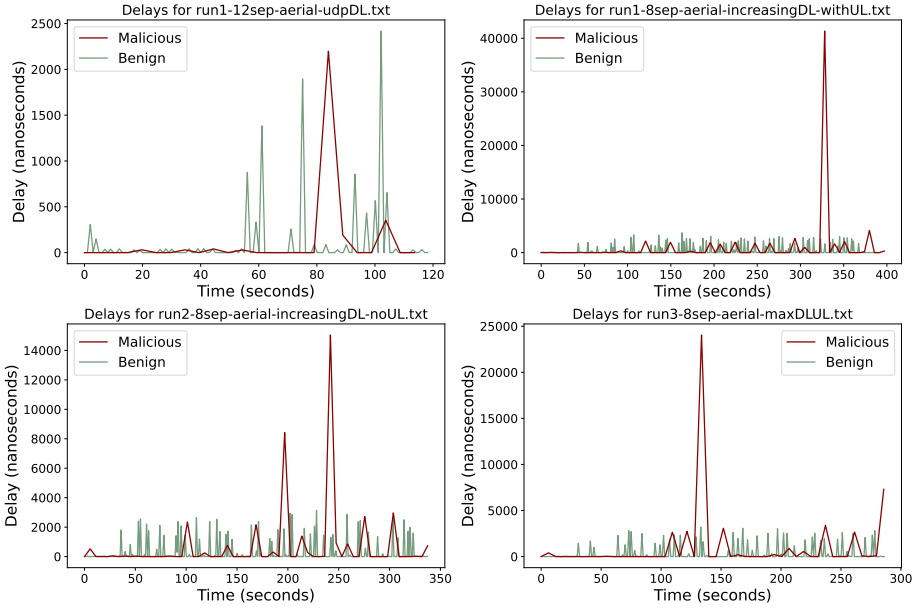


Fig. 6. The drift caused by the spoofing attack (dark red) can exceed 40,000 ns, exceeding normal background traffic (light green). During the 50-second spoofing attack period, the RU relies on its internal clocks. When the 10-second recovery period begins, a large spike in calculated delay represents the synchronization drift caused by the attack.

Comprehensive results are displayed in the Appendix (Section A.1), but here we highlight the effectiveness of both attacks.

Spoofing Attack: Figure 6 shows the impact of the 50/10 (50 s attack, 10 s recovery) spoofing attack. Synchronization is disrupted whenever the attacker becomes the master clock, causing the RU to rely on its internal clocks, leading to synchronization drift. The impact of this attack depends on the accuracy of each devices internal clock as well as the length of time of the attack. The observable spike in delay is actually reported shortly after the attack ends, when the legitimate master and slave begin synchronizing again. This spike shows the amount of synchronization drift that occurred during the attack. While there is some natural variation in delay due to the background traffic, shown in light green in Figure 6, the announce attack (dark red) shows synchronization drift can exceed 40,000 ns.

Replay Attack: This attack has a more severe impact on node synchronization, as shown in Figure 7, where we display the results of the 30/30 s cycle. Normally, delay values range between 25 and 45 nanoseconds, but during the Replay attack, these values spike by several orders of magnitude, peaking at over 20 million nanoseconds. Comparing Figures 6 and 7 it is evident that the replay attack is more disruptive than the spoofing attack, causing a high synchronization drift more quickly and requiring more time for PTP to fully recover.

7 Need for ML Based Detection

As shown in Section 6, synchronization attacks can have catastrophic consequences, including complete outages of the gNB. However, the virtualized, multi-vendor nature of Open RAN and the high precision required for synchronization over the S-plane create challenges for implementing traditional security measures. While hardware-based solutions like MACsec can provide security, they introduce additional costs and complexity to the resource-constrained RU [17, 18].

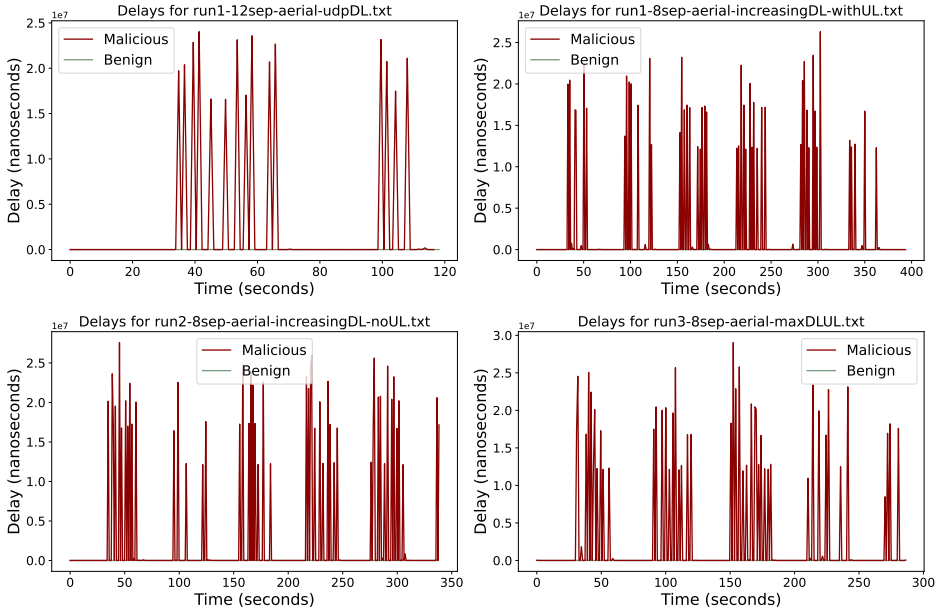


Fig. 7. The delay values during the 30-second replay attack (dark red) spike to over 20 million nanoseconds, compared to normal background traffic (light green). The replay attack causes a much higher synchronization drift more quickly than the spoofing attack, and it takes longer for PTP to recover.

To address these limitations, we propose a monitoring system (Prong D) within the DU, which has more computational resources and flexibility. Given the complexity and variety of potential attacks on synchronization protocols, simple rule-based or threshold mechanisms are insufficient. Instead, ML offers a more adaptive approach for several reasons.

- **Rule-Based Solutions:** Developing rule-based solutions requires extensive expert knowledge to create specific rules for every potential attack. In Section 7.1, we outline the complexity involved in designing expert rules to detect even just two types of attacks.
- **Fixed Thresholds:** Rule-based solutions often rely on fixed thresholds, which are challenging to determine optimally. Even with well-designed thresholds, these methods are limited in their ability to adapt to new or evolving threats. In contrast, ML models do not require setting fixed thresholds. As we will show in Section 9.2, well designed models exhibit resilience againste reshaped attacks.
- **Flexibility:** In contrast to rule-based approaches, a well-trained ML model can generalize across various attack types. For instance, we created a new DoS attack by sending a high rate of random or invalid PTP messages. Although this attack did not disrupt PTP timing, it still indicated malicious activity. Our ML model, despite not being specifically trained on this attack, correctly identified over 66% of the malicious traffic, whereas the heuristic model failed completely. This shows that TIMESAFE's ML-based solution is highly flexible and capable of detecting sophisticated, unpredictable attacks.
- **Accuracy.** An effective monitoring solution must deliver accurate results. As we will demonstrate in Section 9.1, our ML model significantly outperforms the rule-based approach, achieving 99% accuracy in a production environment.

Rule-based systems such as those described in [2] struggle to keep pace with the wide range of attacks, while other heuristic methods require additional clocks or changes to the physical

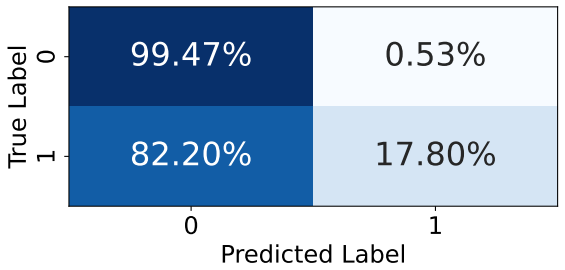


Fig. 8. Confusion matrix of the heuristic solution for detection of PTP attacks showing an extremely high FN rate.

infrastructure [20, 32, 49]. Additionally, legitimate changes in master clocks, changes in network topology, and sequence number resets can often resemble malicious activity, complicating detection efforts. ML models excel at identifying patterns and anomalies within large datasets, enabling them to distinguish between benign fluctuations and actual threats. By leveraging ML, we can develop robust monitoring systems capable of adapting to new and evolving attack vectors, ensuring reliable network performance and security.

7.1 Heuristic Detection Results

To highlight the challenges of rule-based systems, We developed a heuristic model to detect attacks using a set of predefined rules.

Spoofing Attack: This attack sends numerous `Announce` packets to become the master and disrupt synchronization. Our heuristic detects this by monitoring the number of adjacent `Announce` packets. Typically, no more than two adjacent `Announce` messages are seen in normal traffic. If the heuristic detects four or more consecutive `Announce` packets, it flags an attack.

Replay Attack: This sophisticated attack replays benign `Sync` and `FollowUp` packets with delays, mimicking legitimate traffic. Detection relies on the Sequence ID and Message Type, looking for anomalies like `Sync` and `FollowUp` messages separated by other packets, Sequence ID values lower than previous packets, and duplicates. The heuristic uses a queue of the last S packets to identify these issues, but this method may strain memory resources.

Testing under the same conditions we use in Section 9.2 showed that while the heuristic effectively detects `Announce` attacks, it struggles with overall accuracy. The primary issue is its sensitivity to individual packets: legitimate packets observed during an attack can reset the heuristic, causing it to miss subsequent malicious packets. Results, as shown in Figure 8, indicate poor overall performance due to these limitations.

8 TIMESAFE ML-Based Detection

Traditional heuristic methods are inadequate for detecting all types of malicious packets, given the dynamic and adaptive nature of cyberattacks. Instead, a ML approach is better suited to capture the temporal dependencies and patterns in PTP data. PTP packets, characterized by features such as Ethernet addresses, sequence IDs, message types, and inter-arrival times, form a high-dimensional feature space where subtle anomalies may occur. Unlike rule-based systems, ML models can learn these complex patterns and correlations to identify potential malicious activity.

8.1 ML Model Selection

We used our digital twin environment to generate training data, as described in Section 6.2. The data was split into equal-sized chunks (1,000 messages each) without overlaps. We randomly selected

Table 3. All Machine Learning Models Outperform the Heuristic (Rule-based) Approach in Both Accuracy and Recall on the Offline Dataset

Model	Accuracy (%)	Recall (%)
Heuristic	50.95	17.80
Linear Regression	75.70	17.90
KNN	79.91	51.49
Gradient Boosting Classifier	86.59	58.75
Random Forest	84.99	72.00
Decision Tree	84.26	72.45
LSTM	72.97	75.00
Naïve Bayes	65.72	83.26
CNN	**97.90**	**97.05**
Transformer	**97.99**	**98.33**

The CNN and Transformer models achieve significantly higher performance than other methods.

80% of the chunks for training, 10% for validation, and 10% for testing. Upon acceptance, we will open-source our complete data set and training pipeline.

We initially evaluated a wide range of ML models, as shown in Table 3, using the same test traces for all offline accuracy tests. All ML models tested had higher accuracy than the rule-based approach in Section 7.1. However, accuracy alone is not sufficient; the impact of **False Negatives (FNs)** (or missing an attack) is catastrophic, as shown in Section 6.1, while the impact of **False Positives (FPs)** is much lower. Therefore, we also examine each model's recall.

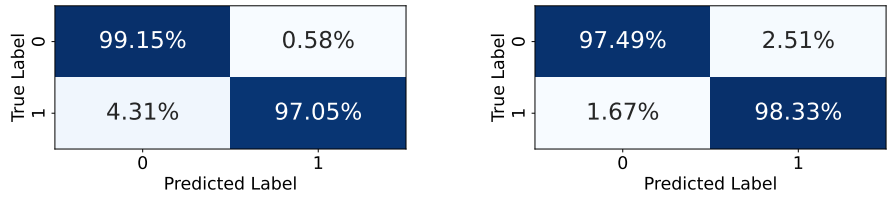
We do recognize that in operational environments, even a modest false positive rate can generate frequent alerts and contribute to operator fatigue, especially when attacks are rare. Heuristic methods may have an advantage of allowing direct adjustment of detection thresholds to accommodate different tolerance for false positives. Although our ML-based approach does not expose simple rule thresholds, the effective sensitivity of the model can be tuned through several mechanisms, such as adjusting the decision threshold on the output probabilities, retraining with modified class weights, or employing cost-sensitive loss functions. These mechanisms allow operators to adapt the false positive/false negative tradeoff to specific deployment needs while maintaining the model's ability to learn complex, non-linear behaviors that heuristic rules cannot capture.

Given the clear performance advantage of the CNN and Transformer models, we selected both for further evaluation. The complete confusion matrices for the initial offline results for these models are shown in Figure 9. Additional details about all models are provided in Appendix A.3.

Our task of classifying packet sequences as malicious or benign parallels sentiment analysis and image classification, where context or spatial features are key. Just as sentiment analysis depends on word context and image classification on visual patterns, our models must capture subtle packet sequence patterns. CNNs excel at extracting spatial features, making them effective at identifying key packet patterns, while transformers are superior in handling sequential data and capturing contextual dependencies. This capability to understand complex patterns intuitively explains why CNN and transformer models outperform others in detecting malicious PTP activities.

8.2 ML Model Description

Here we briefly describe the design of the two best performing models, the CNN and transformer. A complete description of each model (see Section A.3) along with code used will be made open source upon acceptance of this article. All the models use the same six input features, F: Ethernet source



(a) CNN (b) Transformer

Fig. 9. Confusion matrices from initial offline testing for the CNN and transformer.

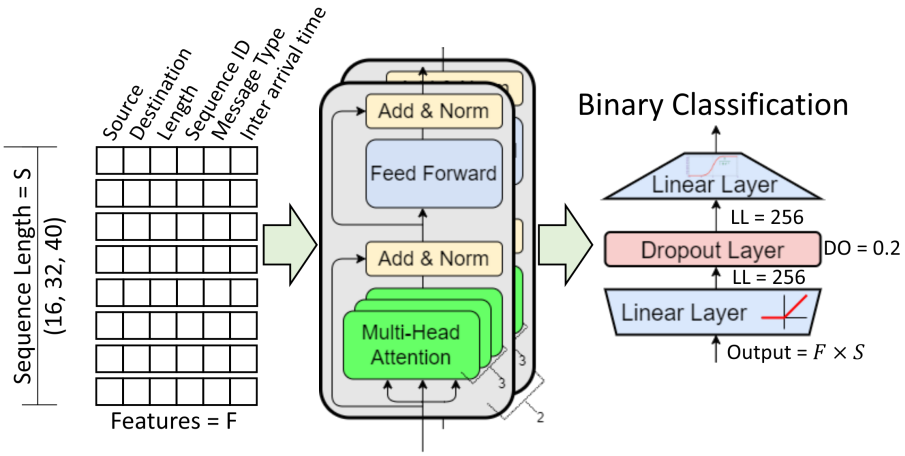


Fig. 10. The final TIMESAFE transformer model uses two transformer layers, a **fully connected (FC)** linear layer with ReLU activation function, a dropout layer, and a second linear layer with a sigmoid activation function. The output is a single binary classification for the entire sequence.

and destination addresses; packet size; PTP sequence ID; PTP message type; and inter-arrival time. The MAC address helps identify flows and packet direction. The PTP message type and sequence ID are crucial for understanding the nature and order of PTP messages, aiding in the detection of deviations from expected behavior. Packet length can indicate malformed or malicious packets with extra payload. Inter-arrival time, specified by the O-RAN ALLIANCE for both announce and sync messages, is another key indicator of potential attacks, though minor variations between packets are expected.

Our CNN model uses a 2D convolutional architecture to process input sequences of fixed length S ($S = 32$ for all results presented) and six features ($F = 6$), represented as a one-channel image with dimensions corresponding to the sequence length and feature count. The model has two convolutional layers: the first uses 16 filters with a 3×3 kernel, followed by ReLU activation and 2×2 max pooling, reducing spatial dimensions. The second layer applies 32 filters with a 3×3 kernel, followed by ReLU and max pooling. The output is flattened and passed through two FC layers with ReLU and sigmoid activations to produce a binary classification.

Our transformer-based model (see Figure 10) uses only the encoder portion of a full transformer architecture. It processes input sequences with fixed lengths, varied during training and evaluation to include lengths of $S = \{16, 32, 40\}$ packets. These lengths balance the trade-off between incorporating more historical data (and requiring more memory) and improving accuracy and recall. We also treated the number of attention heads $\{2, 3\}$ as a hyper-parameter with two configurations. The transformer encoder outputs a feature matrix of dimension $F \times S$, which is then fed into a FC

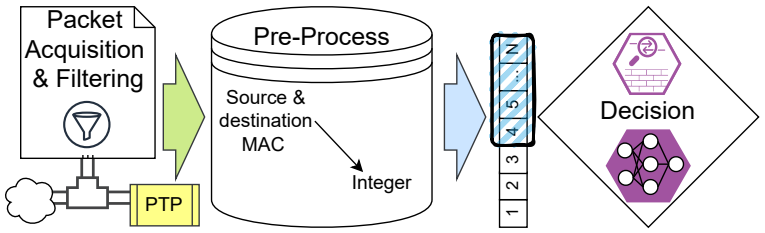


Fig. 11. Overview of the TIMESAFE pipeline deployed for detection in both a production-ready private 5G network and its digital twin environment. The pipeline consists of three modular stages: Packet acquisition and filtering, pre-processing, and decision-making.

linear layer with a ReLU activation function, followed by a dropout layer with a probability of 0.2 and a final linear layer with a sigmoid activation function. The model produces a single binary classification for the sequence.

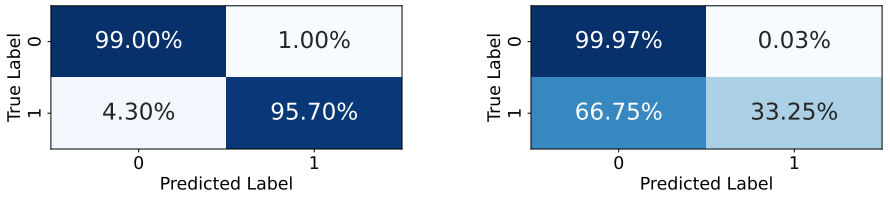
Each sequence of length S (used for both CNN and Transformer models) was classified as malicious if any packet in the sequence was labeled as malicious, allowing the models to learn patterns of attack behavior over time. For both the CNN and transformer models, we employed a custom Weighted Binary Cross-Entropy Loss function to address class imbalance and unequal error costs. However, since the initial setting of one for both weights yielded satisfactory results, we did not adjust these weights further. In practice, these weights, which correspond to the decision threshold, could be modified to adjust the sensitivity of detection. For example, deployments with more stable network baselines may prefer higher precision to reduce false alarms, whereas high-security environments might prioritize recall to minimize missed attacks. Exploring adaptive threshold tuning or online re-weighting in future work could further improve deployability across diverse scenarios. We used the Adam optimizer with an initial learning rate of 0.001 and configured a StepLR scheduler to reduce the learning rate by a factor of 0.1 every 10 epochs. Early stopping was enabled, with monitoring for validation loss improvement and a patience parameter set to 20 epochs.

Training time depends on both the size of the dataset and the hardware used. However, training is performed offline, making it a one-time cost that does not impact real-time performance. On modern hardware—such as a single NVIDIA A100 GPU on a DGX machine—training never exceeded 30 minutes. This cost is negligible in practice, especially when amortized across many long-term deployments. In the broader context, it is a negligible cost to pay for adding robust security to production environments.

8.3 **Deploying** TIMESAFE

We deploy TIMESAFE as a passive detection pipeline to monitor PTP synchronization traffic in the fronthaul segment of the O-RAN architecture. It can run directly on the DU, observing the Ethernet interface connected to the fronthaul switch, or on a separate device connected to a mirror port in the switched network—particularly valuable in LLS-C2 and LLS-C3 profiles where synchronization messages traverse multiple hops and are most vulnerable to attack.

The pipeline consists of three modular stages: packet acquisition and filtering, pre-processing, and decision-making, as shown in Figure 11. The first stage captures Ethernet frames and filters for PTP packets, extracting six features: source and destination MAC addresses, frame length, PTP sequence ID, message type, and inter-arrival time. These are passed via thread-safe queues to the pre-processing stage, which maps MAC addresses to integers to abstract away device-specific identifiers and enable generalizable traffic pattern learning.



(a) Transformer (b) CNN

Fig. 12. Confusion matrices for production environment attack detection. The transformer greatly outperforms the CNN, showing it is able to adapt to new environments.

The decision-making stage maintains a sliding window of size S and evaluates each sequence using either a machine learning model or a rule-based algorithm. After every two new PTP packets, the window advances, and a new classification decision is made. The model's detection process is designed to operate in real time, with low-latency inference on both the CNN and Transformer models—even on moderate hardware. While the Transformer architecture is computationally heavier due to its attention mechanism, both models sustain real-time decision-making without introducing bottlenecks in both of our testbeds. This approach enables prompt detection of anomalies while keeping latency low. The modular design also allows each stage to be updated independently, supporting flexible deployment and future model improvements.

9 TIMESAFE Detection Results

This section demonstrates that TIMESAFE accurately detects attacks in a production-ready 5G network and shows its resilience to reshaping attacks in our Digital Twin environment.

9.1 Production-ready Detection Accuracy

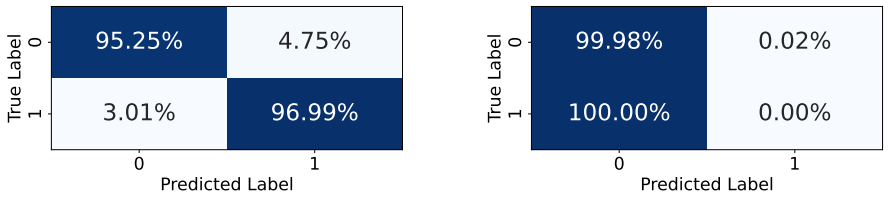
We evaluated the TIMESAFE detection pipeline using a trace (`.pcap`) captured during a successful attack in the production-ready environment, as illustrated in Figure 5. The results demonstrate that the transformer model with two attention heads and a slice length of 32 significantly outperformed other transformer configurations, achieving an accuracy exceeding 99%. The confusion matrix in Figure 12(a) illustrates the model's strong performance, highlighting its ability to accurately distinguish between malicious and benign traffic in a real-world setting.

In contrast, Figure 12(b) shows that the performance of the CNN model was substantially degraded when applied to the new environment. This suggests that the CNN struggled to generalize to unseen data and failed to adapt to the subtle contextual variations present in the production environment. The transformer model, however, maintained its robustness, effectively identifying the nuanced patterns and indicators of the attack, even in a more complex and dynamic setting. This reinforces the transformer's ability to generalize well to new conditions, capturing intricate relationships across sequences that may vary from one environment to another.

While the CNN exhibited slightly lower false positives in controlled digital twin experiments, its inability to generalize to the production environment led to a significant drop in overall detection reliability. In contrast, the transformer's small increase in false positives is outweighed by its robustness to environmental shifts and unseen attack patterns. In practical deployments where detection failures are more costly than occasional false alarms, this tradeoff favors the transformer as the more stable and effective choice.

9.2 Digital Twin Detection Accuracy

To further validate our ML model, we deployed TIMESAFE in a digital twin environment that simulates the production-ready network. This setup provides a realistic testbed for evaluating the



(a) Transformer (b) CNN

Fig. 13. Confusion matrices for evaluating resilience to reshaped attacks with randomized attack and recovery durations. The transformer shows resilience to this strategy change, while the CNN fails to generalize.

model's performance under a wide variety of conditions. While running full PTP with *ptp4l*, we keep system time updates on the DU and attacker disabled, allowing us to log prediction outcomes with accurate DU timestamps. The attacker logs the attack type and its timestamps, enabling us to measure decision accuracy. Analysis reveals that we can detect attacks with just one or two malicious PTP packets reaching the DU.

During the experiment, several tests were performed. At the start of each test, the DU, RU, and attacker exchanged initial messages to confirm readiness. Once prepared, they initiated a 5-minute test. Throughout the test, the DU continuously logged predictions with timestamps, while the attacker recorded the attack types, along with their start and end times, and the duration of both the attack and recovery phases. This cycle repeated for the entire test duration, alternating between attack and recovery phases. Recall that during training, we used cycles of 30/30, 40/20, 50/10 seconds, and continuous attack, as described in Section 6.2. To evaluate the models' ability to generalize to altered attack patterns, the attacker in this experiment randomly selected an attack duration from the continuous range $[10, 30]$ seconds and a recovery duration from the continuous range $[40, 60]$ seconds. The detection results from this experiment are shown in Figure 13. The transformer demonstrated excellent resilience against the reshaped attack, while the CNN completely failed to detect it. This outcome highlights the transformer's robust ability to adapt to previously unseen attack strategies, confirming greater suitability for real-world deployment where attack patterns can be unpredictable.

10 Discussion

The Open RAN framework's push towards increased virtualization and softwarization [9, 28] necessitates security solutions that align with this trend. Prong D (Section 2.3), which involves monitoring, is well-suited to the softwarized DU environment. However, future work should prioritize several additional key areas.

Attacks in Other Synchronization Topologies. Our focus on LLS-C2 and LLS-C3, due to their high vulnerability, does not encompass all risks. For instance, jamming GNSS signals could affect LLS-C4, and supply chain attacks on primary reference clocks pose additional threats. Network redundancy, as suggested in Prong C, enhances protection and resilience, potentially mitigating such threats through redundant paths and clocks [20, 29, 49, 54].

Additional Attacks Against PTP. While TIMESAFE currently addresses two types of attacks, other threats like packet removal and selective delay manipulation also merit investigation [30]. Expanding TIMESAFE to include multi-class classification for various attack types could refine threat detection and response strategies.

Authentication. Enhancing PTP security through authentication, as outlined in Prong A, is crucial. Implementing digital signatures can verify message integrity and authenticity, preventing replay and spoofing attacks. However, integrating these mechanisms must balance security with the

real-time performance needs of PTP in Open RAN. Efficient, lightweight cryptographic algorithms are essential to avoid introducing latency that disrupts timing synchronization. Additionally, the cost of upgrading RU equipment, particularly FPGA-based systems, to support these algorithms must be weighed against the possible risks to enable an informed, risk-based decision on where and how to best implement this type of solution.

Our detection-based approach is designed to complement these preventive measures. Unlike cryptographic or hardware-based defenses, which may require system upgrades or protocol changes, TIMESAFE operates on traffic patterns alone and can be deployed as a lightweight, software-only monitor in existing networks. While authentication mechanisms aim to prevent certain attacks, TIMESAFE helps identify when those defenses may be necessary, have failed or when other forms of manipulation (e.g., asymmetric delay) occur. Thus, detection and defense should be viewed as complementary components of a comprehensive security posture for PTP synchronization.

11 Conclusion

Securing PTP in the S-plane within the Open RAN framework is essential, as demonstrated by our findings. Successful attacks can lead to severe disruptions, such as complete gNB outages requiring manual restoration, underscoring the need for robust security in multi-vendor environments that expand attack surfaces.

Addressing these threats involves balancing security costs against performance and financial implications. While robust measures like encryption introduce latency and computational overhead, TIMESAFE offers an effective solution. Our machine learning-based monitoring system achieves over 97.5% accuracy in detecting malicious attacks, providing a cost-efficient approach with minimal additional cost.

By using advanced machine learning techniques, TIMESAFE enables real-time detection of threats, allowing for targeted application of higher-cost security measures only when necessary. This adaptive strategy optimizes security and performance, highlighting the importance of tailored solutions to safeguard PTP synchronization in Open RAN networks. The demonstrated vulnerabilities in the FH call for proactive, cost-effective security measures to prevent significant network disruptions.

References

[1] 2018. Timing characteristics of synchronous equipment slave clock. *International Telecommuniation Union Recommendation ITU-T G.8262/Y.1362* (2018). Retrieved from https://www.itu.int/rec/T-REC-G.8262-201811-I/en

[2] 2020. IEEE standard for a precision clock synchronization protocol for networked measurement and control systems. *IEEE Std 1588-2019 (Revision ofIEEE Std 1588-2008)* (2020), 1–499. DOI: https://doi.org/10.1109/IEEESTD.2020.9120376

[3] 2020. NIST Special Publication 800-207: Zero Trust Architecture. DOI: https://doi.org/10.6028/NIST.SP.800-207

[4] 2022. Precision time protocol telecom profile for phase/time synchronization with full timing support from the network. *International Telecommuniation Union Recommendation ITU-T G.8275.1/Y.1369.1* (2022). Retrieved from https://www.itu.int/rec/T-REC-G.8275.1-202211-I/en

[5] 2025. Retrieved from https://github.com/genesys-neu/s-plane_security

[6] 2025. Retrieved from https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator

[7] 2025. *OnePlus*. Retrieved from https://www.oneplus.com/us/nord-specs

[8] A. S. Abdalla, P. S. Upadhyaya, V. K. Shah, and V. Marojevic. 2022. Toward next generation open radio access networks: What O-RAN can and cannot do!. In *IEEE Network* 36, 6 (2022), 206–213. DOI: 10.1109/MNET.108.2100659

[9] Ibrahim Afolabi, Tarik Taleb, Konstantinos Samdanis, Adlen Ksentini, and Hannu Flinck. 2018. Network slicing and softwarization: A survey on principles, enabling technologies, and solutions. *IEEE Communications Surveys & Tutorials* 20, 3 (2018), 2429–2453.

[10] Tolga O. Atalay, Sudip Maitra, Dragoslav Stojadinovic, Angelos Stavrou, and Haining Wang. 2023. Securing 5G OpenRAN with a scalable authorization framework for xApps. In *Proceedings of the IEEE INFOCOM 2023 - IEEE Conference on Computer Communications*. 1–10. DOI: https://doi.org/10.1109/INFOCOM53939.2023.10228961

[11] Jason Boswell and Scott Poretsky. 2020. Security considerations of open RAN. *Stockholm: Ericsson* (2020).

[12] Joo Yeon Cho and Andrew Sergeev. 2021. Secure open fronthaul interface for 5G networks. In *Proceedings of the 16th International Conference on Availability, Reliability and Security* (Vienna, Austria) *(ARES 21)*. Association for Computing Machinery, New York, NY, USA, Article 107, 6 pages. DOI: https://doi.org/10.1145/3465481.3470080

[13] Common Public Radio Interface. 2019. *Common Public Radio Interface: eCPRI Interface Specification V2.0*. Technical Report. Common Public Radio Interface. Retrieved from http://www.cpri.info/downloads/eCPRI_v_2.0_2019_05_10c.pdf

[14] Casimer DeCusatis, Robert M. Lynch, William Kluge, John Houston, Paul A. Wojciak, and Steve Guendert. 2019. Impact of cyberattacks on precision time protocol. *IEEE Transactions on Instrumentation and Measurement* 69, 5 (2019), 2172–2181.

[15] Daniel Dik and Michael Stübert Berger. 2021. Transport security considerations for the open-ran fronthaul. In *Proceedings of the 2021 IEEE 4th 5G World Forum (5GWF)*. IEEE, 253–258.

[16] D. Dik and M. S. Berger. 2023. Open-RAN fronthaul transport security architecture and implementation. In *IEEE Access* 11 (2023), 46185–46203. DOI: 10.1109/ACCESS.2023.3274487

[17] Daniel Dik, Iacob Larsen, and Michael Stübert Berger. 2023. MACsec and AES-GCM hardware architecture with frame preemption support for transport security in time sensitive networking. In *Proceedings of the 2023 International Conference on Computer, Information and Telecommunication Systems (CITS)*. IEEE, 01–07.

[18] D. Dik, R. D. Oliveira, E. Arabul, C. Vrontos, M. S. Berger, R. Nejabati, and D. Simeonidou. 2023. 100 Gbps multi-tenant FPGA-based MACsec aggregation to secure the open-RAN fronthaul. In *Proceedings of the 49th European Conference on Optical Communications (ECOC 2023)*, Vol. 2023. IET, 870–873.

[19] ETSI. 2022. *Publicly Available Specification (PAS); O-RAN Fronthaul Control, User and Synchronization Plane Specification v07.02; (O-RAN-WG4.CUS.0-v07.02)*. Technical Report. ETSI TS 103 859 V7.0.2 (2022-09).

[20] A. Finkenzeller, O. Butowski, E. Regnath, M. Hamad, and S. Steinhorst. 2024. PTPsec: Securing the precision time protocol against time delay attacks using cyclic path asymmetry analysis. *IEEE INFOCOM 2024 - IEEE Conference on Computer Communications*, Vancouver, BC, Canada, 461–470. DOI: 10.1109/INFOCOM52122.2024.10621345

[21] J. Groen et al. 2024. Securing O-RAN open interfaces. In *IEEE Transactions on Mobile Computing* 23, 12 (2024), 11265–11277. DOI: 10.1109/TMC.2024.3393430

[22] J. Groen et al. 2025. Implementing and evaluating security in O-RAN: Interfaces, intelligence, and platforms. In *IEEE Network* 39, 1 (2025), 227–234. DOI: 10.1109/MNET.2024.3434419

[23] C. -F. Hung, Y. -R. Chen, C. -H. Tseng, and S. -M. Cheng. 2024. Security threats to xApps access control and E2 interface in O-RAN. In *IEEE Open Journal of the Communications Society* 5 (2024), 1197–1203. DOI: 10.1109/OJCOMS.2024.3364840

[24] C. David Hylender, Philippe Langlois, Alex Pinto, and Suzanne Widup. 2024. *2024 Data Breach Investigations Report*. Technical Report. Verizon Business. Retrieved from https://www.verizon.com/business/resources/Tce6/reports/2024-dbir-data-breach-investigations-report.pdf

[25] IBM. 2023. *IBM X-Force Cloud Threat Landscape Report 2023*. Technical Report. IBM Corporation. Retrieved from https://www.ibm.com/downloads/cas/QWBXVAPL

[26] Eyal Itkin and Avishai Wool. 2017. A security analysis and revised security extension for the precision time protocol. *IEEE Transactions on Dependable and Secure Computing* 17, 1 (2017), 22–34.

[27] Felix Klement, Alessandro Brighente, Michele Polese, Mauro Conti, and Stefan Katzenbeisser. 2024. Securing the open RAN infrastructure: Exploring vulnerabilities in kubernetes deployments. In *Proceedings of the 2024 IEEE 10th International Conference on Network Softwarization (NetSoft)*. 185–189. DOI: https://doi.org/10.1109/NetSoft60951.2024.10588929

[28] Woo-Hyun Ko, Ushasi Ghosh, Ujwal Dinesha, Raini Wu, Srinivas Shakkottai, and Dinesh Bharadia. 2024. EdgeRIC: Empowering real-time intelligent optimization and control in NextG cellular networks. In *Proceedings of the 21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24)*. 1315–1330.

[29] Nikita Lazarev, Tao Ji, Anuj Kalia, Daehyeok Kim, Ilias Marinos, Francis Y. Yan, Christina Delimitrou, Zhiru Zhang, and Aditya Akella. 2023. Resilient baseband processing in virtualized RANs with slingshot. In *Proceedings of the ACM SIGCOMM 2023 Conference*. 654–667.

[30] Assrar Maamary, Hyame Assem Alameddine, Mourad Debbabi, and Chadi Assi. 2025. Synchronization plane in O-RAN: Overview, security and research directions. *IEEE Communications Magazine* 63, 2 (2025), 88–94. DOI: https://doi.org/10.1109/MCOM.001.2300563

[31] Antonino Masaracchia, Vishal Sharma, Muhammad Fahim, Octavia A. Dobre, and Trung Q. Duong. 2023. Digital twin for open RAN: Toward intelligent and resilient 6G radio access networks. *IEEE Communications Magazine* 61, 11 (2023), 112–118. DOI: https://doi.org/10.1109/MCOM.003.2200879

[32] Bassam Moussa, Mourad Debbabi, and Chadi Assi. 2016. A detection and mitigation model for PTP delay attack in an IEC 61850 substation. *IEEE Transactions on Smart Grid* 9, 5 (2016), 3954–3965.

[33] Bassam Moussa, Marthe Kassouf, Rachid Hadjidj, Mourad Debbabi, and Chadi Assi. 2019. An extension to the precision time protocol (PTP) to enable the detection of cyber attacks. *IEEE Transactions on Industrial Informatics* 16, 1 (2019), 18–27.

[34] Bassam Moussa, Chantale Robillard, Alf Zugenmaier, Marthe Kassouf, Mourad Debbabi, and Chadi Assi. 2018. Securing the precision time protocol (PTP) against fake timestamps. *IEEE Communications Letters* 23, 2 (2018), 278–281.

[35] Esteban Municio, Gines Garcia-Aviles, Andres Garcia-Saavedra, and Xavier Costa-Pérez. 2023. O-RAN: Analysis of latency-critical interfaces and overview of time sensitive networking solutions. *IEEE Communications Standards Magazine* 7, 3 (2023), 82–89.

[36] NIS Cooperation Group. 2022. *Report on the cybersecurity of Open RAN*. Technical Report. Retrieved from https://digital-strategy.ec.europa.eu/en/library/cybersecurity-open-radio-access-networks

[37] Nokia. 2025. *Open RAN Security*. Technical Report. Nokia.

[38] O-RAN Working Group 11. 2023. *Security Requirements Specifications*. Technical Report. O-RAN.WG11.Security-Requirements-Specification.O-R003-v06.00.

[39] O-RAN Working Group 11. 2024. *O-RAN Security Test Specifications*. Technical Report. O-RAN.WG11.SecTestSpecs.0-R003-v07.00.

[40] O-RAN Working Group 11. 2024. *O-RAN Security Threat Modeling and Risk Assessment*. Technical Report. O-RAN.WG11.Threat-Modeling.O-R003-v03.0.

[41] O-RAN Working Group 4. 2023. *O-RAN Fronthaul Control, User and Synchronization Plane Specification v12*. Technical Report. O-RAN.WG4.CUS.0-R003-v12.00.

[42] Department of Defense. 2020. *5G Strategy Implementation Plan*. Technical Report. Department of Defense. Retrieved February 15, 2024 from https://apps.dtic.mil/sti/pdfs/AD1118833.pdf

[43] OpenAirInterface Software Alliance. 2024. Retrieved October 30, 2024 from https://openairinterface.org

[44] M. Polese et al. 2024. Colosseum: The open RAN digital twin. In *IEEE Open Journal of the Communications Society* 5 (2024), 5452–5466. DOI: 10.1109/OJCOMS.2024.3447472

[45] Michele Polese, Leonardo Bonati, Salvatore D'oro, Stefano Basagni, and Tommaso Melodia. 2023. Understanding O-RAN: Architecture, interfaces, algorithms, security, and research challenges. *IEEE Communications Surveys & Tutorials* 25, 2 (2023), 1376–1411.

[46] Rapid7 Global Consulting. 2018. *UNDER THE HOODIE, Lessons from a Season of Penetration Testing*. Technical Report. Rapid7. Retrieved from https://www.rapid7.com/globalassets/_pdfs/research/rapid7-under-the-hoodie-2018-research-report.pdf

[47] Filip Rezabek, Max Helm, Tizian Leonhardt, and Georg Carle. 2022. PTP security measures and their impact on synchronization accuracy. In *Proceedings of the 2022 18th International Conference on Network and Service Management (CNSM)*. IEEE, 109–117.

[48] Ezzeldin Shereen, Florian Bitard, György Dán, Tolga Sel, and Steffen Fries. 2019. Next steps in security for time synchronization: Experiences from implementing IEEE 1588 v2. 1. In *Proceedings of the 2019 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication (ISPCS)*. IEEE, 1–6.

[49] Shanghao Shi, Yang Xiao, Changlai Du, Md Hasan Shahriar, Ao Li, Ning Zhang, Y. Thomas Hou, and Wenjing Lou. 2023. MS-PTP: Protecting network timing from byzantine attacks. In *Proceedings of the 16th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. 61–71.

[50] Kashyap Thimmaraju, Altaf Shaik, Sunniva Flück, Pere Joan Fullana Mora, Christian Werling, and Jean-Pierre Seifert. 2024. Security testing the O-RAN near-real time RIC & A1 interface. In *Proceedings of the 17th ACM Conference on Security and Privacy in Wireless and Mobile Networks* (Seoul, Republic of Korea) *(WiSec '24)*. Association for Computing Machinery, New York, NY, USA, 277–287. DOI: https://doi.org/10.1145/3643833.3656118

[51] Walter Tiberti, Eleonora Di Fina, Andrea Marotta, and Dajana Cassioli. 2022. Impact of man-in-the-middle attacks to the O-RAN inter-controllers interface. In *Proceedings of the 2022 IEEE Future Networks World Forum (FNWF)*. IEEE, 367–372.

[52] Davide Villa, Imran Khan, Florian Kaltenberger, Nicholas Hedberg, Ruben Soares da Silva, Anupa Kelkar, Chris Dick, Stefano Basagni, Josep M. Jornet, Tommaso Melodia, Michele Polese, and Dimitrios Koutsonikolas. 2024. An open, programmable, multi-vendor 5G O-RAN testbed with nvidia ARC and OpenAirInterface. *Proceedings of the 2nd IEEE Workshop on Next-generation Open and Programmable Radio Access Networks (NG-OPERA)* (2024).

[53] Marcus Wong, Anand Prasad, and Anthony C. K. Soong. 2022. The security aspect of 5G fronthaul. *IEEE Wireless Communications* 29, 2 (2022), 116–122.

[54] Jiarong Xing, Junzhi Gong, Xenofon Foukas, Anuj Kalia, Daehyeok Kim, and Manikanta Kotaru. 2023. Enabling resilience in virtualized RANs with Atlas. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*. 1–15.

[55] Jiarong Xing, Sophia Yoo, Xenofon Foukas, Daehyeok Kim, and Michael K. Reiter. 2024. On the criticality of integrity protection in 5G fronthaul networks. In *Proceedings of the 33rd USENIX Security Symposium (USENIX Security 24)*. 4463–4479.

A Appendices

A.1 Additional Attack Results

In this section we present additional results demonstrating the impact of our PTP attacks in terms of reported delay. The delay during normal conditions with FH background traffic is shown in light green, while the delay caused by our attacks is shown in dark red.

Figures 14, 15, and 16 illustrate the impact of our spoofing attack. In Figure 16, the attack's impact is visible only after it ends because it stops PTP synchronization messages. The longer the attack lasts, the greater the synchronization drift. After the attack, the large spike in delay shows the synchronization drift that occurred.

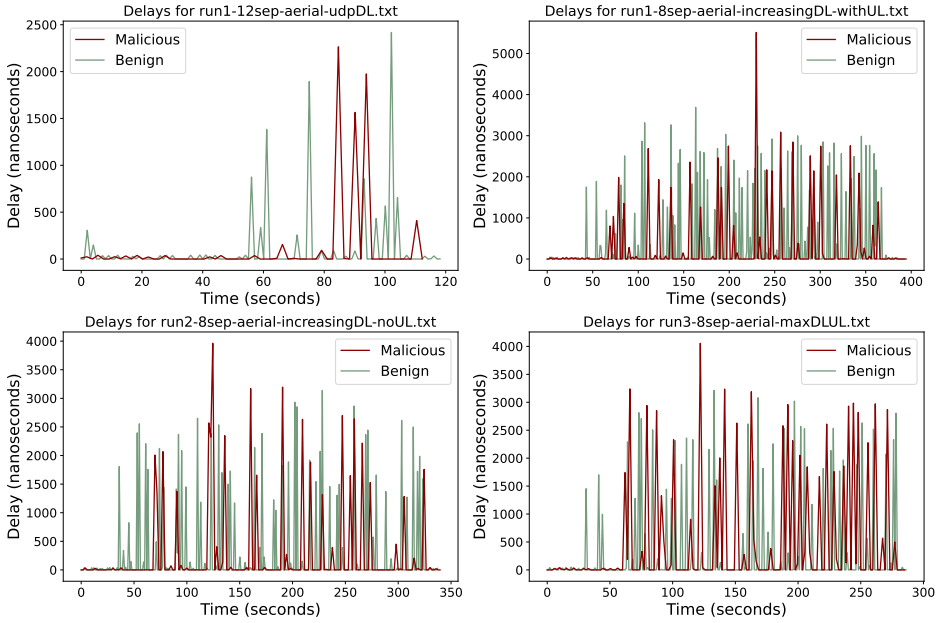


Fig. 14. Impact of the 30/30 second spoofing attack.

Figures 17, 18, and 19 show the impact of our replay attack. Unlike the announce attack, PTP traffic continues during the replay attack, allowing delay calculation throughout. However, the reported delay is far higher than the actual network delay, severely impacting PTP synchronization with reported delays over 8,000 times greater than the actual observed delay.

A.2 PLFS

In the context of the TIMESAFE framework, it is important to discuss PLFS. While FH standards [19, 38] allow for any PLFS, in practice, **Synchronous Ethernet (SyncE)** is commonly used. SyncE is a physical layer protocol [1] that enables high-precision clock synchronization by using the edges in the Ethernet data signal to define the timing content of the signal. This protocol distributes a physical layer clock across a packet network, with each system recovering and forwarding the network timing through the distribution path. SyncE is primarily utilized to assist in frequency synchronization, ensuring that all network elements are aligned to a common frequency reference.

Our production-ready environment employs SyncE, providing robust frequency synchronization to support the timing requirements of PTP. However, despite the presence of SyncE, our attack against PTP was successful, demonstrating the potential vulnerability even with physical layer

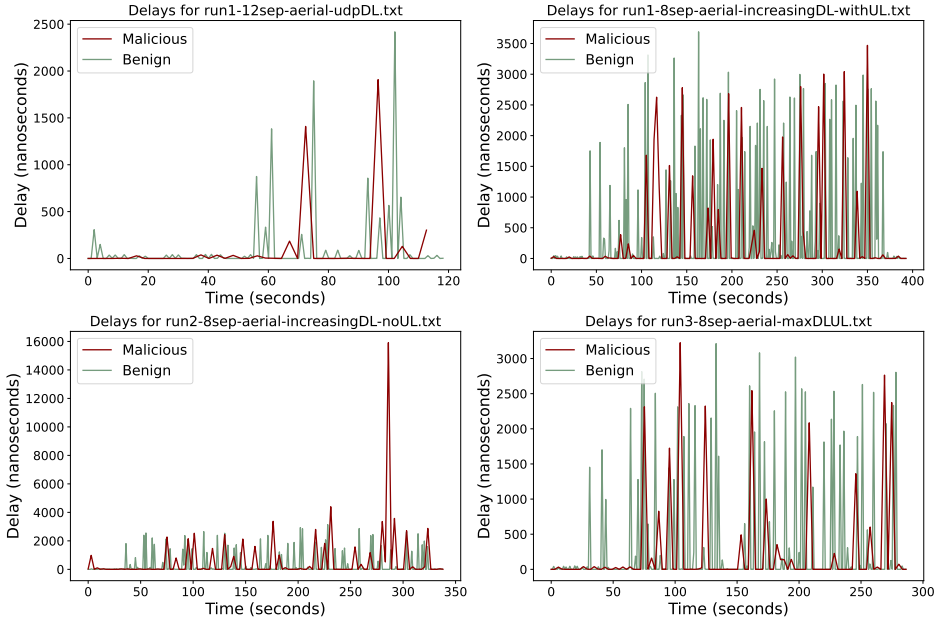


Fig. 15. Impact of the 40/20 second spoofing attack.

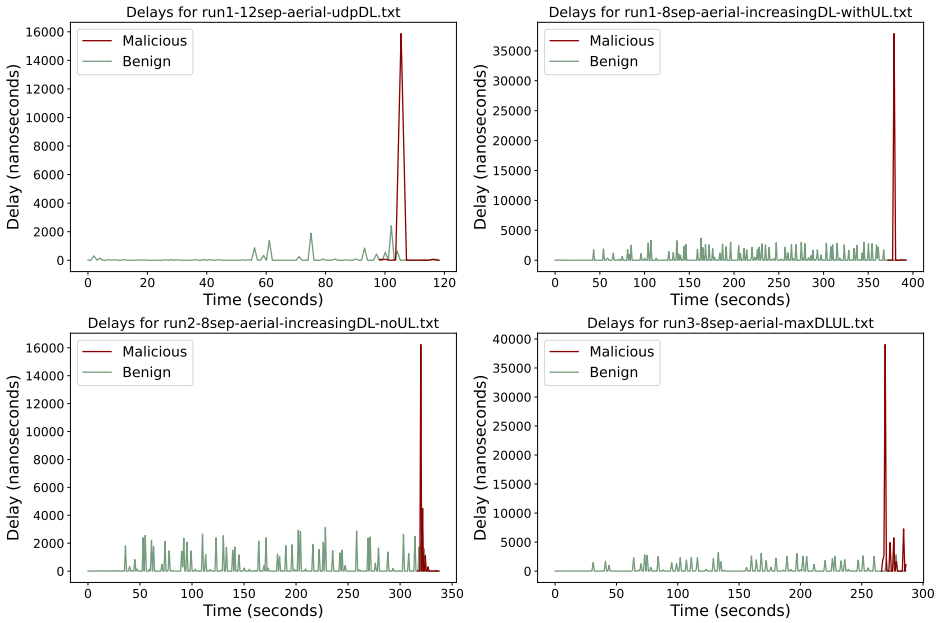


Fig. 16. Impact of the continous spoofing attack.

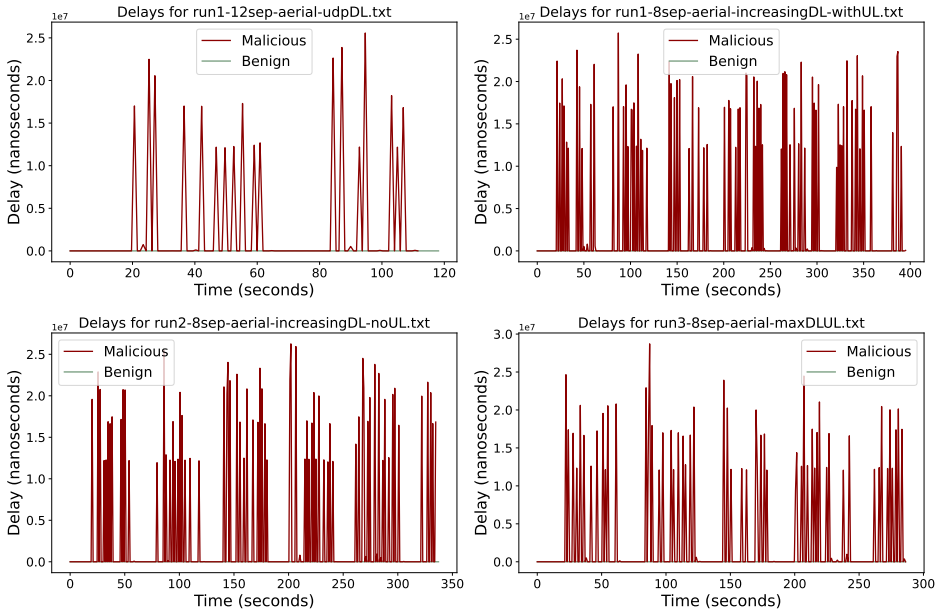


Fig. 17. Impact of the 40/20 replay attack.

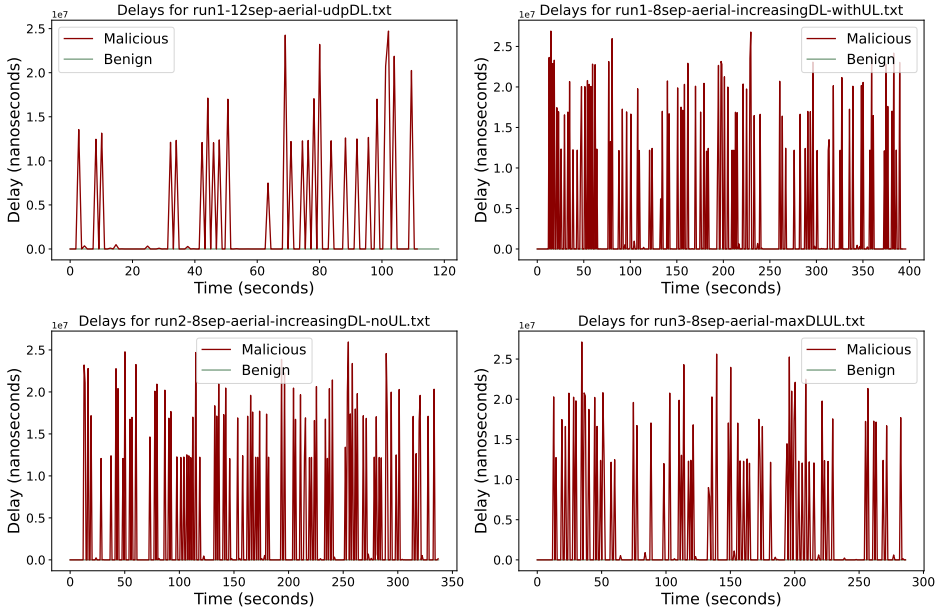


Fig. 18. Impact of the 50/10 replay attack.

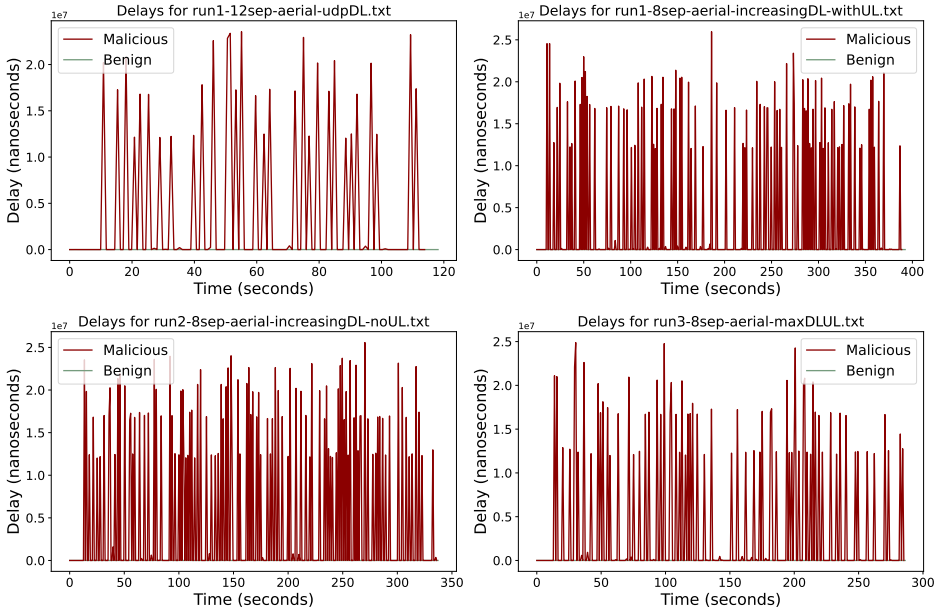


Fig. 19. Impact of the continuous replay attack.

support. Our digital twin environment does not currently use SyncE, as our focus is on evaluating the security and resilience of the higher layer PTP protocol. This approach allows us to isolate and test the specific impacts of attacks on PTP synchronization without the additional complexity introduced by physical layer frequency support.

A.3 ML Model Parameters

In this section, we give a detailed description of each model used.

Logistic Regression: The Logistic Regression model from scikit-learn is a linear model for binary classification that estimates the probability of a binary outcome using the logistic function. The model is configured with the following parameters: “penalty=l2” for L2 regularization, “dual=False” to use the primal formulation for the optimization problem, “tol=1e-4” as the tolerance for stopping criteria, “C=1.0” as the inverse of regularization strength, “fit_intercept=True” to add intercepts to the decision function, “solver=lbfgs” for optimization, “max_iter=100” as the maximum number of iterations for the solver to converge.

k-Nearest Neighbors (k-NN):. The k-Nearest Neighbors model is a non-parametric method for classification that identifies the most common class among the “k” nearest neighbors in the feature space. The model is configured with the following default parameters: “n_neighbors=5” as the number of neighbors to use for k-neighbors queries, “weights=uniform” to weight all points in each neighborhood equally, “algorithm=auto” to automatically select the best algorithm (“ball_tree”, “kd_tree”, or “brute”) based on the input data, “leaf_size=30” as the leaf size for tree-based algorithms, and “p=2” as the power parameter for the Minkowski distance metric, equivalent to the Euclidean distance.

Decision Tree Classifier: The Decision Tree model is a non-parametric supervised learning method that uses a series of decision rules inferred from data features. The model is configured with “criterion=gini” as the splitting criterion based on Gini impurity, “splitter=best” to select the

best split, "max_depth=None" to expand nodes until all leaves are pure or contain fewer than "min_samples_split" samples, "min_samples_split=2" as the minimum number of samples required to split an internal node, "min_samples_leaf=1" as the minimum number of samples required at a leaf node, "max_features=None" to consider all features for the best split, and "random_state=None" for randomization.

Gradient Boosting Classifier: The Gradient Boosting Classifier is an ensemble learning method that builds multiple decision trees sequentially, where each tree corrects the errors of the previous one. The model is configured with "n_estimators=100" as the number of boosting stages (trees), "learning_rate=0.1" as the step size for weight updates, "max_depth=3" as the maximum depth of each decision tree, and "verbose=1" to control the verbosity of the training output. Other parameters include "criterion=friedman_mse" for measuring the quality of a split, "min_samples_split=2" for the minimum number of samples required to split an internal node, "min_samples_leaf=1" for the minimum number of samples required at a leaf node, "max_features=None" to consider all features for the best split, and "random_state=None" for randomization.

Naive Bayes Classifier: The Gaussian Naive Bayes model is a probabilistic classifier that applies Bayes' theorem, assuming feature independence given the class. The model is configured with "priors=None" to use the class priors adjusted according to the data, and "var_smoothing=1e-9" to set the portion of the largest variance of all features, that is, added to variances for calculation stability.

Random Forest Classifier: The Random Forest Classifier is an ensemble learning method that combines multiple decision trees, trained on different parts of the data, for improved accuracy and reduced overfitting. The model is configured with "n_estimators=100" as the number of trees in the forest, "criterion=gini" as the splitting criterion based on Gini impurity, "max_depth=None" to expand nodes until all leaves are pure or contain fewer than "min_samples_split" samples, "min_samples_split=2" as the minimum number of samples required to split an internal node, "min_samples_leaf=1" as the minimum number of samples required at a leaf node, "max_features=sqrt" setting the number of features to consider when looking for the best split to $\sqrt{6}$, and "random_state=None" for randomization.

LSTM Classifier: The LSTM model is designed with a single LSTM layer, followed by a FC layer. The LSTM layer is configured with "batch_first=True", allowing the input and output tensors to have the shape "(batch_size, sequence_length, input_dim)", which facilitates mini-batch training. The LSTM accepts a variable sequence length input ranging from 10 to 40 packets, enabling it to produce a regular, time-based output while processing a variable number of input packets. The hidden layer's dimensionality is defined by the "hidden_dim" parameter, and the model outputs a binary classification for each input packet.

The output layer is a FC layer that maps the hidden states from the LSTM layer to a single output node for each input packet. The model uses a sigmoid activation function at the final layer to ensure the output is a probability value, making it suitable for binary classification tasks.

During training, the slice length is randomly selected from the range [10, 40] for each epoch. A custom Weighted Binary Cross-Entropy Loss function is employed to handle class imbalance and unequal error costs. For the LSTM model, the loss weight for false positives (**FP**) is kept constant at one, while the weight for false negatives (**FN**) is varied between 1 and 10, achieving maximum recall with a value of 7. The Adam optimizer is used with an initial learning rate of 0.001, and a StepLR scheduler reduces the learning rate by a factor of 0.1 every 10 epochs to help the model converge more effectively. Early stopping is enabled by monitoring validation loss with a patience parameter of 20 epochs.

CNN Model: The Convolutional Neural Network model is designed with a 2D convolutional architecture. The model processes input sequences of fixed length $S = 32$ with features $F = 6$, formatted as a single-channel image of dimensions $S \times F$. The model consists of two convolutional layers.

The first convolutional layer applies 16 filters, each with a kernel size of 3×3, using a stride of one and padding of one to preserve the input's spatial dimensions. This is followed by a ReLU activation function and a max pooling operation with a 2×2 kernel, which reduces the spatial dimensions by a factor of two. The second convolutional layer uses 32 filters, again with a 3×3 kernel, and follows the same ReLU activation and max pooling steps. After the second layer, the feature map dimensions are further reduced.

The output from the convolutional layers is flattened to create a vector input for the FC layers. The first FC layer contains 64 neurons with a ReLU activation function, and the final FC layer outputs a single value, which is passed through a sigmoid activation function to produce a binary classification. The model is trained to minimize a custom Weighted Binary Cross-Entropy Loss function to handle class imbalance and uneven costs of false positives and false negatives, although we kept both weights at the default setting of one.

The Adam optimizer is employed with an initial learning rate of 0.001, and a StepLR scheduler reduces the learning rate by a factor of 0.1 every 10 epochs. Early stopping is enabled by monitoring validation loss with a patience parameter of 20 epochs.

Transformer Model: The transformer-based model leverages the encoder portion of the transformer architecture, omitting the decoder, to process input sequences with fixed lengths of $S = 16$, 32, or 40 packets during training and evaluation. These lengths are chosen to balance the trade-off between the need to incorporate more historical data, which improves accuracy and recall, and the associated computational cost and memory requirements.

The encoder is composed of a stack of $N = 2$ layers, each layer consisting of a multi-head self-attention mechanism with n_{head} attention heads, followed by a feed-forward network. The input sequence has six features, and is normalized by a layer normalization step before being processed by the encoder layers. The encoder outputs a contextualized feature matrix of size $F \times S$, where F is the number of features and S is the sequence length.

The encoded output is then flattened and passed through a FC linear layer with 256 neurons, followed by a ReLU activation function to introduce non-linearity. After this, a dropout layer with a dropout probability of 0.2 is applied to mitigate overfitting. The final output is generated by another FC layer with a single output neuron, followed by a sigmoid activation function, to produce a binary classification result for the entire input sequence.

We treated the number of attention heads ($n_{head} = 2, 3$) and the slice length ($S = 16, 32, 40$) as hyper-parameters and tuned them for optimal recall. The model is trained using the Adam optimizer with an initial learning rate of 0.001. A StepLR scheduler is used to reduce the learning rate by a factor of 0.1 every 10 epochs, facilitating convergence by fine-tuning the learning rate during training. Early stopping is enabled by monitoring the validation loss with a patience parameter of 20 epochs. We use the same custom Weighted Binary Cross-Entropy Loss function with both weights kept constant at one.

Received 9 May 2025; revised 9 May 2025; accepted 5 November 2025