



CellularLint: A Systematic Approach to Identify Inconsistent Behavior in Cellular Network Specifications

Mirza Masfiqur Rahman, Imtiaz Karim, and Elisa Bertino, *Purdue University*

<https://www.usenix.org/conference/usenixsecurity24/presentation/rahman>

**This paper is included in the Proceedings of the
33rd USENIX Security Symposium.**

August 14-16, 2024 • Philadelphia, PA, USA

978-1-939133-44-1

**Open access to the Proceedings of the
33rd USENIX Security Symposium
is sponsored by USENIX.**

CellularLint: A Systematic Approach to Identify Inconsistent Behavior in Cellular Network Specifications

Mirza Masfiquir Rahman*, Intiaz Karim*, Elisa Bertino
Purdue University

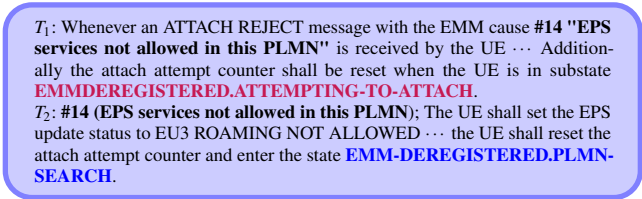
Abstract

In recent years, there has been a growing focus on scrutinizing the security of cellular networks, often attributing security vulnerabilities to issues in the underlying protocol design descriptions. These protocol design specifications, typically extensive documents that are thousands of pages long, can harbor inaccuracies, underspecifications, implicit assumptions, and internal inconsistencies. In light of the evolving landscape, we introduce CellularLint—a semi-automatic framework for inconsistency detection within the standards of 4G and 5G, capitalizing on a suite of natural language processing techniques. Our proposed method uses a revamped few-shot learning mechanism on domain-adapted large language models. Pre-trained on a vast corpus of cellular network protocols, this method enables CellularLint to simultaneously detect inconsistencies at various levels of semantics and practical use cases. In doing so, CellularLint significantly advances the automated analysis of protocol specifications in a scalable fashion. In our investigation, we focused on the Non-Access Stratum (NAS) and the security specifications of 4G and 5G networks, ultimately uncovering 157 inconsistencies with 82.67% accuracy. After verification of these inconsistencies on 3 open-source implementations and 17 commercial devices, we confirm that they indeed have a substantial impact on design decisions, potentially leading to concerns related to privacy, integrity, availability, and interoperability.

1 Introduction

Cellular networks have emerged as the primary means of communication in the modern world, with 4G and 5G being the latest commercially available versions. Ranging from mobile communications to IoT devices to vehicular communications, cellular networks are a critical infrastructure for the digital world. With more than 1.4 billion subscribers, 5G is expected to exceed its much larger predecessor, 4G, and is forecast to gain more than 13 billion subscribers globally by the end of

*Equal contribution. The student author's name is given first.



T_1 : Whenever an ATTACH REJECT message with the EMM cause #14 "EPS services not allowed in this PLMN" is received by the UE ... Additionally the attach attempt counter shall be reset when the UE is in substate **EMM-DEREGISTERED.ATTEMPTING-TO-ATTACH**.
 T_2 : #14 (EPS services not allowed in this PLMN); The UE shall set the EPS update status to EU3 ROAMING NOT ALLOWED ... the UE shall reset the attach attempt counter and enter the state **EMM-DEREGISTERED.PLMN-SEARCH**.

Figure 1: An example inconsistency identified by CellularLint—two different sub-state transitions for same precondition. T_1 is from section 5.5.1.1 and T_2 is from 5.5.1.3.5 of TS 24.301.

2026 [4, 5]. These networks embody large infrastructure and support different features, including backward compatibility, interoperability, and heterogeneity.

Like any large-scale, complex, layered system, cellular network designs are governed by protocol documentation. 3GPP, the 3rd Generation Partnership Project—a consortium of vendors, is the supervising body that handles the design and distribution of these protocol documents [2]. These documents are quite lengthy, developed by many stakeholders, and improved through releases and change requests over the years. Incorporating these changes and releases subsequently introduces diverging descriptions. Recently, few studies have shown, albeit through manual analysis during implementation testing, that protocol documents have inconsistent descriptions accounting for differing design possibilities and undefined behaviors in real-world devices, often with severe consequences [38, 54]. Fig. 1 shows an example scenario for diverging sub-state transitions on the same condition, found in the 4G Non-Access-Stratum (NAS) specification. Furthermore, in most cases, one of the inconsistent or conflicting behaviors is less secure than the other. For example, if one design suggests clearing some network identifier(s) after a connection has been lost while the other design suggests keeping it without specifying a clear motivation, the latter can make the user vulnerable to privacy violation attacks. Therefore, these diverging specifications defined in the standards can have severe security and privacy consequences.

Prior Research. Although prior research works [24–26, 35, 37, 38, 44, 46, 48, 54] have proposed approaches for detecting security and implementation flaws in cellular network protocols, they have at least one of the following limitations: (A) They are completely manual [26, 38, 44, 48] and inherently have limited scalability. (B) They employ formal verification [35, 37], foundationally relying on the quality of the properties, and do not primarily deal with inconsistent descriptions—subsequently choosing the most secure option when an inconsistent or confusing situation is encountered. (C) They focus on implementation testing [45, 54] and only report a few inconsistent descriptions on the way without proposing an approach to uncover them. (D) They use differential testing for noncompliance checking [38] without focusing on conflicting behavior detection in the standards. (E) They either analyze change requests or specifications through NLP techniques to understand security hazards [24, 25] without directly considering differing description analysis in specifications.

Problem. With the current state-of-the-art approaches having above-mentioned limitations, in this paper, we aim to take the first steps to answer the following research question- *Is it possible to develop a systematic, generalizable, and scalable specification analysis framework that can identify inconsistent descriptions from 4G and 5G specifications and associate them with differential design choices?*

A promising direction. Following the major advances in natural language processing (NLP) [20, 61], recent works have demonstrated the automated generation of Finite State Machine from both Request For Comments (RFC) documents and cellular network specifications [40, 53]. Such results indicate that a data-driven approach can be taken to discover inconsistencies in specifications as a root cause of design flaws and vulnerabilities as well.

Challenges. Detecting inconsistent behaviors systematically from the large and complex cellular network specifications using NLP requires tackling some very important research challenges. The first and foremost challenge is to quantize the specification into *segments*, which we can use for inconsistency detection. The specifications explain an event in a very large context; in some cases, an event is described in 5-6 pages. With such a large segment, any learning algorithm would fail to capture the finer details and attributes of the segment. Therefore, we need to devise a mechanism to create segments of reasonable size from the specifications. Second, the specifications are so large that in case all the segments are pairwise compared, the search space becomes intractable. For this, we need to devise an approach that reduces the search space substantially. Last, and most importantly, to the best of our knowledge, there are no datasets that could be utilized to detect inconsistent statements in cellular networks. In case there is a large number of inconsistencies that can be manually found, this trivially solves the problem and does not require a systematic framework. Fortunately or rather unfortunately, there are very few manually found conflicts in the specifica-

tions that can be used for closed-form solutions that we can use to train. This lack of ground truth creates a major hurdle for any learning-based solution.

Our Approach. In this paper, we introduce CellularLint—the first framework for uncovering inconsistencies from the upper layer of 4G and 5G specifications. For our method, we employ few-shot domain adaptation of language models.

To obtain a scalable solution, we initially cast the documents in sizeable segments based on domain-specific knowledge. These quantized segments preserve the context and event, which are essential before understanding inconsistencies. Second, based on this quantized pool of intra-document and inter-document test cases/segments, we produce *Pairs of Segments (PoS)*. CellularLint compresses the search space of PoS into a manageable and informative dataset based on interpretable similarity measures. This guides our method to look into only relevant content—making the process highly scalable.

To address the lack of labeled examples, instead of heavily relying on expert annotations, CellularLint incorporates domain-aware annotations and ensemble decision mechanisms and takes only a fraction of labeled examples. We design seven domain-adapted, consistency-wise meaningful labels and map them to general-purpose natural language inference (NLI) annotations. CellularLint also utilizes human-in-the-loop active learning on multiple language models to finalize inconsistent descriptions in varying granularities.

To address the lack of supervision, we divide the learning objective into multiple phases- (1) initially converting the learning problem to a generalized setting of NLI and then (2) using multiple phases of active learning on a domain-specific dataset.

Findings. CellularLint discovered 157 conflicting descriptions from specifications. To evaluate the effectiveness of the discoveries, we investigate them on 3 open-source implementations and test them on 17 commercial UEs.

We uncover that implementations indeed take different design choices based on conflicting or inconsistent standards. Furthermore, in some cases, we have found that implementations choose the less secure option, and in some cases, tried to implement both options. For ease of discussion, we characterize a subset of our findings into four different categories—impacting security enforcement, privacy, interoperability, and causing denial-of-service. In these categories, we discuss a total of 11 issues that can have a severe impact on the privacy, integrity, availability, and interoperability of cellular network infrastructure.

Responsible disclosure and open-source. To further improve the cellular network standards, we have responsibly disclosed all our 157 findings to the 3GPP standardization body and the affected UE vendors. Furthermore, we will completely open-source CellularLint and all the detected inconsistencies to foster research in inconsistency detection in other

important protocol specifications [13]¹.

Contributions. To summarize, we make the following contributions:

- **Framework.** We design and implement CellularLint. To the best of our knowledge, this is the first systematic approach towards inconsistency detection for both 4G and 5G specifications.
- **Few-shot learner.** We demonstrate a novel few-shot approach with active multiphase learning for cellular network-specific pre-trained language models to subsidize insufficient labeled data effectively.
- **New findings.** We found 157 inconsistent descriptions and investigated the results on 3 open-source and 17 commercial device implementations.

2 Background

In this section, we introduce various preliminaries ranging from the 4G Long Term Evaluation (LTE) and 5G New Radio (NR) architectures to various NLP methods that are relevant to our work.

2.1 Cellular Network Preliminary

For cellular network preliminaries, here we discuss the most common components that are relevant to this paper.

2.1.1 Cellular Network Architecture

The network is composed of three main components: the user equipment (UE), the radio access network (RAN), and the core network. The architectural difference between 4G and 5G mostly lies on the RAN and Core Network side.

User Equipment (UE). The UE, a terminal device on the user end, is equipped with a Universal Subscriber Identity Module (USIM). The USIM contains the user identifier, master secret key, and session key for connectivity. These are essential for mutual Authentication and Key Agreement (AKA) between the user and the network. The most common UE are cell phones, tablet computers, and IoT devices with cellular connectivity.

eNodeB. eNodeBs or eNBs are base stations or, most commonly, mobile network towers in the 4G architecture. They stand as a middle entity in the connection establishment and maintenance for which the Radio Resource Control (RRC) protocol is implemented. eNodeB is replaced by gNodeB in the 5G network. eNodeB is responsible for transmitting and receiving signals to and from UEs.

gNodeB. The gNodeB or gNB—also called the Next-Generation NodeB—is used in 5G network. It is a significant upgrade from eNodeB. gNodeB supports Massive MIMO

¹<https://cellularlint.github.io/>

technology, has higher throughput, lower latency, and enables advanced network slicing.

4G-Core The core network in 4G LTE is called Evolved Packet Core (EPC). Two functional components—Mobility Management Entity (MME) and Home Subscriber Server (HSS) are relevant to our paper; hence we briefly discuss them in what follows.

- **Mobility Management Entity (MME).** The MME is responsible for handling signals between the UE and the core, as well as the eNodeB and the core. It is responsible for UE authentication, detach procedure, tracking area updates, etc.
- **Home Subscriber Server.** HSS works as a database to store subscriber information (IMEI and IMSI). It also provides relevant information for calls and IP sessions.

5G-Core. The 5G Core comprises many functional components such as the Access and Mobility Management Function (AMF), Session Management Function (SMF), User Plane Function (UPF), Authentication Server Function (AUSF), and so on. 5G leverages a large number of antennas on the core side to enhance signal quality.

2.2 ML Preliminaries

CellularLint detects inconsistencies from a natural language perspective, meaning that it focuses on the lexical and contextual analysis of the salient features of the protocol text documents to reveal inconsistencies from protocol specifications. Here, we discuss the NLP preliminaries required for CellularLint.

2.2.1 Few Shot Learning

Few-Shot Learning (FSL) [19,31] is a meta-learning approach that utilizes a pre-trained model to generalize over new categories of data using only a few labeled examples per class. For N -way- k -shot learning in NLP, given k labeled examples of each class from N classes for a new NLP task where k is generally very low, the pre-trained model has to learn efficiently to solve the new task.

2.2.2 Active Learning

Active learning [30, 52], especially human-in-the-loop active learning, is a machine learning paradigm that focuses on improving learning performance by minimizing the need to label large amounts of data. Often, domain-specific supervised learning is hindered by the insufficiency of ground truth data. Guiding the model by associating expert annotation with a subset of important data points can largely boost the performance of supervised training.

2.2.3 NLP Methods

Textual Entailment (TE). Textual entailment, or Natural Language Inference (NLI), is a binary relation where two text sequences can either form an agreement, a contradiction or be completely irrelevant to each other [17, 39]. From a NLP perspective, given two text sequences, where the first is a hypothesis and the second is a premise, the model has to find out whether the first text is in agreement with the second (or vice versa) or is in contradiction or has no logical relation.

Language Modeling. Recent years have seen revolutionary improvements in machine learning on textual data, much of which is due the use of transformers [28, 49, 62, 64, 66]. Transformers are widely used for sequence-to-sequence modeling, semantic parsing, machine translation, question-answering, and most recently for large language models. Transformers rely on the attention mechanism, which excels at learning complex linguistic patterns effectively through the understanding of each word's importance and its ability to identify its surrounding words, thereby characterizing the context.

3 Overview

In this section, we discuss the problem analysis scope and state the problem formally. Next, we discuss the challenges and corresponding approaches.

3.1 Scope of Analysis

The cellular network protocols (4G and 5G) comprise thousands of specifications—from the low layers to the link layer to the upper layers (Layer 3) of the protocols. Among all these specifications and related procedures, we focus on the NAS layer procedures. NAS layer procedures contain the mobility management and session management components, which in turn manage the most critical control plane procedures such as connection setup, initial authentication, mobility, hand-off, and service notifications. Previous works have shown several vulnerabilities in these components, resulting in severe security and privacy issues such as authentication-bypass [45], location exposure [36], impersonation [57], downgrading [41], to name a few. Keeping these in mind, in this work we focus on the NAS layer procedure. More specifically, we focus on four documents in two categories- the NAS technical specification TS (24.301, v17.6.0) [9] and the Security Architecture and Procedures specification (TS 33.401, v17.1.0) [7] for 4G, and the NAS specification (TS 24.501, v17.7.1) [8] and the Security Architecture and Procedures specification (TS 33.501, v17.5.0) [3] for 5G. All of them are available from the 3GPP archive. Release 17 is the latest complete version to have both NAS and its security specifications. Furthermore, having the same release ensures consistency and removes the possibility of unexpected inconsistency prediction (false positives) in-

roduced by version mismatch. In fact, finding cross-version inconsistency is a different task that we leave as future work.

3.2 Problem Formulation

Here we define the problem formally: given a set of corpora $\mathcal{G} = \{C_{Nas}^{Ag}, C_{Sec}^{Ag}, C_{Nas}^{5g}, C_{Sec}^{5g}\}$ (where C_{Nas}^{Ag} is the corpus produced from 4G NAS specification, C_{Sec}^{Ag} is the corpus produced from 4G security architecture specification, similar notations are applicable for 5G), we need to quantize each corpus in text segments, that is, $C_i = \{t_1, t_2, \dots, t_n\}_i$ where t_a^i describes a -th meaningful event/sub-event/directive from corpus C_i . Now our first objective is to construct intermediate sets, $\mathcal{S}_1 = (C_{Nas}^{Ag} \times C_{Nas}^{Ag}) \cup (C_{Nas}^{Ag} \times C_{Sec}^{Ag}) \cup (C_{Sec}^{Ag} \times C_{Sec}^{Ag})$ and $\mathcal{S}_2 = (C_{Nas}^{5g} \times C_{Nas}^{5g}) \cup (C_{Nas}^{5g} \times C_{Sec}^{5g}) \cup (C_{Sec}^{5g} \times C_{Sec}^{5g})$. Next, from each intermediate set $\mathcal{S}_m \in \{\mathcal{S}_1, \mathcal{S}_2\}$ containing tuples or pairs (t_a^i, t_b^j) of text Segments (PoS), our objective is to devise a model \mathcal{M} which can determine all pairs that conflict between each other, that is, $\mathcal{M}(t_a^i, t_b^j) = k$. Here, k takes the value *consistent* or *inconsistent*.

3.3 Challenges & Solution Outline

We now discuss the main challenges and corresponding approaches that have driven the CellularLint's high-level design choices. We then present the key steps in our overall solution. **(C1) Segment quantization.** In numerous scenarios, the specifications explain an event in a vast context. For example, in the 4G NAS specification, Section 5.5.1.2.4 describes "Attach accepted by the network", which is a six-page long description concerning one major event where the network accepts an attach request from the UE. If we consider this whole event as one complete and discrete, meaningful *segment*, any learning algorithm will fail to capture the intricate details and attributes, causing differential and anomalous behaviors. Moreover, in the same section, there is a precondition "If the UE indicates support for EMM-REGISTERED without PDN connection", which shall trigger an event on the MME side. A similar precondition is found for `tracking_area_update_request` message further away in the document. If the whole section is considered as a potential segment, one would not be able to match such sub-event triggers and may miss numerous possible interesting scenarios. An alternative choice can be selecting each sentence as an atomic unit governing a segment. However, this leads to a serious problem because, often, a single sentence from the specification is not sufficient to infer all the entities, states, or messages involved in a process. Also, due to the semi-structured nature of specifications, many sentences are repeated in different parts of the specifications. Thus, considering each sentence a meaningful segment would produce too many repeating units in the dataset, and if needed, it would be impossible to map back uniquely to the specification. Thus, this trivial solution is not applicable to

our case.

(A1) Sub-event driven segment quantization. As we have already established that considering a whole event as a segment likely affects the effectiveness and purpose of detected inconsistencies, we quantize such larger descriptions into smaller segments. However, in doing so, we preserve the completeness of the description of sub-events. We combine domain-specific understanding with standard NLP techniques to pinpoint sub-events. Specifically, we consider subsections and complete paragraphs as atomic units while constraining the sequence length and entity based on Parts-Of-Speech filtering. Details about the approach for segment quantization are given in §4.2.

(C2) Intractable search space. The protocol specifications are large documents defining and explaining all entities, events, state transitions, message structures, and so on in finer detail. Moreover, when we execute segment quantization to address C1, the resulting quantized dataset is even larger. Thus, traversing and matching texts directly using the entire dataset of quantized segments makes the finalized search space intractable with millions of potential segment pairs. For example, after the segment quantization, the brute-force traverse and match solution gives us more than *12 million* segment pairs for 5G-NAS specification alone. On top of that, to find inconsistencies, one may have to consider multiple related specifications rather than generating segments from just one. For instance, in our case, for 5G we consider the 5G NAS technical specifications and the security architecture and procedures specification. This, in turn, makes the search space tremendously larger and thereby causes a state-space explosion.

(A2) Finite segment filtration. To address C2, we reduce the segment sequence pair space by filtering the initial dataset based on similarity measures. The intuition behind this idea is that two completely different segments should not be considered inconsistent as they talk about totally different aspects of the protocol. Contrary to this, an inconsistent PoS would have at least the same pre-condition and hence would have a better similarity score. To achieve such a quantitative measure of texts, we first consider the vector embedding of texts. However, we observe that choosing an arbitrary embedding can severely affect the search space shrinkage and effectiveness of similarity scoring. We have thus carried out an extensive evaluation to determine the most reliable text vectorization technique for our problem. The details of the evaluation are discussed in §6.1. Thus we reduce the number of segments from millions to a few thousand—reducing the problem space to a tractable size.

(C3) Absence of ground truth. To the best of our knowledge, there are no datasets that could be utilized for NLP-based machine learning to detect conflicting statements from cellular protocols. Even if a processed, readily usable dataset existed, no closed-form solution would be able to trivially map text segments from multiple documents to specific inconsistencies.

This is a novel problem—solving which is a key contribution of our work. Moreover, although state-of-the-art language models are very effective in learning contexts, they require a massive amount of labeled data due to the larger parameter landscape. Acquiring such a domain-specific dataset is often cumbersome and for this problem setup, even harder. Thus we have to design a technique that can work with only a handful of labeled examples.

(A3) Ground truth generation. We employ multi-phase supervision through human-in-the-loop active learning to subdue the insufficiency of ground truth. Our learning method neither requires a large amount of annotation nor suffers from too much uncertainty introduced by insufficient learning. In short, we first establish the learning landscape using a general-purpose dataset with some supervised data. Note that this general-purpose dataset is not from the cellular network domain but rather from image-crowdsourced caption writing and is publicly available. This intermediary model is utilized in consensus with human involvement alongside minimal synthesized data repeatedly to prepare the finalized model. The whole process never requires the complete ground truth of our problem. In fact, having a complete ground truth from the beginning is fundamentally contradictory here, as it would trivially solve the problem we have at hand. Indeed, if there were already a large number of inconsistent descriptions in specifications, then those could be directly reported to improve the protocol and would not require any additional analysis. Likewise, the lack of ground truth makes the problem both challenging and rewarding to some extent.

3.4 Approach Skeleton

Based on our discussion of challenges and approaches to the solution, we divide the inconsistency detection problem into multiple steps: (i) We employ domain-specific preprocessing alongside standard NLP techniques to quantize each $C_i \in \mathcal{G}$. (ii) To produce each S_m , we utilize a syntactic and contextual equivalence metric ψ to filter out irrelevant PoS. (iii) To address the absence of labeled examples, we use \mathcal{M}' instead of \mathcal{M} which first learns from a general dataset \mathcal{P} to solve textual entailment task. In this task, for any two text sequences $t_x, t_y \in \mathcal{P}$, \mathcal{M}' learns if t_y entails t_x or not. Formally, $\mathcal{M}'(t_x, t_y) = k'$ where k' takes the value of either entailment or contradiction or irrelevance (often called neutral). Subsequently we use \mathcal{M}' with few domain-specific labeled data to generate our specialized model \mathcal{M} with the learning objective of $\mathcal{M}(t_a^i, t_b^j) = k'$, where $t_a^i, t_b^j \in S_m$.

4 Detailed Design

In this section, we discuss CellularLint in detail. Fig. 2 shows the main components of our framework.

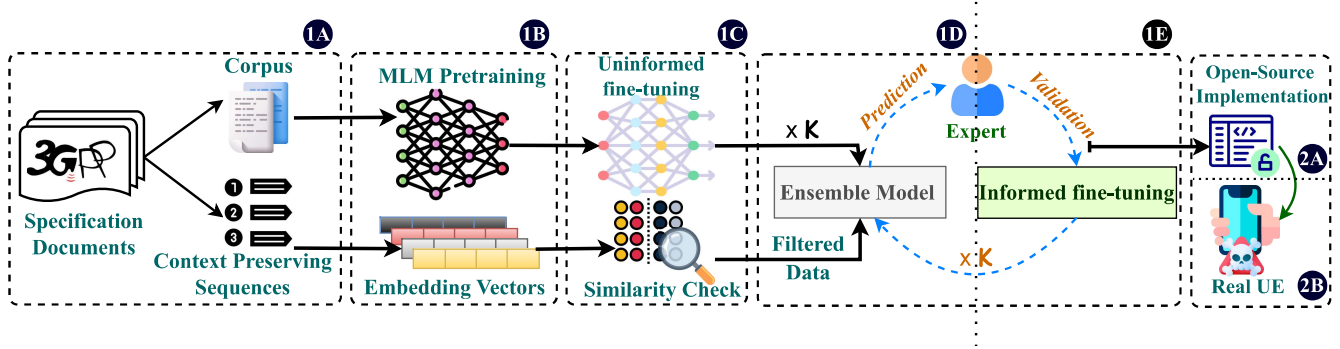


Figure 2: Architecture of CellularLint. The two-arrow process shown in 1A, 1B, and 1C represents independent and parallel processing.

4.1 Architecture

The overall framework is organized into two parts: 1 the Learner, and 2 the Dispatcher. Each of them has multiple submodules shown in Figure 2.

Learner. The learner consists of five sub-modules.

1A The *first* is preprocessing, which performs standard NLP preprocessing as well as domain-specific and task-specific preprocessing of all cellular network protocol specifications from the 3GPP archive. This generates a large corpus for our pre-training step. Here, a second-order preprocessing is also executed on the documents of our problem scope (NAS and Security of 4G and 5G), which produces the sub-event-oriented, context-preserving text segments. We call them context-preserving as when extracting them, our method makes sure that they do not describe multiple sections and the segments are not trimmed off halfway through an event description. For example, if a section is short and within our token limit (such as 5.3.10 in TS 24.301), we keep it as one segment. Alternatively, if a section contains multiple paragraphs (such as 5.5.3.2.3 from TS 24.301), each describing an independent event (they might be related when taken in a very large context), we consider each paragraph as a different segment. Details of the module can be found in §4.2.

1B The *second* sub-module is the pre-trainer. In this sub-module, we utilize the large corpus generated from the preprocessing module. This ensures that the model understands the domain with specific vocabularies and semantics. The detailed experimental setup is discussed in §5. In parallel to the training, we create embedding vectors for each segment. The embedding vectors are utilized for pairing and filtration of PoSs in the next sub-module.

1C In the *third* sub-module, we execute the uninformed fine-tuning. This is an important step for weak supervision and our first concrete endeavor toward task-specific learning. In short, we first fine-tune our candidate transformer models separately on the well-known Stanford Natural Language Inference (SNLI) corpus [18]. We call it uninformed supervision because the SNLI corpus contains examples that are not

completely in accordance with our definition of consistency from a protocol perspective. For example, "A group of people are ice skating in a big city." and "The people are outside skating." are labeled as entailments in that corpus. Apparently, the second sequence is a vague representation of the first, i.e., a large and significant part of the first sequence is not expressed through the second sequence. Moreover, our main objective is to capture the protocol inconsistencies based on complex text patterns and intricate details of various cellular events for which fine-tuning on SNLI corpus is not solely sufficient. Thus, training on the SNLI corpus only gives us a loose estimate of our objective model. Nonetheless, this uninformed supervision is the first stepping stone in our analysis. In parallel, we complete the segment *pairing and filtration* based on the embedding vectors generated in the previous step. In short, we create all possible pairs of the segment vectors and filter them based on a similarity measure. Details of the method can be found in §4.3

1D At the *fourth* sub-module, we combine the predictions of k models on our dataset. Note that these k models are now fine-tuned on SNLI corpus from the previous step. It is known that different models have the ability to capture different semantics in varying capabilities. Therefore, combining their predictions through majority voting allows us to boost the prediction confidence. However, these models have not learned what consistency means from the perspective of cellular networks. Therefore, these predictions are not completely reliable as final output.

1E The most important sub-module for the learner is the multi-phase informed fine-tuning, which is the *fifth* and most crucial step. Here, we first take the ensemble prediction from the previous sub-module. Next, we sample a small subset of data from the predicted set, then validate and rectify the labels through domain-expert human annotators. This approach ensures ground truth addition while keeping the human involvement minimal. Also, rectifying annotations instead of annotating from scratch reduces the human effort even more. The models are again trained on the reconditioned examples in a supervised manner and used subsequently for predictions,

again on the whole dataset. In such a manner, we complete one phase of training. This method is followed k times (each defining a phase) before finalizing the prediction. Details on the experimental choices of this step can be found in §5.

Dispatcher. We take the high-confidence predictions as our results after multi-phase informed training and manually analyze them. To further show the impact of the inconsistent behaviors of the specifications, we use the dispatcher. The dispatcher has two sub-modules.

2A In the *first* one, we map the discovered inconsistent descriptions to the open-source implementations—srsRAN, open5GS, and OpenAirInterface [10–12]. Note that many boundary cases and uncommon events are not available (or available in non-granularity) in the open-source implementations. Even so, the predicted inconsistent sets are checked against these multiple sources to determine their design choices and security implications. Consequently, we create a subset of inconsistencies for the next sub-module.

2B In the *second* sub-module of the dispatcher, we consider each of the inconsistencies gathered from the previous step to determine whether they cause issues in real-world devices. Details of the setup can be found on our website [13].

4.2 Dataset Preparation

It has been shown that pre-training Large Language Models (LLM) can help understand domain-specific terminologies better than LLMs with no domain-specific pre-training (i.e., only pre-trained on general datasets such as Wikipedia corpus) [19, 22, 32, 55]. We first process the raw specifications from the 3GPP archive and pre-train language models on it to leverage the domain-specific learning in an enhanced capability. Our textual entailment task is a harness over the aforementioned pre-trained model. The details of the dataset preparation are discussed in what follows.

We remove the tables, figures, cross-document references, code segments, and additional ill-formed texts from the specifications to create the pre-training corpora. For the downstream fine-tuning, since we need semantically meaningful segments of texts to compare, we first extract section-wise texts. The sections usually comprise of many sub-descriptions based on cause values. Otherwise, each paragraph in a section is usually self-contained. Also, to conform to the sequence length limit of our candidate transformer models, we sometimes do finer fragmentation. In this case, sections 4 to 8 are considered for NAS in both 4G and 5G as the rest of the sections mostly contain definitions, abbreviations, glossary, scope, etc. For security, we consider section 4 to the annex for both 4G and 5G.

4.3 Pairing & Filtration

As discussed earlier, comparing all segments with each other for inconsistency will result in a massive search space, quan-

titatively, $\binom{N}{2}$ datapoints where N is the total number of extracted segments. To overcome this, first, we use the Term Frequency- Inverse Document Frequency (TF-IDF) to vectorize such segments (submodule **1B** in Figure 2). To answer why TF-IDF is effective here, we compare five different embedding techniques—Sentence BERT (SBERT), Doc2Vec, Universal Sentence Encoder (USE), Word2Vec, and TF-IDF [21, 47, 51, 56, 60]. Among them, TF-IDF appears to be most effective (see details in §6.1).

Considering each segment to be a document, formally for a term t found in a document $d \in D$:

$$tf(t, d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}}$$

$$idf(t, D) = -\log P(t|D) = \log \frac{n}{\sum_{d \in D: t \in d} 1}$$

$$tfidf(t, d, D) = tf(t, d) \cdot idf(t, D)$$

Here $f_{t,d}$ denotes the raw frequency of term t in document d . In our problem, d represents a text segment, and D represents the corpus. TF-IDF maps the text segments d in a k -dimensional latent space \mathcal{X} -

$$\phi: \mathcal{S} \rightarrow \mathcal{X} \quad d \in \mathcal{S}$$

In the next step, we use the cosine similarity score (ψ) to measure the similarity between each pair of TF-IDF vectors. For each vector \mathbf{x}_1 and \mathbf{x}_2 :

$$\psi(\mathbf{x}_1, \mathbf{x}_2) = \frac{\mathbf{x}_1 \cdot \mathbf{x}_2}{\|\mathbf{x}_1\| \|\mathbf{x}_2\|} = \frac{\sum_{i=1}^k \mathbf{x}_{1i} \mathbf{x}_{2i}}{\sqrt{\sum_{i=1}^k \mathbf{x}_{1i}^2 \sum_{i=1}^k \mathbf{x}_{2i}^2}}$$

We generate the symmetric matrix of all pair similarity scores. Now, from this symmetric matrix (Figure 3), we take the lower triangular half and remove the datapoints that have $\psi < \psi_{min}$ and $\psi > \psi_{max}$. We consider the following cases when choosing values for ψ_{min} and ψ_{max} :

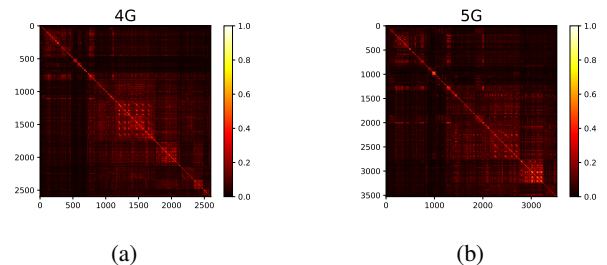


Figure 3: Heatmap of the similarity matrices from 4G and 5G. The x and y axis represent the index of text segments extracted from the specifications. A brighter cell in the matrix represents more similarity.

- When ψ_{min} is too small, the segments are essentially describing two totally different events with a few words matched in the protocol specification. For a refined dataset, ψ_{min} should be kept relatively high.

- Using a ψ_{max} helps to filter out segments where two text segments have very small changes in vocabularies, i.e., synonymous words or changes of articles ("a" instead of "the", pronouns instead of proper noun phrases, etc.). These segment pairs state the same event with (almost) exact description.

Further evaluation and discussion for the choice of ψ are in section §6.

This gives us a very useful set of filtered PoS. For supervised training, we go through another round of filtering to remove a few common PoS that are generated from the most common sentences in the documents.

4.4 Protocol Language Entailment Annotation

We now discuss the hierarchical annotations of our dataset. The key challenge in finding inconsistencies in 4G/5G protocols is that there are no ground truth labels for model training. Thus, any supervised training task is hindered. We address this challenge by multi-phase training where the 0th phase ensures the general capability of the model in distinguishing what an entailment or contradiction is, and the next phases are to precisely capture what sort of disparity is present in our actual cellular dataset.

The Stanford Natural Language Inference (SNLI) dataset contains datapoints with 3 unique annotations, namely, *entailment*, *conflict*, and *neutral* [18]. Since our model is first fine-tuned on this dataset (submodule 1C) as a 0th phase, we keep the same annotations to train the model(s) for subsequent phases. To characterize all scenarios, we consider 7 cases:

- 1: $t_1 = t_2$: t_1 is consistent with t_2
- 2: $t_1 \neq t_2$: t_1 is inconsistent with t_2
- 3: $t_1 \otimes t_2$: t_1 is not related to t_2
- 4: $t_1 \rightarrow t_2$: t_1 is related to t_2 . t_1 happens before t_2
- 5: $t_1 \leftarrow t_2$: t_1 is related to t_2 . t_2 happens before t_1
- 6: $t_1 \sqsupset t_2$: t_1 is related to t_2 . t_1 contains more/detailed information than t_2
- 7: $t_1 \sqsubset t_2$: t_1 is related to t_2 . t_2 contains more/detailed information than t_1

We argue that cases 1, 4, and 5 fall under entailment. Cases 2, 6, and 7 fall under contradiction. Case 3 maps onto the neutral label in the NLI task. For the contradiction class, cases 6 and 7 are difficult to perceive. We discuss this with a possible scenario. Suppose in a scenario both t_1 and t_2 independently describe the actions after an `attach_reject` is received by the UE with some specific EMM cause. Now, t_1 describes both the EPS status update and the security context clearing. In contrast, t_2 describes only the process of updating the EPS status but does not mention anything about security context clearing. This will fall under case 6. This is considered as an inconsistency because the missing security context clearing

in t_2 might create confusion in the implementation design and later on cause serious security issues.

4.5 NLI Adaptation

At each phase, our annotated protocol dataset contains 150 PoS (in total 450 for three phases), whereas the SNLI dataset contains 570k English sentence pairs, which are ~ 1266 times higher than the number of PoS in our dataset. Achieving even a reasonable performance is difficult if only this small dataset is used for transformer-based supervised learning, which contains millions of learnable parameters. Thus we first train transformer \mathcal{T} on the SNLI dataset. This, in turn, produces a model \mathcal{T}' that is ready to understand the difficult representation of conflicting statements. We then train \mathcal{T}' on our 150 data points to obtain \mathcal{T}'_i for each fine-tune phase i , and finally test it on the large test dataset containing 1881 PoS for 4G and 2541 for 5G. Formally, the classification task can be defined as follows. For input x ,

$$\begin{aligned} h_{\mathcal{T}} &= \mathcal{T}(x) \\ h_{fc} &= W_{fc} * h_{\mathcal{T}} + b_{fc} \\ h_{drop} &= \text{Dropout}(h_{fc}, p_d) \\ \hat{y} &= \text{Softmax}(h_{drop}) \end{aligned}$$

where \hat{y} denotes the predicted label. W_{fc} denotes the weights of the fully connected layer with b_{fc} being the bias. p_d is the dropout rate. The overall model can be represented as:

$$\hat{y} = \text{Softmax}(\text{Dropout}(W_{fc} * \mathcal{T}(x) + b_{fc}, p_d))$$

4.6 Few Shot Learning Optimization

We emphasize that the straightforward end-to-end training is less interpretable and not robust at all. Since we have no ground truth validation set, in order to ensure the correctness of the learning phase, we must ensure that the learning process is robust and interpretable as much as possible. Therefore, we leverage two key techniques here over the end-to-end training: ① **Transform & Ensemble**. In this method, we use majority voting of multiple transformer models. We take BERT, RoBERTa, and XLNet for this approach [28, 49, 66]. Note that we complete pre-training of these models before using them here. We call this ensemble model EnCell. In §6, we discuss how this approach improves the confidence of the framework. ② **Inconsistency-aware penalty**. As discussed earlier, we have a very small expert-annotated dataset, worse yet, with even fewer datapoints under the inconsistency class. In this method, we modify the standard cross entropy loss to penalize for wrong prediction on “inconsistent” labels. We use weighted cross-entropy loss:

$$\mathcal{L}_{wce} = \sum_{i=1}^m \hat{w}_y y_i \log \hat{y}_i, \quad \text{where } w_y = \frac{N}{n_i}, \hat{w}_y = \frac{w_i}{\sum_j^C w_j}$$

Here, N is the total number of training PoS, n_i is the number of training PoS for class i , and C is the total number of classes.

This assigns a higher penalty for misclassifying labels that have fewer PoS (in our case, the inconsistency class) than other classes.

5 Implementation

In this section, we discuss the implementation details of CellularLint.

Computational Hardware. For pre-training, we use about 2500 computing units from Google Colab pro+, which is equipped with Nvidia A100 GPUs. Multi-phase fine-tuning is performed on a computing server equipped with a 64-core CPU, 3x RTX 3090Tis, and 128GB RAM. Pre-training the base version of each model takes about 4 days in total. The uninformed fine-tuning on SNLI takes about 8-12 hours for each model, and for the informed multi-phase, each model requires different times, ranging between 10-15 minutes for each phase.

Pre-training. We use BERT-base, RoBERTa-base, and XLNET-base as our candidate models. Each of these models is first pre-trained on the specification corpus. BERT-base and RoBERTa-base work on the Masked Language Model (MLM) objective and XLNET works on the Permutation Language Model (PLM) objective. The first two are pre-trained for 5 epochs, and since XLNET-base is much larger in size, we pre-train it for 1 epoch.

Fine-tuning. We use $k = 3$ for the k -phase training. In the 0th phase with the SNLI dataset, we keep the learning rate to be 3×10^{-5} and batch size 32. The models are trained for 8 epochs. For all the consecutive fine-tuning phases, we observe that learning rate 2×10^{-5} and batch size 32 work better. The Adam optimizer with $\epsilon = 10^{-8}$ has been used. To subsidize the low training data, we add $\frac{N}{10}$ synthetic PoS at each phase. We augment the dataset with Easy Data Augmentation (EDA) to gain these synthetic PoS [63]. For both pre-training and fine-tuning, Huggingface transformers standard pipeline based on PyTorch has been used [6].

To validate the inconsistencies in UE implementation, we prepared Software Defined Radio (SDR)-based testbeds. For details of the testbed preparation, we refer the reader to [13]. Table 5 lists the devices we tested.

6 Evaluation

In this section, we evaluate the performance of CellularLint from multiple perspectives—starting from ML training performance, identifying inconsistent descriptions of different levels, and ending with security implications. On a high level, we aim to answer the following research questions:

- *RQ1.* How does the choice of embedding affect the search space contraction?

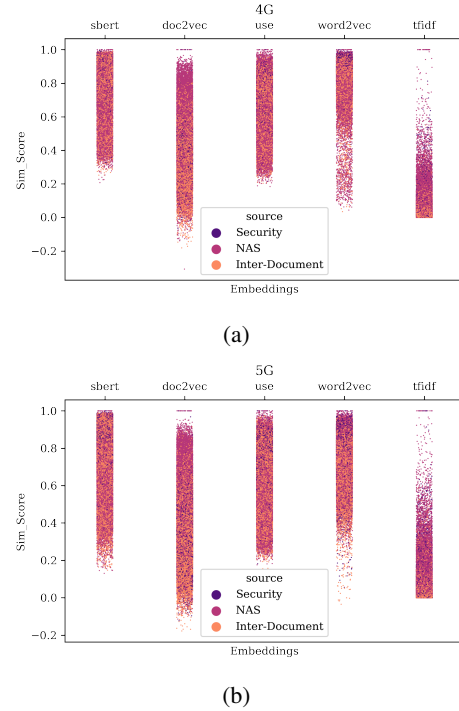


Figure 4: Embedding comparisons. Only 0.8% data were randomly sampled to generate the comparison for better visualization

- *RQ2.* What is the performance of EnCell for inconsistency classification and how do the optimization techniques affect the performance?
- *RQ3.* What are the issues found by CellularLint from specification documents and what are the security implications?
- *RQ4.* how does this methodology compare to other related approaches in terms of the final issues found?

6.1 RQ1. Effect of embedding choice

To benefit from the search space shrinkage through similarity measures, we identify the most effective embedding method for our dataset. Figure 4 presents the comparison of embeddings based on the distribution of ψ . For an embedding method ϵ , each point in the figure represents the ψ value of the PoS when ϵ is used. For better visualization, only 0.8% randomly sampled PoS are selected here. Recall that our method filters out irrelevant pairs and only considers the highly similar (syntactically) segments for further stages. Among the 5 different embeddings, namely SBERT, Doc2Vec, USE, Word2Vec, and TF-IDF, TF-IDF gives the most sparse representation on the high similarity span. It is evident from the comparison that SBERT, Word2Vec, and USE cannot be utilized to reduce the search space due to their very dense representation on the high similarity region. Doc2Vec shows

some sparsity above $\psi = 0.9$; however, the amount of PoS above that is very low. Following these results, we choose TF-IDF as our embedding technique for filtering.

6.1.1 PoS Filtration

For PoS filtering we set $\psi_{min} = 0.65$ for 4G and $\psi_{min} = 0.7$ for 5G, and $\psi_{max} = 0.99$ for both datasets. For 4G, we initially extracted 890 and 120 segments from NAS and security specifications, respectively. After sub-event driven quantization, we found 2599 unique segments for 4G. Considering all possible matches would give us $\sim 6.75M$ datapoints, whereas our fine-grained approach limits the number of datapoints to only 1881, which is $\sim 3591x$ lower. For 5G, 846 and 329 segments were extracted from NAS and security. After sub-event realization, we found 3529 unique sequences for 5G. Considering all possible matches would give us $\sim 12.45M$ datapoints whereas our fine-grained approach limits it to only 2541 datapoints, which is $\sim 4901x$ lower.

6.2 RQ2. EnCell performance

To evaluate how effectively the models and their ensemble EnCell adapt to the domain and classify inconsistencies, we evaluate them at each phase of the fine-tuning. Here, we show the precision, recall, F1-score, and accuracy of the models at each phase (Table 1 and Figure 5). Note that these are the $k = 0, \dots, 3$ phases of training we mentioned in submodule 1B of our method. Before the evaluation, all these models are pre-trained on the cellular network specification corpus and fine-tuned on the general-purpose SNLI dataset. Moreover, here all the models use weighted cross-entropy loss in all phases.

We observe that from phase 0 to phase 3, the F1 score improves about 29% for EnCell. RoBERTa performs better in the first two phases while EnCell does better in phase 3. This is expected as RoBERTa performs the best among the available pre-trained transformers on general NLI tasks. Once BERT and XLNet go through two phases of training, they start to perform in the vicinity of RoBERTa. This coherence of performance motivates the use of the ensemble approach in our task. Interestingly, although the accuracy of phase-3 RoBERTa (81.33%) and EnCell (82.67%) is similar, we observe approximately 2.9% improvement of EnCell over the F1 score of RoBERTa and 3.4% improvement over the best recall score provided by BERT. This emphasizes the need to use EnCell in an uncertain problem setting such as ours.

Effect of inconsistency-aware penalty. To understand the effectiveness of the data distribution-based weighted cross-entropy loss, we compare what happens when the general cross-entropy loss is used at phase 1 training and when the

inconsistency-aware penalty is used. When general cross-entropy loss is used, we see only 3 – 5% improvement on F1 score of our model. In contrast, the inconsistency-aware loss improves the F1 score of about 6 – 13% at phase 1. This clearly indicates the usefulness of the penalty in our setting.

6.3 RQ3. Issues found by CellularLint

CellularLint found a total of 186 inconsistent PoS from 4G and 5G specifications. We observe that among them, there are certain scenarios that, if taken in a broader context, do not contradict. Yet the model predicts them as inconsistencies since it cannot always understand the large event at a single inference. Note that we do not consider them as false positives; provided only the context in the PoS, we can see that they are semantically conflicting. However, the additional context provided in the original specification shows that the larger context is different. In some cases, the larger context is defined 3-4 pages earlier than the original PoS.

This is a fundamental challenge for NLP-based conflict detection for cellular network protocol specifications. Therefore, we manually filter out such apparent inconsistencies and group the PoS into different categories. After filtering, we found a total of 157 PoS that may affect design choices. Table 2 shows the categorical breakdown of inconsistent pairs found by CellularLint. To do this manual filtration, we spent 6 human hours of time and found around 16% of issues detected by CellularLint had to be filtered out due to broader context.

Table 2: Conflicting statements breakdown by category

	Category	Count
4G	Integrity & ciphering	26
	Security context handling	2
	Bearer context	1
	GUTI related	41
	State transition	12
	Counters	6
	Misc	3
5G	Security key generation & handling	3
	Integrity & ciphering	3
	Timers	7
	State transition	2
	Paging	3
	PLMN handling	3
	GUTI related	3
	QoS rules	5
	PDU session establishment	30
	Counters	2
	Misc	5
	Total	157

6.3.1 Investigation of the issues

To further investigate how the discovered issues may impact design choices, we examine them against open-source 4G and 5G implementations from srsRAN (v23.04.1), Open5GS (v 2.6.6), and OpenAirInterface (v2.0.0) [10–12]. We look into both the UE and Core-Network implementations of these open-source implementations, which are widely used to analyze cellular network security [24, 25, 38]. Furthermore, in

Table 1: Precision, Recall, F1 Score, and Accuracy for the models at each phase. *Italic* represents the best performance in each phase. **Bold** represents the best overall. Model_p denotes that the model has been *pretrained*.

Model	Phase 0				Phase 1			
	Precision	Recall	F1 Score	Accuracy	Precision	Recall	F1 Score	Accuracy
BERT _p	0.4606	0.4536	0.4547	0.5133	0.6125	0.6222	0.6114	0.68
RoBERTa _p	<i>0.5033</i>	0.5027	<i>0.5017</i>	<i>0.5667</i>	<i>0.6558</i>	<i>0.6606</i>	<i>0.6515</i>	<i>0.7133</i>
XLNet _p	0.4718	0.4748	0.4695	0.5467	0.5914	0.6041	0.591	0.6667
EnCell _p	0.5002	<i>0.5029</i>	0.498	<i>0.5667</i>	0.6167	0.6343	0.6171	0.6867

Model	Phase 2				Phase 3			
	Precision	Recall	F1 Score	Accuracy	Precision	Recall	F1 Score	Accuracy
BERT _p	0.6859	0.7026	0.689	0.7467	0.7473	0.7787	0.7562	0.7933
RoBERTa _p	0.6967	0.7062	0.6947	0.7533	0.7788	0.7695	0.7676	0.8133
XLNet _p	0.6534	0.6669	0.6552	0.72	0.6986	0.7097	0.6901	0.7533
EnCell _p	<i>0.7262</i>	<i>0.7404</i>	<i>0.7249</i>	<i>0.78</i>	0.7871	0.8148	0.7962	0.8267

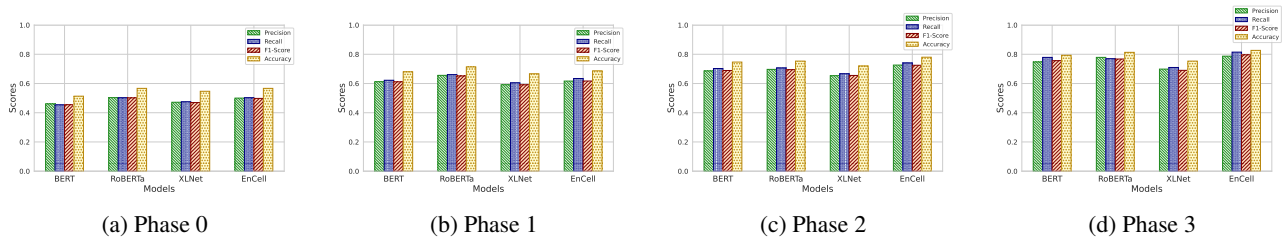


Figure 5: Performance metrics for different models used by CellularLint

case the inconsistencies can lead to security, privacy, availability or interoperability issues, we manually investigate and design exploits for the scenarios.

Table 3: Conflicting behaviors discovered from 4G Implementations

▲= 1st type, ▼= 2nd type, ✕= Did not implement, Red : Unsafe choice, *: partially implemented/related implementation found

Finding	Description	srsRAN	OAI
1	NAS message ciphering	▲	▲
2	Condition over Integrity Check	▼	▼
3	Integrity check failure	▼	▲
5	GUTI Deletion	▲*	✕
7.1	EMM deregistered sub-state transition	▼*	✕
7.2	EMM registered sub-state transition	▲	✕
9 [13]	Timer expiry as precondition	▲*	▲
10 [13]	TAU/Service attempt counter usage	▼	✕

Table 3 and 4 report the result of our observation. Note that, even in open-source implementations a lot of functionalities are not implemented and scenarios related to protocol handovers could not be tested. To further validate the impact of the identified inconsistent descriptions, we perform end-to-end attacks in 17 COTS UE devices (Table 5). Note that here as well we could not confirm issues related to network states as it is not possible to confirm the specific substate of the commercial devices. Furthermore, we could not execute any attacks on the commercial carrier side as it is prohibited by law. Nonetheless, we have reported all our findings to 3GPP and the vendors. We want to emphasize the ultimate aim of

Table 4: Conflicting behaviors discovered from 5G Implementations

▲: 1st type, ▼: 2nd type, ✕: Did not implement, Red : Unsafe choice, A: Additional

Finding	Description	Open5GS (Core)	SrsRAN (UE)	OAI
1	NAS message ciphering	▲	▲	▲
2	Condition over integrity check	▼	▼	▼
4	NCC reuseage	▼	▼	▼
5.1	4G-GUTI deletion	✕	▲*	✕
5.2	5G-GUTI deletion	✕	▲*	✕
8 [13]	Reset attempt counter	✕	▼	✕
A [13]	SUPI in auth_response	▼	✕	✕

CellularLint is to find inconsistent statements in the specifications, and the implementation testing (both open-source and commercial UEs) is done to further validate the impact conflicting statements can have on implementation design. Though we are inherently bound by the issues we can test, we take a *best-effort* approach in validating the impacts. In the following sections, we go into our threat model and the details of the findings.

Threat model. We consider the communication channel between the core network, base station, and UE to be under the influence of a Dolev-Yao [29] attacker. In this threat model, an adversary can generate/parse messages from scratch or modify any intercepted message while not having any cryptographic capabilities. Concretely, we consider three scenarios commonly assumed in 4G and 5G: Man-in-the-Middle

(MitM), Fake Base Station (FBS), and signal injection attacks. The threat model is consistent with the existing approaches that consider real-world threats [38, 54, 65].

6.3.2 Security Enforcement

Finding 1. NAS message ciphering. After the completion of security mode procedures, the specifications suggest that, except for a few messages, all messages should be integrity-protected and ciphered. However, on two occasions (shown in Figure 6, it is emphasized that all messages after the security procedure should be integrity-protected and ciphered. Failure to enforce integrity and ciphering is a serious design flaw and can leave implementations subject to vulnerabilities and interoperability issues.

T_1 : From this time onward the UE shall cipher and integrity protect all NAS signalling messages with the selected NAS ciphering and NAS integrity algorithms.
 T_2 : From this time onward, all NAS messages exchanged between the UE and the MME are sent integrity protected and **except for the messages** specified in clause 4.4.5, all NAS messages exchanged between the UE and the MME are sent ciphered

Figure 6: Ciphering exceptions. T_1 is from section 5.4.3.3 of TS 24.301 and T_2 is from section 4.4.2.5 of TS 24.301

Inconsistency to exploit. In the case of T_1 , the specification states after the security context has been established, the UE shall cipher and integrity protect all NAS signaling messages with selected NAS ciphering and integrity algorithms. However, T_1 does not mention the scenarios where the UE accepts plain-text messages. On the contrary, T_2 is more lucid and explicitly mentions such scenarios. Due to this inconsistency of not explicitly mentioning exception cases, an implementation might accept unexpected plain-text messages even after the security context is established. Therefore, we design an attack based on this scenario.

Attack. For the attack, the adversary connects with a victim UE through a fake base station to send plain-text `identity_request` or `authentication_request` [54]. Similarly, the attacker can overshadow downlink packets to create plain-text messages. The affected UE accepts, processes and responds to the messages.

Investigation. In open-source implementations, we found all implementations accepting plain-text authentication and identity requests even after the security context has been established. On the other hand, on commercial UEs, we found 4 UEs accepting and responding to such plain-text messages (Table 6). Interestingly, for a very recent UE (Google Pixel 7a, 2023), plain-text `authentication_request` is accepted, whereas plain-text `identity_request` is not accepted. This is another interesting scenario where vendors try to implement different behaviors for different exception case messages.

Impact. The impact of accepting plain-text or integrity-failed NAS messages can be catastrophic and can be exploited by attackers to fingerprint users, traceability, and denial of service

attacks. Note that DoLTesT [54] also reports several UEs accepting unprotected messages after the security context has been established, albeit from the implementation testing perspective. We, on the other hand, show that this issue in different implementations can be traced to the inconsistent behavior defined in the standards.

Finding 2. Condition over integrity check. On many occasions, the UE sets the counter for "SIM/USIM considered for GPRS/non-GPRS/5GS services" to implement a specific maximum value (Figure 7). While this flexibility is acceptable and standard practice, the precondition to check the integrity of the received message is often neglected.

T_1 : The UE shall consider the USIM as invalid for EPS services and non-EPS services until switching off or the UICC containing the USIM is removed or the timer T3245 expires as described in clause 5.3.7a. Additionally, the UE shall delete the list of equivalent PLMNs and enter state EMM-DEREGISTERED.NO-IMSI. **If the message has been successfully integrity checked** by the NAS and the UE maintains a counter for "SIM/USIM considered invalid for GPRS services", then the UE shall set this counter to UE implementation-specific maximum value.
 T_2 : The UE shall consider the USIM as invalid for EPS services until switching off or the UICC containing the USIM is removed or the timer T3245 expires as described in clause 5.3.7a. The UE shall delete the list of equivalent PLMNs and shall enter the state EMM-DEREGISTERED.NO-IMSI. If the UE maintains a counter for "SIM/USIM considered invalid for GPRS services", then the UE shall set this counter to UE implementation-specific maximum value.

Figure 7: Condition over integrity check. T_1 is from section 5.5.1.2.5 and T_2 is from section 5.5.2.3.2 of TS 24.301.

Inconsistency to exploit. In the case of T_1 , the protocol specifies successful integrity checking by the NAS. However, in T_2 , this integrity checking is skipped. These PoS are related to `attach_reject` and `detach_request`. However, we have also found instances of network-initiated `detach_request`, where there are inconsistencies. As the inconsistency is related to the integrity checking of messages, the attack steps and impact are the same as finding 1.

Investigation. In the case of open-source implementation, we see that all the open-source implementations in both 4G and 5G are vulnerable. However, in the case of commercial UEs, there were no instances of vulnerable behavior.

Finding 3. Integrity check failure. We found two conflicting PoS where different statements are found for failed integrity checks of the control plane messages (Figure 8). Following this PoS to 4G security specification, ultimately, we found three (one segment is common) instances of different statements. Two of them clearly suggest that messages that have faulty MAC should be discarded, whereas the third one directs to a slightly flexible strategy suggesting the processing of certain messages even if they fail integrity checks.

Inconsistency to exploit. In T_1 , whenever there are some exceptional NAS messages that can trigger further MME actions, even with failed integrity, the specification mentions that. However, in T_2 for RRC, it does not specify such exceptional cases, though such exceptional cases exist, for instance, when `RRC_connection_resume` fails an integrity check, rather than just discarding, there are further steps. Through further

T_1 : In case of failed integrity check (i.e. faulty or missing MAC-I) is detected after the start of integrity protection, the concerned message shall be discarded. This can happen on the UE side or on the eNB side
 T_2 : In case of failed integrity check (i.e. faulty or missing NAS-MAC) is detected after the start of NAS integrity protection the concerned message shall be discarded **except for some NAS messages** specified in TS 24.301 [9]. For those exceptions the MME shall take the actions ... NAS message with faulty or missing NAS-MAC

Figure 8: Allowing exceptions for integrity failure. T_1 is from section 7.3.2 and T_2 is from section 8.1.1 of TS 33.401.

manual analysis, we find these cases are not mentioned in the TS 33.501 (Security architecture and procedures for 5G Systems) specification but in TS 38.331 (RRC specification), which were not included in the scope of CellularLint. In a practical scenario, an implementor would use the RRC specification first, which clearly states exceptional messages with failed integrity. But later on, when the security specifications would be taken into account, the implementor would find it confusing with a more strict description. There is a possibility that such inconsistency may not directly result in vulnerabilities. Still, it may cause differing implementations as both specifications cannot be logically taken into consideration at the same time. Hence, some implementations might not properly follow the integrity failing scenarios.

Investigation. In open-source implementations, we found 1 implementation accepting integrity-failed messages. On the contrary, in commercial UEs, none of the UEs accept control-plane messages with failed integrity. However, in our investigation, we found another interesting behavior: 16 UEs dropped the connection after receiving an RRC packet with failed integrity, whereas 1 UE did not (Table 6). These inconsistent UE behaviors can be traced back to T_2 , which does not properly specify the exception cases and what to do in such scenarios.

Finding 4. NCC reuse. We found a conflicting PoS in the 5G Security specification where two different conditions are expressed for the validity of the Next hop Chaining Counter (NCC) (shown in Figure 9). NCC is used for cryptographic derivation of the AS security algorithms and, hence, is a very important identifier. One segment dictates that the NCC value has to be *fresh* and previously *unused* to be accepted. On the contrary, the other segment claims that the only condition for acceptance is that the NCC value has to be *different*.

T_1 : If the sent NCC value is **fresh** and belongs to an **unused pair** of NCC, NH, the gNB shall save the pair of {NCC, NH} in the current UE AS security context and shall delete the current AS key K_{gNB} .
 T_2 : The UE shall take the received NCC value and save it as stored NCC If the stored NCC value is **different** from the NCC value associated with the current K_{gNB} , the UE shall delete the current AS key K_{gNB}

Figure 9: Underspecified action for NCC. Both T_1 and T_2 are from section 6.8.2.1.2 of TS 33.501

Implication. NCC and NH are critical parameters that are used to establish AS security and derive K_{gNB} . These parameters are used during the RRC_Reestablish procedure to re-establish the RRC connection. This procedure is particularly important during handover. The expectation is that these

Table 5: List of devices tested

Device	Version	Release Year	Baseband
Google Pixel 7a	13	2023	Google Tensor G2
Samsung Galaxy S20 FE	10	2020	Qualcomm SM8250 Snapdragon 865 5G
HTC One E9	7.0	2015	Mediatek MT6795M Helio X10
Huawei Y5	9.0	2019	Mediatek MT6761 Helio A22
Xiaomi Mi 11 Lite 5G	11	2021	Qualcomm SM7350-AB Snapdragon 780G
Motorola Edge 30 Pro	12	2022	Qualcomm SM8450 Snapdragon 8 Gen
OnePlus 9 Pro	11	2021	Qualcomm SM8350 Snapdragon 888 5G
Honor 8X	8.1	2018	Kirin 710
Apple iPhone 12 Pro	iOS 17.3	2020	Apple A14 Bionic
Google Pixel 3a	9	2019	Qualcomm SDM670 Snapdragon 670
Samsung Galaxy A04	12	2022	Mediatek MT6765 Helio P35
LG Velvet 5G	10	2020	Qualcomm SM7250 Snapdragon 765G 5G
OnePlus 8T	11	2020	Qualcomm SM8250 Snapdragon 865 5G
BLU C5L Max	11	2021	Unisoc SC9832
TCL 30	12	2022	Mediatek MT6765V/CB Helio G37
Samsung Galaxy S8+	9	2018	Qualcomm MSM8998 Snapdragon 835
Motorola Moto G Play	12	2022	Mediatek MT6765 Helio G37

session key creation parameters (NCC and NH) would be fresh and unused to create diverse keys (precisely described in T_1). This is an important assumption ensuring forward and future secrecy guarantees of the keys. Forward and future secrecy ensures that the protocol defends the past and future sessions even if the current session is compromised [16]. These parameters are essentially used as nonces to ensure the diversification of the keys. However, if the fresh and unused NCC/NH value usage is not mandated, then the forward and future secrecy guarantees can be broken. Furthermore, as the RRC_Reestablish message (containing these parameters) is unencrypted, the attacker can easily detect the sessions where the same NCC/NH values are used for key derivation.

Investigation. We found that none of the open-source implementations properly check these parameters and just accept if they differ from the previously accepted ones.

6.3.3 Privacy

Finding 5. GUTI deletion. On many occasions, when a reject message is received from the network, the UE updates its EPS/5GS status, clears the context, and subsequently moves to a deregistered state. We observe that in many cases, a rejection cause \mathcal{T}_C in a reject message \mathcal{R}_M would suggest the UE to clear the context and move to deregistered state while the same cause \mathcal{T}_C would keep the UE in registered state without clearing the security context (shown in Figure 10). For example, 5GM cause #13 received through `registration_reject` suggests the UE to delete

Table 6: COTS UE behavior based on the inconsistencies found by CellularLint. Red marks the Exploitable cases. ☆ denotes cases where the result could not be verified.

Device	F1		F2	F3	F5	F6			
	Plain-text auth request accepted	Plain-text identity request accepted	Plain-text detach request accepted	Integrity-failed message accepted	Causes connection drop	attach reject clears context	service reject clears context	tau reject clears context	tau & detach collision
Google Pixel 7a	✓	✗	✗	✗	✓	✓	✗	✗	TAU progressed
Samsung Galaxy S20 FE	✗	✗	✗	✗	✓	✓	✗	✗	TAU progressed
HTC One E9 Plus	✓	✓	✗	✗	✓	✓	✗	✗	TAU progressed
Huawei Y5	✓	✓	✗	✗	✓	✓	✗	✗	TAU progressed
Xiaomi 11 Lite	✗	✗	✗	✗	✓	✓	✗	✗	TAU progressed
Motorola Edge 30 Pro	✗	✗	✗	✗	✓	✓	✗	✗	TAU progressed
OnePlus 9 Pro	✗	✗	✗	✗	✓	✓	✗	✗	TAU progressed
Huawei Honor 8X	✗	✗	✗	✗	✗	✓	✗	✗	TAU progressed
Apple iPhone 12 Pro	✗	✗	✗	✗	✓	✓	✓	☆	☆
Google Pixel 3a	✗	✗	✗	✗	✓	✓	✗	✗	TAU progressed
Samsung Galaxy A04	✗	✗	✗	✗	✓	✓	✗	✗	TAU progressed
LG Velvet 5G	✓	✓	✗	✗	✓	✓	✗	✗	TAU progressed
OnePlus 8T	✗	✗	✗	✗	✓	✓	✗	✗	TAU progressed
BLU C5L Max	✗	✗	✗	✗	✓	✓	✗	✗	TAU progressed
TCL 30	✗	✗	✗	✗	✓	✓	✗	✗	TAU progressed
Samsung Galaxy S8+	✗	✗	✗	✗	✓	✓	✗	✗	TAU progressed
Motorola Moto G Play	✗	✗	✗	✗	✓	✓	✗	✗	TAU progressed

GUTI, TAI, ngKSI while the same cause received through `tau_reject` or `service_reject` message would keep the UE in a registered state without deleting the context. A cross-examination may suggest that since the messages are different, the same cause may trigger different behavior. However, other cause values such as #12 (tracking area not allowed) trigger similar behavior and state transition for different reject messages. Thus, it concretely verifies the conflicting suggestions about security context.

T_1 : #13 (Roaming not allowed in this tracking area) The UE shall set the 5GS update status to 5U3 ROAMING NOT ALLOWED (and shall store it according to subclause 5.1.3.2.2) and shall delete 5G-GUTI, last visited registered TAI, TAI list and ngKSI
 T_2 : #13 (Roaming not allowed in this tracking area) The UE shall set the 5GS update status to 5U3 ROAMING NOT ALLOWED (and shall store it according to subclause 5.1.3.2.2) and shall delete the list of equivalent PLMNs (if available).

Figure 10: GUTI, TAI, eKSI deletion. T_1 is from section 5.5.1.2.5 and T_2 is from section 5.5.1.3.5 of TS 24.501.

Inconsistency to exploit. GUTI (Globally Unique Temporary ID) is a kind of temporary ID used to identify UEs. Each UE has a couple of different kinds of unique IDs, like IMSI, IMEI, etc. These sorts of temporary identifiers, like GUTI, are used to prevent attackers from tracking users. However, if these identifiers are not changed or reset, this can cause

several privacy issues. An adversary can utilize old GUTI to track a UE through a linkability attack, violating privacy [33]. **Investigation.** In commercial UEs, we found a total of 16 devices not properly deleting GUTI for these reject messages (Table 6). One recent UE (Apple iPhone 12 Pro), however, deletes the security context with GUTI after receiving `service_reject`.

6.3.4 Denial of Service

Finding 6. TAU and detach collision. We found a PoS of conflicting directions when TAU and detach procedure collide (shown in Figure 11). The first one states that the `tracking_area_update` procedure shall be aborted and the detach procedure shall be progressed. On the other hand, the second segment states that the detach procedure shall be aborted and re-initiated while the `tracking_area_update` procedure is fully performed.

Inconsistency to exploit. An attacker can achieve downgrade/denial-of-service by injecting `detach_request` through fake-base-stations or signal-injection attacks [35]. In this case, an implementation that aims to implement both scenarios can exacerbate the situation by creating a deadlock state. As the statements are mutually exclusive either way,

T_1 : Tracking area updating and detach procedure collision EPS detach containing detach type "re-attach required" or "re-attach not required": If the UE receives a DETACH REQUEST message before the tracking area updating procedure has been completed, the **tracking area updating procedure shall be aborted** and the **detach procedure shall be progressed**.
 T_2 : If a cell change into a new tracking area that is not in the stored TAI list occurs before the UE initiated detach procedure is completed, the UE proceeds as follows: 1) If the detach procedure was initiated for reasons other than removal of the USIM or the UE is to be switched off, the **detach procedure shall be aborted** and re-initiated **after success- fully performing a tracking area updating procedure**.

Figure 11: TAU and Detach Procedure precedence conflicts. T_1 is from section 5.5.3.2.6 and T_2 is from section 5.5.2.2.4 of TS 24.301.

an implementation violates the specification. In our investigation of commercial implementation, we found all the UEs progress with the `tracking_area_update` procedure when such collision occurs (Table 6).

6.3.5 Interoperability

For this class of inconsistencies, we could not devise an attack scenario. However, these inconsistencies range from specifying different sub-state transitions for the same condition to missing timer expiration directives. The potential impact of these inconsistencies can cause interoperability issues between different implementations, i.e., one implementation following one directive and the other one following a different directive. For brevity, one of them is discussed here; we discuss the rest in [13].

Finding 7. Sub-state transition confusion. When `attach_reject` message is received with the `emm` cause #14, the 4G specification has differing state transition descriptions (shown in Figure 1). This certainly can cause confusion in implementation design.

Investigation. While looking further into open-source implementations, we found a very interesting scenario regarding this. In srsRAN, we see that the developers tried to implement both of them. We show the code snippet of srsRAN in Figure 12. Here, we can see that both of the conditional statements contain the same cause (lines 1 and 6) but have different sub-state transitions (lines 3 and 7). Of course, UE will go to the `EMM_DEREGISTERED.PLMN_SEARCH` sub-state due to the obscurity of the first "if". Such conflicts can potentially cause interoperability issues, leaving a possibility for further synchronization problems.

Finding 8 is about registration counter resetting, *finding 9* is about utilizing/not-utilizing timer expiry as a precondition, *finding 10* is about utilization of service attempt counter, *finding 11* is about PDCP counter resetting. We refer the interested reader to [13] for details.

6.4 RQ4. Comparison of other methodologies

```

1 if (...|| attach_rej.emm_cause ==
  LIBLTE_MME_EMM_CAUSE_EPS_SERVICES_NOT_
  ALLOWED_IN_THIS_PLMN ||...){
2   attach_attempt_counter = 0;
3   enter_emm_deregistered(emm_state_t::
  deregistered_substate_t::attempting_to_attach);
4 }
5 // TODO: handle other relevant reject causes
6 if (...|| attach_rej.emm_cause ==
  LIBLTE_MME_EMM_CAUSE_EPS_SERVICES_NOT_
  ALLOWED_IN_THIS_PLMN ||...){
7   enter_emm_deregistered(emm_state_t::
  deregistered_substate_t::plmn_search);
  }

```

Figure 12: srsRAN implementation of `emm` cause #14

Table 7: Comparison with existing approaches.

■ : Did not discover ■ : Closely related issue discovered
 ■ : Only context is similar ■ : Discovered

Findings	DIKEUE [38]	DoLTEst [54]	ATOMIC [25]	Instructions Unclear [46]	CellularLint
1	■	■	■	■	■
2	■	■	■	■	■
3	■	■	■	■	■
4	■	■	■	■	■
5	■	■	■	■	■
6	■	■	■	■	■
7	■	■	■	■	■
8 [13]	■	■	■	■	■
9 [13]	■	■	■	■	■
10 [13]	■	■	■	■	■
11 [13]	■	■	■	■	■

Although to the best of our knowledge CellularLint is the first approach to detect inconsistencies from cellular specifications, we compare our methodology with the existing approaches based on the issues discovered. Table 7 shows comparison results based on the subset of findings presented in the paper. In terms of methodologies we compare with DIKEUE and DoLTEst that are implementation testing frameworks and, therefore, are not directly comparable to our methodology. The closest methods that deal with specification descriptions are [25] and [46]. We compare with respect to four characteristics: (1) if the issue was discovered, (2) if the context aligned to the issue has been discussed, (3) if a related root cause has been discussed, and (4) if the issue has not been discussed at all.

7 Related Works

We discuss the related works in three categories:

NLP efforts. Sequence classification tasks, especially supervised textual entailment, have largely been used recently in various domains and applications [18, 25, 34, 58, 59]. In contrast to those techniques, Andow et al. [14, 15] use Named Entity Recognition to leverage a rule-based approach for detecting privacy leaks in privacy policies of Android apps. The method, however, considers domain-specific assumptions for

ontology structuring and is not generalizable to cellular protocol documents.

NLP for cellular network. In recent days, NLP research in cellular networks has been gaining popularity due to the enormous information and the emergence of large language models. For instance, methodologies have been proposed to automatically analyze specification, RFC, and Change Request documents in various problem setting [23–25, 40, 42, 53]. Pacheco et al. [53] designed an approach that automatically builds state machines from RFC protocol documents using a data-driven zero-shot approach and synthesizes attacks based on that. Ishtiaq et al. [40] take a similar approach to synthesize FSM from 4G and 5G specifications. Chen et al. [25] discover Hazard Indicators (HI) from 4G specifications and verify implementations against them. Briefly, they input Risky Operation Description (ROD) and threat models in their framework and run a textual entailment model to find the HIs from which the test cases are generated. While they generate test cases from individual text descriptions, our method does a differential analysis on pairs of descriptions to generate test cases. A conformance testing approach has also been proposed based on NLP and causal relation extraction [23]. However, these methods do not primarily focus on inconsistencies in standards.

Vulnerability detection in cellular networks: Over the years a lot of work has focused on finding issues in various generations of cellular network design and implementation using formal verification [27, 37], dynamic analysis [45], differential testing [38], fuzzing, etc. Klischies et al. [46] consider underspecifications from protocol standards that are closely aligned with our goal. Baseband firmware analysis has also been effective in finding implementation issues [1, 43, 50]. However, these methods do not relate the root causes of implementation issues to the protocol standards.

8 Discussion

Manual efforts and limitations in CellularLint. Although CellularLint can discover conflicts under weakly supervised settings, there is still some manual labor involved. During the active learning phase, the annotators have to cross-examine the predictions from the model and refine them accordingly. This required approximately 16 human hours from domain experts. Furthermore, one of the critical limitations of CellularLint is that there are certain inconsistent PoS that, when taken into a broader specification context, are no longer inconsistent. This issue, with a larger context, requires manual efforts to parse through the results of CellularLint and filter out. This, in some sense, is the limitation of current NLP models that are unable to take into account this large context of cellular network specifications. To provide an example, in some cases, the context is defined 3–4 pages before the actual segment. It took about 6 human hours to detect the

false positive from the final results.

Inconsistency-to-attack scenario. The aim of CellularLint is to find inconsistencies in the 4G and 5G specifications. As mentioned in §6.3.1, the design from inconsistent PoS to concrete attacks is a manual effort. We look into the POSs to uncover whether these design choices can lead to an attack. We want to emphasize that not all inconsistencies can lead to exploitable attacks. Similarly, as this part of the findings is manually designed, there can be other inconsistencies that lead to exploitable attacks but are not discussed in detail in the paper. Therefore, we open-source the list of major inconsistencies [13].

Implementation vs. specification issues. In terms of security issues in the 4G and 5G, they can be characterized in either one of the two categories: issues stemming from the specification design or from the implementation design. In this paper, we focus on the first category. For instance, the issues in findings 1 and 3 can be categorized as implementation issues where the UE accepts plain-text or integrity-failed messages. However, we focus on the specification consistency. The aim of our work is to clear scenarios in the standards that can cause inconsistencies in the implementation. We believe our work would inspire further specification-based textual analysis and ultimately realize the goal of machine-readable specifications.

Scalability and relevance to further security research. CellularLint is designed in a fashion that can be almost readily used for other cellular specifications and, with some training effort, can be utilized for other communication (non-cellular) protocols. For other cellular specifications, one can use the pre-trained+SNLI-fine-tuned state of EnCell. Only the last fine-tuning step with a low amount of data would be required. For completely different communication protocols, one can utilize the methodology of CellularLint to pre-train and fine-tune the models. Pairing and filtration, PoS selection, and optimization techniques would still follow the same methods. Furthermore, the impact of CellularLint is not limited to cellular network security, it can be applied to enhance the security of other communication protocols as well.

Impact of tables, figures and codes. While preprocessing, we removed tabular data, figures, and code snippets. These artifacts can potentially add more information to decide which of the text pairs are more secure. For example, CellularLint captured an inconsistent description regarding EAP-TLS authentication procedure, which we had to verify as a false positive manually. However, figure B.2.1.1-1 under Annex B.2 in TS 33.501 describes a timing diagram of that authentication procedure more clearly. Considering this could potentially help CellularLint capture the false positive in initial stage of analysis. Moreover, developers often focus on the finite state machines presented through the figures. Thus, learning from these artifacts alongside texts could strengthen the learning process, and we leave it as future work.

9 Conclusion and future work

In the paper, we propose CellularLint to detect inconsistencies in cellular protocol specifications in a scalable fashion. CellularLint uncovers 157 inconsistencies with 82.67% accuracy in the NAS and security specifications. After verification of these inconsistencies on open-source implementations and commercial devices, we confirm that they indeed have a substantial impact on design decisions, potentially leading to concerns related to privacy, integrity, availability, and interoperability. **Future work.** In the future, we will work on improving CellularLint to include further context and improve the accuracy. Furthermore, we will aim to detect underspecifications with NLP techniques.

Acknowledgments

We thank the shepherd and the reviewers for their suggestions and insightful discussion. The work reported in this paper has been supported by NSF under grant 2112471.

References

- [1] Baseband attacks: Remote exploitation of memory corruptions in cellular protocol stacks. In *WOOT*, 2012.
- [2] *3GPP, A Global Initiative*, 2023. <https://www.3gpp.org/>.
- [3] *5G Security Architecture*, 2023. https://www.etsi.org/deliver/etsi_ts/133500_133599/133501/17.05.00_60/ts_133501v170500p.pdf.
- [4] *Ericsson*, 2023. <https://www.ericsson.com/en/reports-and-papers/mobility-report/dataforecasts/mobile-subscriptions-outlook>.
- [5] *Global Mobile Suppliers Association*, 2023. <https://gsacom.com/paper/5g-subscribers-march-2023-update/>.
- [6] *Huggingface Transformers*, 2023. <https://huggingface.co/docs/transformers/index>.
- [7] *LTE Security Architecture*, 2023. https://www.etsi.org/deliver/etsi_ts/133400_133499/133401/17.01.00_60/ts_133401v170100p.pdf.
- [8] *Non-Access-Stratum (NAS) protocol for 5G System (5GS)*, 2023. https://www.etsi.org/deliver/etsi_ts/124500_124599/124501/17.07.01_60/ts_124501v170701p.pdf.
- [9] *Non-Access-Stratum (NAS) protocol for Evolved Packet System (EPS)*, 2023. https://www.etsi.org/deliver/etsi_ts/124300_124399/124301/17.06.00_60/ts_124301v170600p.pdf.
- [10] *Open5GS, Open Source implementation for 5G Core and EPC*, 2023. <https://www.open5gs.org/>.
- [11] *OpenAirInterface | 5G software alliance for democratising wireless innovation*, 2023. <https://www.openairinterface.org>.
- [12] *SrsRAN Project*, 2023. <https://www.srslte.com/5g>.
- [13] *CellularLint: A Systematic Approach to Identify Inconsistent Behavior in Cellular Network Specifications*, 2024. <https://cellularlint.github.io/>.
- [14] Benjamin Andow, Samin Yaseer Mahmud, Wenyu Wang, Justin Whitaker, William Enck, Bradley Reaves, Kapil Singh, and Tao Xie. PolicyLint: Investigating internal privacy policy contradictions on google play. In *USENIX Security*, 2019.
- [15] Benjamin Andow, Samin Yaseer Mahmud, Justin Whitaker, William Enck, Bradley Reaves, Kapil Singh, and Serge Egelman. Actions speak louder than words: Entity-Sensitive privacy policy and data flow analysis with PoliCheck. In *USENIX Security*, 2020.
- [16] Daniele Antonioli. Bluffs: Bluetooth forward and future secrecy attacks and defenses. In *Proc. of CCS*, 2023.
- [17] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Proc. of EMNLP*, 2015.
- [18] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Proc. of EMNLP*, 2015.
- [19] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in NeurIPS*, 2020.
- [20] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *CoRR*, abs/2005.14165, 2020.
- [21] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. Universal sentence encoder for English. In *Proc. of EMNLP: System Demonstrations*, 2018.
- [22] Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. LEGAL-BERT: The muppets straight out of law school. In *Findings of EMNLP*, 2020.
- [23] Yi Chen, Di Tang, Yepeng Yao, Mingming Zha, XiaoFeng Wang, Xiaozhong Liu, Haixu Tang, and Baoxu Liu. Sherlock on specs: Building LTE conformance tests through automated reasoning. In *USENIX Security*, 2023.
- [24] Yi Chen, Di Tang, Yepeng Yao, Mingming Zha, XiaoFeng Wang, Xiaozhong Liu, Haixu Tang, and Dongfang Zhao. Seeing the forest for the trees: Understanding security hazards in the 3GPP ecosystem through intelligent analysis on change requests. In *USENIX Security*, 2022.
- [25] Yi Chen, Yepeng Yao, XiaoFeng Wang, Dandan Xu, Chang Yue, Xiaozhong Liu, Kai Chen, Haixu Tang, and Baoxu Liu. Bookworm game: Automatic discovery of lte vulnerabilities through documentation analysis. In *IEEE Security and Privacy (SP)*, 2021.
- [26] Merlin Chlosta, David Rupperecht, Thorsten Holz, and Christina Pöpper. Lte security disabled: Misconfiguration in commercial networks. In *Proc. of WiSec*, 2019.
- [27] Cas Cremers and Martin Dehnel-Wild. Component-based formal analysis of 5g-aka: Channel assumptions and session confusion. In *NDSS*, 2019.
- [28] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.
- [29] D. Dolev and A. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 1983.

- [30] Liat Ein Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. Active learning for bert: an empirical study. In *Proc. of EMNLP*, 2020.
- [31] Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. In *Proc. of ACL-IJCNLP (Volume 1: Long Papers)*, 2021.
- [32] Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare (HEALTH)*, 2021.
- [33] Byeongdo Hong, Sangwook Bae, and Yongdae Kim. Gut: reallocation demystified: Cellular location tracking with changing temporary identifier. In *NDSS Symposium*, 2018.
- [34] Hai Hu, Kyle Richardson, Liang Xu, Lu Li, Sandra Kübler, and Lawrence Moss. OCNLI: Original Chinese Natural Language Inference. In *Findings of EMNLP*, 2020.
- [35] Syed Rafiul Hussain, Omar Chowdhury, Shagufta Mehnaz, and Elisa Bertino. Lteinspector: A systematic approach for adversarial testing of 4g lte. In *NDSS Symposium*, 2018.
- [36] Syed Rafiul Hussain, Mitziu Echeverria, Omar Chowdhury, Ninghui Li, and Elisa Bertino. Privacy attacks to the 4g and 5g cellular paging protocols using side channel information. *Proc. of NDSS Symposium*, 2019.
- [37] Syed Rafiul Hussain, Mitziu Echeverria, Imtiaz Karim, Omar Chowdhury, and Elisa Bertino. 5greasoner: A property-directed security and privacy analysis framework for 5g cellular network protocol. In *Proc. of ACM CCS*, 2019.
- [38] Syed Rafiul Hussain, Imtiaz Karim, Abdullah Al Ishtiaq, Omar Chowdhury, and Elisa Bertino. Noncompliance as deviant behavior: An automated black-box noncompliance checker for 4g lte cellular devices. In *Proc. of CCS*, 2021.
- [39] Jinbae Im and Sungzoon Cho. Distance-based self-attention network for natural language inference. *arXiv preprint arXiv:1712.02047*, 2017.
- [40] Abdullah Al Ishtiaq, Sarkar Snigdha Sarathi Das, Syed Md Mukit Rashid, Ali Ranjbar, Kai Tu, Tianwei Wu, Zhezheng Song, Weixuan Wang, Mujtahid Akon, Rui Zhang, and Syed Rafiul Hussain. Hermes: Unlocking security analysis of cellular network protocols by synthesizing finite state machines from natural language specifications, 2023.
- [41] Bedran Karakoc, Nils Fürste, David Rupprecht, and Katharina Kohls. Never let me down again: Bidding-down attacks and mitigations in 5g and 4g. In *Proc. of WiSec*, 2023.
- [42] Imtiaz Karim, Kazi Samin Mubasshir, Mirza Masfiqur Rahman, and Elisa Bertino. Spec5g: A dataset for 5g cellular network protocol analysis. In *Proc. of IJCNLP-AACL*, 2023.
- [43] Eunsoo Kim, Dongkwan Kim, CheolJun Park, Insu Yun, and Yongdae Kim. Basespec: Comparative analysis of baseband software and cellular specifications for l3 protocols. *Proc. of NDSS Symposium*, 2021.
- [44] Hongil Kim, Dongkwan Kim, Minhee Kwon, Hyungseok Han, Yeongjin Jang, Dongsu Han, Taesoo Kim, and Yongdae Kim. Breaking and fixing volte: Exploiting hidden data channels and misimplementations. In *Proc. of CCS*, 2015.
- [45] Hongil Kim, Jiho Lee, Eunkyu Lee, and Yongdae Kim. Touching the untouchables: Dynamic security analysis of the lte control plane. In *IEEE Security and Privacy (SP)*, 2019.
- [46] Daniel Klischies, Moritz Schloegel, Tobias Scharnowski, Mikhail Bogodukhov, David Rupprecht, and Veelasha Moonsamy. Instructions unclear: Undefined behaviour in cellular network specifications. In *USENIX Security*, 2023.
- [47] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In Eric P. Xing and Tony Jebara, editors, *Proc. of ICML*, 2014.
- [48] Chi-Yu Li, Guan-Hua Tu, Chunyi Peng, Zengwen Yuan, Yuanjie Li, Songwu Lu, and Xinbing Wang. Insecurity of voice solution volte in lte mobile networks. In *Proc. of CCS*, 2015.
- [49] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.
- [50] Dominik Maier, Lukas Seidel, and Shinjo Park. Basesafe: Baseband sanitized fuzzing through emulation. In *Proc. of WiSec*, 2020.
- [51] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.
- [52] Robert Munro Monarch. *Human-in-the-Loop Machine Learning: Active learning and annotation for human-centered AI*. Simon and Schuster, 2021.
- [53] Maria Leonor Pacheco, Max von Hippel, Ben Weintraub, Dan Goldwasser, and Cristina Nita-Rotaru. Automated attack synthesis by extracting finite state machines from protocol specification documents. In *IEEE Security and Privacy (SP)*, 2022.
- [54] CheolJun Park, Sangwook Bae, BeomSeok Oh, Jiho Lee, Eunkyu Lee, Insu Yun, and Yongdae Kim. overshadow: In-depth downlink negative testing framework for LTE devices. In *USENIX Security*, 2022.
- [55] Laila Rasmay, Yang Xiang, Ziqian Xie, Cui Tao, and Degui Zhi. Medbert: pretrained contextualized embeddings on large-scale structured electronic health records for disease prediction. *NPJ digital medicine*, 2021.
- [56] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proc. of EMNLP-IJCNLP*, 2019.
- [57] David Rupprecht, Katharina Kohls, Thorsten Holz, and Christina Pöpper. Imp4gt: Impersonation attacks in 4g networks. In *ISOC NDSS*, 2020.
- [58] Mobashir Sadat and Cornelia Caragea. Scinli: A corpus for natural language inference on scientific text, 2022.
- [59] Rafet Sifa, Maren Pielka, Rajkumar Ramamurthy, Anna Ladi, Lars Hillebrand, and Christian Bauckhage. Towards contradiction detection in german: a translation-driven approach. In *IEEE SSCI*, 2019.
- [60] Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 1972.
- [61] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.
- [62] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in NeurIPS*, 2017.
- [63] Jason Wei and Kai Zou. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proc. of EMNLP-IJCNLP*, 2019.
- [64] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Proc. of EMNLP: System Demonstrations*, 2020.
- [65] Hojoon Yang, Sangwook Bae, Mincheol Son, Hongil Kim, Song Min Kim, and Yongdae Kim. Hiding in plain signal: Physical signal overshadowing attack on LTE. In *USENIX Security*, 2019.
- [66] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in NeurIPS*, 2019.