Seeing the Forest for the Trees: Understanding Security Hazards in the 3GPP Ecosystem through Intelligent Analysis on Change Requests

Yi Chen[1], Di Tang[1], Yepeng Yao[2,4]*, Mingming Zha[1], XiaoFeng Wang[1]*
Xiaozhong Liu[3], Haixu Tang[1], Dongfang Zhao[1]
[1]*Indiana University Bloomington*
[2]*{CAS-KLONAT†, BKLONSPT‡}, Institute of Information Engineering, CAS*
[3]*Worcester Polytechnic Institute,* [4]*School of Cyber Security, University of Chinese Academy of Sciences*
{chen481, tangd, mzha, xw7, hatang, zhaodo}@iu.edu, yaoyepeng@iie.ac.cn, xliu14@wpi.edu

Abstract

With the recent report of erroneous content in 3GPP specifications leading to real-world vulnerabilities, attention has been drawn to not only the specifications but also the way they are maintained and adopted by manufacturers and carriers. In this paper, we report the first study on this 3GPP ecosystem, for the purpose of understanding its security hazards. Our research leverages 414,488 *Change Requests* (CRs) that document the problems discovered from specifications and proposed changes, which provides valuable information about the security assurance of the 3GPP ecosystem.

Analyzing these CRs is impeded by the challenge in finding security-relevant CRs (SR-CRs), whose security connections cannot be easily established by even human experts. To identify them, we developed a novel NLP/ML pipeline that utilizes a small set of positively labeled CRs to recover 1,270 high-confidence SR-CRs. Our measurement on them reveals serious consequences of specification errors and their causes, including design errors and presentation issues, particularly the pervasiveness of inconsistent descriptions (*misalignment*) in security-relevant content. Also important is the discovery of a security weakness inherent to the 3GPP ecosystem, which publishes an SR-CR long before the specification has been fixed and related systems have been patched. This opens an "attack window", which can be as long as 11 years! Interestingly, we found that some recently reported vulnerabilities are actually related to the CRs published years ago. Further, we identified a set of vulnerabilities affecting major carriers and mobile phones that have not been addressed even today. With the trend of SR-CRs not showing any sign of abating, we propose measures to improve the security assurance of the ecosystem, including responsible handling of SR-CRs.

1 Introduction

The rapid advancement of telecommunication technologies and perspectives of their applications to security-critical domains like autonomous driving, emergency services, energy infrastructure, have brought to spotlight their security assurance. At the center is the ecosystem that supports development, maintenance and adoption of telecommunication standards, as organized by the 3rd Generation Partnership Project (3GPP) [1], a consortium involving all major telecommunication standards organizations around the world. In the past two decades, 3GPP has been responsible for standardizing 2G/3G/4G/5G protocols. Recent years, however, have witnessed concerns being raised about the security quality of its specifications: studies show that security flaws can be found from the design described in protocol documents [31,32] or predicted from their statements [16]. These reported hazards can well be just a tip of the iceberg, given 3GPP's complicated, error-prone procedure for specification development (involving hundreds of parties across 46 countries), and its indiscreet release of vulnerability information. An in-depth analysis of the 3GPP ecosystem therefore becomes critical to understanding the security guarantees of today's telecommunication technologies, but has never been done before.

Challenges in the ecosystem analysis. In the way of such a security analysis is the complexity of 3GPP specifications, which are characterized by convoluted descriptions in thousands of documents, on millions of pages. Understanding the content of these documents is painstaking, not to mention analysis of their security quality and measurement of security weaknesses they may carry. In our research, however, we found a unique resource that can be leveraged: a large number of *Change Requests* (CRs) that specify the details of the changes proposed by 3GPP members. Among them, these *Security-Relevant* CRs (SR-CRs) report the descriptions that could lead to security risks, which essentially are samples of the security weaknesses from specifications. So the nature of these problems and the ways they are handled can help us assess the security assurance offered by the 3GPP ecosystem.

However, finding SR-CRs is highly nontrivial. Already there are over 400K CRs, which continue to accumulate at a fast pace. Only a very small portion of them are security-related. These CRs are not explicitly labeled, determining

*Corresponding Authors

†Key Laboratory of Network Assessment Technology, CAS.

‡Beijing Key Laboratory of Network Security and Protection Technology

their security connections requires in-depth domain knowledge. As an example, *S3-171355* reports the absence of details about computation of $HASH_{MME}$ and $HASH_{UE}$. However, without knowing the purpose these hash values serve, one would have no idea about the CR's relation to the defense against a bidding down attack.

The challenge in understanding CRs and their large volume make any manual effort hard to succeed. Even an attempt to automate the analysis, finding SR-CRs through machine learning (ML), faces the difficulty in labeling training data, a painstaking process that can only be handled by 3GPP experts. As a result, any ML-based solution can only count on a small set of ground-truth data (301 SR-CRs in our research).

Intelligent CR analysis. To address this challenge, we developed a new Natural-Language Processing (NLP) and ML pipeline, called *CREEK* (CR Seeker), based upon the recent progress in these areas. Our approach utilizes a small set of manually identified *positive instances* (which are easier to label than negative instances) to train a binary classier for finding SR-CRs. For this purpose, we leveraged the idea of transduction to learn a related but easier task: whether a given paragraph comes from a security specification (e.g., Technical Specification (TS) 33.401[1]), which is explicitly labeled by 3GPP. This learning process results in a transformer generating embeddings for input paragraphs. On the embeddings of the labeled positive instances and a subset of unlabeled CRs, we run Positive-Unlabeled (PU) learning to train a classifier. The classifier is further refined using self-training on the rest of the unlabeled CRs. Here our transduction learning uses the information learnt from the related (easier) task to enrich the knowledge necessary for finding SR-CRs, PU-learning builds the classifier just on positively labeled instances and self-training propagates labels to unlabeled data. Not to mention our innovation on the loss function for the adversarial training framework for PU-learning, which addresses the potential bias. Our study shows that the CREEK pipeline is effective at capturing SR-CRs: over 400K CRs, it reported 1,270 SR-CRs with a precision of 91.6%.

Measurement and findings. Our NLP/ML pipeline enables us to focus on SR-CRs to study security hazards in the 3GPP ecosystem. In our research, we analyzed the 1,270 SR-CRs detected with high confidence, which reveals serious, sometime surprising risks. More specifically, we found that the security issues discovered from 3GPP documents have significant and diverse consequences, including denial of service, information leak, overcharging, etc. Over 70% of them are design errors, often present in security-related operations. Remains are problematic presentations including "unclear description" that misses security-relevant details, and inconsistent statements (called *misalignment*). Of particular interest is the pervasiveness of the misalignment that however is claimed by 3GPP that they struggle to avoid. The inconsistency is in security-related content across specifications, including those at different stages, for different releases and about different telecommunication generations (2G/3G/4G/5G). Even the attempt to address these inconsistencies can cause new misalignment, due to miscoordination among the 3GPP groups working on different documents.

Looking into how these SR-CRs are managed by 3GPP and affect protocol implementation, we observe a large window between their publications and proposed changes finally made to specifications. Such a window typically extends around 58 days, that exposes reported security weaknesses to the adversary and leaves a long time for an attack to happen. Even after the specifications were mended, we witnessed significant delays, which can be as long as 11 years, in updating implemented systems by device manufacturers and cellular network carriers. Also interestingly, we found that 14 weaknesses reported by SR-CRs end up being discovered in real systems many years later, while 6 of them are still out there today: not only has our experiment demonstrated their presence in popular mobile phones (Samsung Galaxy S10, Google Pixel 3 and Nexus 6P), but we also got the evidence for the existence of 1 weakness in real-world carrier networks (Section 4.2).

Also concerning is the trend of 3GPP security assurance. Over years, we observed the increase of SR-CRs, with the problems reported for the new telecommunication generation outnumbering those found in the old one. The presentation issues do not seem to improve over time either. Across releases, the attack window actually becomes larger, from 43 days for Release 4 to 71 days for Release 16. To mitigate the risks, we propose measures to improve the security assurance of the 3GPP ecosystem, including responsible handling of SR-CRs.

Contributions. Our contributions are outlined as follows:

- **New technique**. We developed a new NLP/ML pipeline that effectively identified from a large number of CRs those security-relevant. Our technique overcomes the challenge in labeling SR-CRs and is capable of capturing complicated SR-CRs. Not only has it enabled our measurement study, but it can also help enhance the security assurance of the 3GPP ecosystem, by flagging the CRs likely security-relevant and thus requiring special attention.

- **New findings**. We performed the first security analysis and measurement study on the 3GPP ecosystem, bringing to light surprising findings with significant security implications: e.g., difficulty in maintaining consistency across security-relevant content, large attack windows exposing published weaknesses, etc. We further propose improved procedures to better protect the ecosystem, which has never been done before.

2 Background

2.1 3GPP Ecosystem

Organization. 3GPP unites 7 telecommunication standard development organizations (e.g., ATIS [2], CCSA [3]) with the

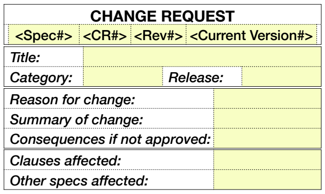
[1] All the 3GPP specifications, CRs can be found in 3GPP file server: https://www.3gpp.org/ftp

capability and authority to define, publish and set standards within the 3GPP scope in their nations or regions, 23 market representatives offering market advice and bringing market requirements (e.g., GSM [7], CTIA [5]), and 758 individual members (e.g., Qualcomm [9], Ericsson [6], Huawei [8]) committed to technical contribution to 3GPP specifications. These partners and members form *Technical Specification Groups (TSGs)* that prepare, approve and maintain 3GPP *Technical Specifications (TS)*. Now, 3GPP has 3 TSGs responsible for different functionalities: Radio Access Network (RAN) TSG, Service & System Aspects (SA) TSG, and Core Network & Terminals (CT) TSG. Under each TSG are several *Working Groups (WGs)*, such as RAN WG1 focusing on radio physical layer protocols, CT WG1 building the user equipment (UE) for core network protocols, and SA WG3 identifying the requirements and specifying the architectures and protocols for security and privacy in 3GPP systems.

Development methodology. Using the recommended stage methodology characterizing telecommunication services [54], TSGs develop specifications in 4 stages: stage 1 is an overall service description from the user's standpoint; stage 2 provides an overall description for network functions and capabilities; stage 3 defines network implementation, such as switching and signaling, which supports services specified in the previous stages; stage 4 is for testing. For example, SA WG3 produces TS 33.501 (security architecture and procedures for 5G System) for stage 2, which should be supported by stage 3 protocols, such as those for the user equipment to the core network (like TS 24.501 Non-Access-Stratum protocol for 5G system) developed by CT WG1.

3GPP organizes specifications into different *Releases*, each with distinguishable network capabilities and features, e.g., Release 8 for LTE and Release 15 for 5G. When all TSGs determine when a Release is *ready*, that is, all its features being defined and all its functionalities and required modifications being incorporated, they will declare that the Release is stable enough to be "frozen". Each Release development usually takes around 3 years. For instance, Release 8 was started in January 2006 and frozen in March 2009.

Change Request. Before a Release is formally frozen, the drafts of its specifications are published on the 3GPP file server. From that point on, all modifications on these specifications (even after the Release is frozen) need to be made through *Change Requests (CRs)*. A CR documents a proposed change raised by an *individual member* (e.g., Qualcomm), and brought to the attention of the responsible WG, which should pertain to a single technical topic only and relate to a specific version of a specification. In response to the WG's comments, the CR may undergo one or more rounds of revisions before approved by the WG and presented to the TSG. It may further go through additional changes upon request of the TSG, which makes the final decision on whether to approve the CR entirely without change or to reject or postpone unconditionally. If a CR is approved, a new version number of the specification will be allocated and published online. Figure 1 shows a CR's standardized front form. Each CR with a unique ID (e.g., *C1-094446*) contains relevant management information and proposed changes, such as the number of the target specification, its version and affected Release, the reason for the proposed modifications, the summary of how to change, and the consequences if the TSG does not accept it. Also, the form puts the CR into a certain *category*, including *A* (the change to ensure the consistency with another CR in a different category made to an earlier Release), *B* (addition or deletion of a feature), *C* (functional modification), *D* (editorial modification), and *F* (correction). Specifically, the category *F* is meant to correct a problem in the specification that might lead to an erroneous operation, an ambiguity in the specification that could cause wrong implementation, and other specification errors [12]. All the CRs including their revised versions are public on the 3GPP file server once they have been proposed to discuss at the (WG and TSG) meetings. The CR database on 16th, Aug 2021 shows 414,488 CRs, including 248,254 in Category *F*, which include all specification problems (e.g., security weaknesses) reported by 3GPP individual members in the history and therefore can be a valuable resource for understanding security hazards in the 3GPP ecosystem. Notably, these 248,254 CRs discussed only 166,657 different weaknesses. Thus, we only focus on the last CR for each weakness, and ignore their prior revisions talking the same weakness.



CHANGE REQUEST

<Spec#> <CR#> <Rev#> <Current Version#>

Title:

Category: *Release:*

Reason for change:

Summary of change:

Consequences if not approved:

Clauses affected:

Other specs affected:

Figure 1: CR front form template.

2.2 NLP and ML

BERT and domain adaptation. Bidirectional Encoder Representations from Transformer (BERT [20]) is developed as an NLP pre-training technique, which was originally trained on a combination of BOOKCORPUS [60] and English WIKIPEDIA, and has later been extensively utilized in many NLP tasks after fine-tuning. Fine-tuning BERT can be done through either domain-adaptive or task-adaptive pretraining [24]. Particularly, fine-tune with *Masked Language Modeling (MLM)*, which lets BERT predict randomly masked words in input sentences according to contexts, enables effective adaptation of the model to different domains. This approach is therefore incorporated into our CREEK pipeline (Section 3.3).

Positive-Unlabeled learning. *Positive-Unlabeled (PU)* learning is an ML technique for training a binary classifier using only positive and unlabeled data. Formally speaking, during training, we have labeled positive data ($\{x_i^{\mathsf{P}_{tr}}\}$) together with unlabeled data ($\{x_i^{\mathsf{U}_{tr}}\}$) but are not given labeled negative data. Suppose that those $n_{\mathsf{P}_{tr}}$ labeled positive data $\{x_i^{\mathsf{P}_{tr}}\}_{i=1}^{n_{\mathsf{P}_{tr}}}$ follow a distribution $p_{tr}(\mathbf{x}|y=+1)$, where $y \in \{+1,-1\}$ is the label of $\mathbf{x}$, and those $n_{\mathsf{U}_{tr}}$ unlabeled data $\{x_i^{\mathsf{U}_{tr}}\}_{i=1}^{n_{\mathsf{U}_{tr}}}$ follow a distribution $p_{tr}(\mathbf{x})$:

$$\begin{aligned} \{x_i^{\mathsf{P}_{tr}}\}_{i=1}^{n_{\mathsf{P}_{tr}}} \quad &\overset{i.i.d.}{\sim} \quad p_{tr}(\mathbf{x}|y=+1) \\ \{x_i^{\mathsf{U}_{tr}}\}_{i=1}^{n_{\mathsf{U}_{tr}}} \quad &\overset{i.i.d.}{\sim} \quad p_{tr}(\mathbf{x}) = \pi_{\mathsf{P}_{tr}} p_{tr}(\mathbf{x}|y=+1) \\ &\qquad\qquad\qquad + \pi_{\mathsf{N}_{tr}} p_{tr}(\mathbf{x}|y=-1) \end{aligned} \tag{1}$$

where $\pi_{\mathsf{P}_{tr}} := p_{tr}(y=+1)$ is the fraction of positive samples in the training data set (including the labeled and unlabeled samples), $\pi_{\mathsf{N}_{tr}} := p_{tr}(y=-1) = 1 - \pi_{\mathsf{P}_{tr}}$ is the fraction of negative samples in the training data set. The goal of the PU learning is to learn a classifier $g : \mathbb{R}^d \rightarrow \mathbb{R}$ that minimizes the expected *risk* on the testing data following the distribution $p_{te}(\mathbf{x},y) = p_{te}(\mathbf{x}) p_{te}(y|\mathbf{x})$:

$$\mathcal{R}^{te}(g) := \mathbf{E}_{p_{te}(\mathbf{x},y)}[\ell(yg(\mathbf{x}))] \tag{2}$$

where $\mathbf{E}_{p_{te}(\mathbf{x},y)}$ denotes the expectation , and $\ell(\cdot)$ is the loss function (e.g., the negative logarithm loss function). The ordinary PU learning [17,23,59] assumes that the positive labeled set has been *Selected Completely At Random (SCAR)* , and thus it follows the same distribution as the positive samples in the testing data set, i.e., $p_{tr}(\mathbf{x}|y=+1) = p_{te}(\mathbf{x}|y=+1) = p(\mathbf{x}|y=+1)$. However, this SCAR assumption may not hold in our SR-CR finding scenario, because bias may be present in the training data due to the limited knowledge of the experts (to some specific specifications). So we propose a new learning technique to address this challenge (Section 3.2).

Self-training. A self-training mechanism iterates a teacher-student training process till convergence: the base teacher model is trained on a labeled set, which is applied to a subset of the unlabeled data to generate their pseudo labels; a student model can then be learned on the combination of the labeled set and the pseudo-labeled set. At the center of this self-training process is how to select a representative subset of unlabeled data for producing the pseudo-labeled set. This problem has been studied in the prior research using predictive entropy [51], variation ratios [39], standard deviation and more recently using model uncertainty, such as *Bayesian Active Learning by Disagreement (BALD)* [28], which selects the unlabeled samples that maximize information gain (Eq. 10). In our research, BALD and [42] was used in our research to select representative unlabeled samples for self-training. (Section 3.3).

3 Finding Security-Relevant CRs

A CR is considered to be security-relevant (that is, an SR-CR) when it reports a problem that if not fixed, may allow security policies to be violated by the adversary. These security policies are meant to protect a system's confidentiality, integrity, and availability. For instance, *S3-180838* provides a protection mechanism to address an information leak risk that the permanent identity IMSI could be exposed to the passive or active attacker; *C1-183426* fixes a bidding down risk that a *User Equipment* (UE) could only receive the 4G-level security protection while the network provides the 5G service; *C1-094446* discloses a security weakness that the UE could accept a message without integrity protection, allowing a fake base station to disable the service of the UE.

Finding such SR-CRs is nontrivial. The straightforward method, keyword search, does not work well, with a low precision and a low recall (see the last paragraph of Section 3.4). Therefore in our research, we leveraged machine learning (ML) techniques to classify CRs and identified those security-relevant. Development of such an ML classifier, however, is nontrivial, due to the difficulty in labeling CR data, which relies on experts who are often only knowledgeable about some specifications. To address the labeling related challenges (as elaborated in Section 3.1), we designed and implemented an NLP pipeline, called *CREEK* (Section 3.2 and 3.3), and further reported our evaluations of the pipeline (Section 3.4).

3.1 Challenges in Finding SR-CRs

Challenge 1: small labeled dataset. As aforementioned, manual labeling of the 166,657 CRs in Category F is hard, due to the challenge in understanding the semantics of each CR, which requires in-depth knowledge about the related 3GPP specification. *C1-095712* presents an example, whose consequence is *"The entries may be incorrectly removed from the allowed CSG list causing persistent inability of UE to access a CSG cell."*. It is not easy to establish its connection with security due to the lack of knowledge about the CSG cell's functionality. To avoid the intensive labor involved in labeling, we searched the CR base with two keywords, "attack" and "vulnerability", and further manually inspected those discovered to identify the CRs indeed security-relevant. In the end, we labeled 301 SR-CRs in this way, which were later used to train the CREEK pipeline that found 1,270 SR-CRs.

Challenge 2: positive instances only. The keyword approach, unfortunately, cannot correctly locate non-SR-CRs. Random sampling the whole CR dataset for manual analysis is hard to ensure that a selected CR indeed has nothing to do with security and privacy, given the requirement for an in-depth understanding of all related specifications. So our NLP/ML pipeline has to be built upon positive instances only.

Challenge 3: biased training set. The labeled CRs selected using keywords may not follow the general distribution of SR-CRs across different specifications. This could undermine the effectiveness of the ML models trained on the data. In this study, we propose an enhanced PU Learning (Positive-Unlabeled Learning) model to address the bias.

3.2 Design

As discussed above, finding SR-CRs is a binary text classification problem with unlabeled data and a small set of positive examples that is biased. To solve this problem, we designed CREEK with the following steps: 1) embedding generation, 2) PU learning, 3) self-training. Here 1) and 3) are meant to

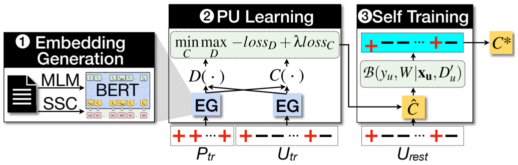


Figure 2: CREEK pipeline.

enrich the information carried by the small labeled set, while 2) addresses the constraint of positive instance only and the potential bias. Figure 2 illustrates our design.

Step 1: embedding generation. The first step is to transfer every sentence in each CR into an embedding, a feature vector of the same size that captures the key information of the input sentence with a various length. This purpose can be well served by BERT, which produces high-quality embeddings. However direct applying pre-trained BERT does not work well, due to its lack of domain specific information: we found that only 19.8% of the top 10K most frequent words (excluding stopwords) used in 3GPP CRs also appear on the top 10K list of the original BERT training corpus. So in our research, we fine-tuned a pre-trained BERT through huggingface [56] on all 3GPP specifications using two tasks – masked language modeling and binary classification for security-related specifications. Note that the second task is different from (and much easier than) finding SR-CRs: it is meant to determine whether a paragraph comes from a security-related specification explicitly labeled by 3GPP. This task could help our fine-tuned BERT gain knowledge about security-related nouns in 3GPP specifications including abbreviation, specification number, etc., and learn the language model of 3GPP CRs.

Step 2: PU learning. For the embeddings generated by Step 1, we need high-quality labeling for training a classifier. However, as mentioned earlier, we only have a small set of positive instances (Section 3.1) so we have to use *Positive-Unlabeled (PU)* learning to build the classifier. A problem here is that the SCAR assumption (Section 2.2) may not be held, as all these positive instances were found by keywords and therefore can have a different distribution than the testing distribution. Such a difference is called covariate shift [52], i.e., $p_{tr}(\mathbf{x}) \neq p_{te}(\mathbf{x})$, the probability tensity of training distribution is different from the probability tensity of testing distribution. Inspired by the prior research [29], we developed an adversarial learning framework with a classifier C and a discriminator D: D tries to recover the bias between the training distribution and the testing distribution, while C seeks an optimal separation between positive instances and negative ones with sample weights calculated from the bias recovered by D. After convergence, our classifier C learns how to figure out SR-CRs without the bias introduced by the keywords. Notice that, we utilized 10% of the CRs to train this classifier since training with all CRs would trap our model so it outputs negative labels for all unlabeled data, given that our positive instances were merely 0.2% of all CRs and are easily overwhelmed by the unlabeled data.

Step 3: self-training. After training a classifier $\hat{C}$ on 10% of unlabeled data, we further ran *Uncertainty-aware self-training (UST)* [42], a self-training algorithm, on the remaining 90% to refine the classifier. UST selects the unlabeled data with less uncertainty produced by $\hat{C}$ and measured by BALD. This self-training process helps $\hat{C}$ increase the distance between SR-CRs and non-SR-CRs, making it more robust.

3.3 Details and Implementation

Fine-tuning BERT. We use 3GPP specifications as the corpus for BERT fine-tuning. Specifically, we established two objectives: *Masked Language Model (MLM)* and *Security Specification Classification (SSC)*. The MLM objective is to train our BERT to predict randomly masked words in a sentence. We use the cross entropy loss for MLM objective. The SSC objective is to train our BERT to judge whether a given text belongs to security specifications. We use binary cross entropy function as the loss function for SSC. We defer details to Appendix A.1

PU learning with covariate shift. To train our SR-CR classifier, we leveraged an adversarial learning framework containing a discriminator D and a classifier C. D tries to recover the covariate bias, while C seeks an unbiased classifier with the help of the covariate bias recovered by D.

To recover the covariate shift [52], $w(\mathbf{x}) = \frac{p_{te}(\mathbf{x})}{p_{tr}(\mathbf{x})}$, we did following transformation:

$$\begin{aligned} w(\mathbf{x}) &= \frac{p(\mathbf{x}|\mathbf{x}\sim p_{te}(\mathbf{x}))}{p(\mathbf{x}|\mathbf{x}\sim p_{tr}(\mathbf{x}))} \\ &= \frac{p(\mathbf{x}\sim p_{te}(\mathbf{x})|\mathbf{x})p(\mathbf{x})/p(\mathbf{x}\sim p_{te}(\mathbf{x}))}{p(\mathbf{x}\sim p_{tr}(\mathbf{x})|\mathbf{x})p(\mathbf{x})/p(\mathbf{x}\sim p_{tr}(\mathbf{x}))} \\ &= \frac{p(\mathbf{x}\sim p_{te}(\mathbf{x})|\mathbf{x})}{p(\mathbf{x}\sim p_{tr}(\mathbf{x})|\mathbf{x})} \frac{p(\mathbf{x}\sim p_{tr}(\mathbf{x}))}{p(\mathbf{x}\sim p_{te}(\mathbf{x}))} \\ &= (\frac{1}{p(\mathbf{x}\sim p_{tr}(\mathbf{x})|\mathbf{x})} - 1)\frac{p(\mathbf{x}\sim p_{tr}(\mathbf{x}))}{p(\mathbf{x}\sim p_{te}(\mathbf{x}))} \end{aligned} \tag{3}$$

Note that, here we assume the testing and training data are random split, and thus $p(\mathbf{x} \sim p_{tr}(\mathbf{x})) = p(\mathbf{x} \sim p_{te}(\mathbf{x}))$. As a result, $w(\mathbf{x})$ is only related to the probability of a given $\mathbf{x}$ belonging to the training set, $p(\mathbf{x} \sim p_{tr}(\mathbf{x})|\mathbf{x})$. Using De Morgan's laws, we can further expand $p(\mathbf{x} \sim p_{tr}(\mathbf{x})|\mathbf{x}) = \frac{1}{2} + \frac{1}{2}p(\mathbf{x} \in \mathsf{P}_{tr}|\mathbf{x})$, where $p(\mathbf{x} \in \mathsf{P}_{tr}|\mathbf{x})$ is the probability of a given $\mathbf{x}$ belonging to the labeled positive set. Note that we hope to use the output of the discriminator $D(\mathbf{x})$ to approximate $p(\mathbf{x} \in \mathsf{P}_{tr}|\mathbf{x})$, and thus we get:

$$w(\mathbf{x}) \approx \frac{1}{\frac{1}{2}D(\mathbf{x})+\frac{1}{2}} - 1 = (1 - D(\mathbf{x}))/(1 + D(\mathbf{x})) \tag{4}$$

Empirically, we designed the following loss function $loss_D$ to let D learn the distribution of $p(\mathbf{x} \in \mathsf{P}_{tr}|\mathbf{x})$:

$$loss_D = -\sum_{i=1}^{m} \log D(x_i^{\mathsf{P}_{tr}}) + \log(1 - D(x_i^{\mathsf{U}_{tr}})) \tag{5}$$

where $x_i^{\mathsf{P}_{tr}}$ is the instance in the positively labeled training set and $x_i^{\mathsf{U}_{tr}}$ is the instance in the unlabeled training set. Notice that we replace the instance in the testing set with the instance in the unlabeled training set, as they have the same probability densities.

With recovered covariate shift $w(\mathbf{x})$, the testing risk (Eq. 2) of PU learning can be represented by:

$$
\begin{aligned}
\mathcal{R}_w^{tr}(g) &= \mathbf{E}_{p_{tr}(\mathbf{x})}[\ell(yg(\mathbf{x}))w(\mathbf{x})p_{tr}(y|\mathbf{x})] \\
&= \mathbf{E}_{p_{te}(\mathbf{x})}[\ell(yg(\mathbf{x}))p_{te}(y|\mathbf{x})] \\
&= \mathcal{R}^{te}(g)
\end{aligned} \tag{6}
$$

where $p_{tr}(y|\mathbf{x}) = p_{te}(y|\mathbf{x})$ and $w(\mathbf{x})p_{tr}(\mathbf{x}) = p_{te}(\mathbf{x})$ are used in the second step. Further, replacing g by our classifier C, the training risk $\mathcal{R}^{tr}(C)$ can be decomposed as following , with the help of the deduction in the prior work, Plessis et al. [21]:

$$
\begin{aligned}
&\mathcal{R}^{tr}(C) = \sum_i^m [-\pi_{\mathsf{P}_{tr}} \log C(x_i^{\mathsf{P}_{tr}}) \\
&+ |\pi_{\mathsf{P}_{tr}} \log(1 - C(x_i^{\mathsf{P}_{tr}})) - \log(1 - C(x_i^{\mathsf{U}_{tr}}))|]
\end{aligned} \tag{7}
$$

Together with $w(\mathbf{x})$, we define the loss function $loss_C$:

$$
\begin{aligned}
&loss_C = \mathcal{R}_w^{tr}(C) = \sum_i^m [-\pi_{\mathsf{P}_{tr}} \log C(x_i^{\mathsf{P}_{tr}}) w(x_i^{\mathsf{P}_{tr}}) \\
&+ |\pi_{\mathsf{P}_{tr}} \log(1 - C(x_i^{\mathsf{P}_{tr}})) w(x_i^{\mathsf{P}_{tr}}) - \log(1 - C(x_i^{\mathsf{U}_{tr}}))|]
\end{aligned} \tag{8}
$$

where $\pi_{\mathsf{P}_{tr}}$ is the ratio of SR-CR in the training set that can be estimated by using [33]. Empirically, we set $\pi_{\mathsf{P}_{tr}} = 0.2$. Combining $loss_D$ and $loss_C$, we get our final objective:

$$
\min_C \max_D \; -loss_D + \lambda loss_C \tag{9}
$$

where the discriminator D judges whether an instance comes from the training set, while the proposed model mutually trains a classifier C minimizing the training risk-weighted by $w(\mathbf{x})$ (Eq. 4) that takes into consideration the covariate shift caused by the bias in the labeled positive samples.

In practice, C and D were implemented using the same model structure as a two-layer bidirectional LSTM [30] with hidden size 256 and output size 64. We set a small $\lambda = 1e^{-3}$, making D learn the bias quickly. We take RMSprop optimizer [26] with fix learning rate $5e^{-6}$. We train 1000 epochs with batch size 256.

Self-training improvement. After training a classifier $\hat{C}$ through our PU learning algorithm, we leverage self-training on unlabeled data to strengthen the distinguishability of $\hat{C}$. Specifically, we use $\hat{C}$ to label those rest unlabeled data, and train our final classifier C^* on these pseudo-labeled data weighted by prediction uncertainty of $\hat{C}$. The prediction uncertainty was measured by cooperating BALD with [42]. Concretely, we let C^* focus more on those less-uncertain data predicted by $\hat{C}$. The details could be found in Appendix A.2

3.4 Evaluation

Following, we report our evaluation of CREEK, which aims at answering two questions: 1) how well does CREEK perform in finding SR-CRs? 2) how does each component of the pipeline contribute to addressing aforementioned challenges?

Effectiveness. To answer the first question, we ran experiments to analyze CREEK's performance from two aspects: its capability to overcome the biased positive labels and generalizability on various CRs. Notice that we keep the number of positives the same as that of negatives in each testing set we constructed.

• *Overcoming bias*. Limited by the three challenges (Section 3.1), our CREEK was trained on positively labeled instances found using keywords ("attack" and "vulnerability"). The bias brought by keywords is present not only in the training set but also in the single testing set. To reduce the impact of such bias when evaluating our approach, we utilized multiple testing sets with different keywords. For each set, we chose a security-related keyword (independent of those used to build our training set) and labeled the CRs carrying the keyword as positives and those without the keyword as negatives. Totally, we constructed six testing sets with six different security-related keywords and the `Overall` testing set where positives are CRs containing one of the six keywords[2] and the negatives are those containing none of these keywords. Note that we deliberately selected these keywords (e.g., "security threat", "malicious") to be specific, so as to ensure that the CRs carrying them are indeed SR-CRs (similar to those for constructing the training set) but their coverage is limited: altogether, 211 CRs containing these words.

The testing results of CREEK on these datasets are presented in Table 1. From the results, we can see that our CREEK achieves a high coverage (recall) on different testing sets, showing that CREEK can detect not only the CRs with training keywords but also those with unseen keywords. Together with the high recall, we also observe a stable and high precision on these datasets. This provides the evidence that CREEK is capable of overcoming the bias brought by positive-only training set.

• *Generalization to various CRs*. To further investigate the performance of CREEK on various CRs, we manually labeled 25 SR-CRs and 25 non-SR-CRs (see detailed SR-CRs on our website[3]) to construct the `Manual` testing set.

On these 50 manually labeled CRs, our CREEK correctly predicted 45 CRs (90.0% accuracy) with 3 false negatives and 2 false positives, and achieved 91.6% precision and 88.0% recall. Among those successfully predicted SR-CRs, we discovered that CREEK can figure out complicated SR-CRs that our experts need several hours to determine their security relevance. For instance, the consequence of *S3-040743* is *"MUK and MSK keys could be used during their validity time by another user inserting his UICC in the ME"*, which does not include explict security-related keyword that can serve as an indicator to build the CR's connection with security. However, by looking into relevant specifications (TS 33.246), one can learn that "MUK" and "MSK" are two keys for protecting the confidentiality of Multimedia Broadcast/Multicast Service traffic. This SR-CR reveals that an attack could be launched by reusing these two keys when another user inserts the SIM card to the victim UE from his mobile phone. Another example is *S3-091125*, which contains some sophisticated terminologies but no clues for its security relation. However, after

[2] We exclude the duplication for those CR containing multiple keywords.

[3] https://sites.google.com/view/3gpp-creek

Table 1: Testing results on different datasets.

	Denial of service	Security threat	Malicious	Spoof	Eavesdrop	Privacy risk	Overall	Manual
Precision (%)	96.2	100	100	89.1	100	100	96.3	91.6
Recall (%)	88.1	82.7	90.7	82.0	100	100	87.6	88.0
Accuracy (%)	92.3	93.1	95.3	86.0	100	100	92.3	90.0
Positives (#)	59	29	54	50	11	8	211	25

reading TS 33.220, a document with 96 pages, we found that the consequence of *S3-091125* implies a Denial-of-Service (DoS) attack since the NAF (Network Application Function, an element in the core network) will stop providing service to the UE due to the failure in retrieving its phone number (called MSISDN in 3GPP). One more example is *S3-151926* with the consequence that *"UEs may not be able to contact the PKMF to fetch their keys"*. After reading TS 33.303, we know that the PKMF stands for "Proximity-based Service Keys Management Function", which is used to provide a set of keys to protect the messages to UE. So, failure to fetch the keys from PKMF may lead to potential security risks, such as privacy violation. Correctly predicting these complicated SR-CRs indicates that CREEK has learnt security-related terminologies and the connection between them (thanks to our transductive learning).

Also, we found that CREEK can correctly predict SR-CRs with indicators missing in our training set. For instance, the indicator "confidentiality ... is compromised" is not present in our training set, yet CREEK correctly identifies the SR-CR *S3-020229* containing this indicator, which describes leakage of sensitive information, the IMS session keys.

As for the three false negatives, we ascribe to the obscure descriptions in CRs. Specifically, *C1-092847* contains "SA3", the item is highly related to security, however, appears neither in our training set nor in the 3GPP security-related specifications. Thus, without explicit labeling, the connection between "SA3" and security could not be established by CREEK. Similarly in *C1-051071*, the indicator is "stage 2", a term whose security implication has not been well specified. For *C1-100105*, the problem is missing details: one can hardly determine whether "absence of a general introduction" would lead to a security risk, considering that the word "absence" is ubiquitous in 3GPP CRs and mostly unrelated to a DoS risk.

We believe that the two false positives are caused by the lack of knowledge. Specifically, CREEK falsely labeled SR-CRs *R5-094440* and *R5-160901*, which are about failures in compliant UE. Without knowing that compliant UE refers to a testing scenario, these two CRs would easily be linked to DoS attacks.

Based upon the evaluation results, we believe that CREEK has been well generalized to cover various CRs, even complicated ones. This is important for determining the security implications of a CR. In our research, we spent one hour on average to label a complicated CR, showing the high cost of manual labeling (labeling 1,000 may take one month).

Ablation study and comparison. We further conducted ablation studies to evaluate the utilities of each component in our CREEK pipeline: embedding generation, PU learning and self-training. Their accuracies on the overall keyword searching testing set and manual labeled testing set were used as the metric. Our ablation studies' results are present in Table 2. Here, the "BERT-3GPP" column refers to our BERT model fine-tuned on the 3GPP corpus with the MLM and SSC tasks, and the "BERT" column refers to using the original BERT model. The accuracies reported in "BERT-3GPP" columns are always higher than those in "BERT" columns no matter which models follow the component on the pipeline, indicating that the 3GPP security-related information captured by our fine-tuning model helps generate more representative embeddings for the follow-up components to identify SR-CRs. Also in the table, rows "Ordinary PU" and "Ordinary PU + Self-training" refer to the ordinary unbiased PU learning (PAN [29]) and the rest two rows are about our biased PU learning. Comparing these rows, we found that our biased PU learning greatly outperforms the ordinary alternatives, improving the accuracy on both two testing sets. Note that without our new algorithm, the highest accuracy is only 67.3%, much lower than what can be achieved (92.3%) by using our PU learning, which indicates that our PU learning component successfully addresses the bias introduced by positively labeled instances only and the training data built upon two keywords ("attack" and "vulnerability"). Also, rows with "+Self-training" refer to the self-training component applied after the PU learning. The results demonstrate that self-training is more useful on non-optimum solutions, considering that it achieves an accuracy gain 2.2% for "BERT-3GPP" while 5.6% for "BERT" on the overall keywords testing set. A possible explanation is that the information gained by self-training has been partially obtained by fine-tuning BERT on the 3GPP corpus.

Table 2: Accuracy for ablation studies.

Accuracy (%)	BERT		BERT-3GPP	
	Overall	Manual	Overall	Manual
Ordinary PU	53.9	52.0	64.9	58.0
Ordinary PU+Self-training	51.4	56.0	67.3	62.0
Our PU	80.7	80.0	90.1	86.0
Our PU+Self-training	86.3	84.0	92.3	90.0

Comparison with keyword search. To understand the performance of simple keyword search, we manually selected 49 representative security-related keywords from SR-CRs. Specifically, we first came up with an ordered list of most frequent keywords from 1,270 high confident SR-CRs predicted by CREEK, and selected from the list 49 most frequently used security-related words (e.g., "attack", "security threat", etc.) out of total 5,000 frequently used words. The full list of the

Table 3: Statistics of keyword search.

Note: $\geq k$ column represents the results of keyword search by using the set of keywords that leads to the discovery of at least k confirmed SR-CRs out of 4 randomly selected CRs for the word.

	≥ 4	≥ 3	≥ 2	≥ 1	≥ 0
# of Keywords	14	20	31	43	49
# of SR-CRs (discovered)	313	576	1,500	3,798	7,869
Expected # of real SR-CRs	313	510.25	972.25	1,546.75	1,546.75
Estimated Precision	100%	88.6%	64.8%	40.7%	19.7%

49 keywords is posted on our website. Then we estimated the precision of each keyword through manual analysis of 4 randomly selected CRs carrying the word, and further analyzed the precision of applying a set of these keywords to find SR-CRs, when each word in the set leads to the discovery of at least 0 (≥ 0), 1 (≥ 1), 2 (≥ 2), 3 (≥ 3) or 4 (≥ 4) confirmed SR-CRs out of the 4 randomly selected CRs for the word. The results are present in Table 3, with more details provided on the website.

Using each keyword's precision and the number of CRs containing the word, we further estimated the precision of keyword search using a given set of keywords by calculating the proportion of the expected number of real SR-CRs to the total number of identified CRs. For example, for the set of ≥ 3, 313 CRs are discovered by 14 keywords with a precision of 100% (=4/4) and 263 (=576-313) additional CRs are found using 6 keywords with a precision of 75% (=3/4); so, the expected number of real SR-CRs is 510.25 ($= 313 * 100\% + 263 * 75\%$), and the estimated precision of the whole set is 88.6% ($= 510.25/576$). At this precision level (88.6%), which is comparable but a bit lower than that of CREEK (91.6%), we found that the keyword search reports much fewer expected real SR-CRs than CREEK (510.25 vs. 1163.32), as shown in Table 3. Further, when we use the set of the keywords (≥ 2) that detect fewer (but a similar number of) expected SR-CRs compared with CREEK (972.25 vs. 1163.32), the precision of the keyword search drops to 64.8%, which is much lower than that of CREEK at 91.6%. If we use all 49 keywords, the precision goes further down to 19.7%. This indicates that the simple keyword search is inadequate for effectively identifying SR-CRs.

4 Analyzing SR-CRs

In this section, we report our measurement study on 1,270 SR-CRs detected by CREEK with high confidence (> 0.99) to understand the security hazards in the 3GPP ecosystem.

4.1 Security Quality of Specifications

Security consequences. Previous studies [16, 27] report that erroneous and problematic content of 3GPP specifications could lead to vulnerable implementations, which can be exploited for Denial of Service (DoS) or private identity leaks. By leveraging the discovered SR-CRs that record security-related specification problems reported during their development, we are able to gain a more comprehensive understanding about their security quality and impacts of their weaknesses. Specifically, as Figure 1 shows, every CR has a field **consequences if not approved** , which is supposed to explain the potential harms once the weakness is attacked. In practice, however, some SR-CRs only roughly mention that the weaknesses they document could be used to violate a security requirement without providing any detail. So in our study, we inspected 1,270 SR-CRs and selected from them 616 SR-CRs carrying detailed consequence fields.

Manually analyzing these SR-CRs, we classified their consequences into five categories, as shown in Table 4, including DoS attack, sensitive data leak, failure to prevent attacks, battery draining, and overcharge. Our research shows that the most common consequences are those related to service interruption (DoS), which can happen both on the UE and the core network. For instance, SR-CR *C1-094446* reports a weakness in TS 24.301 that could be used by a malicious base station to kick a UE out of service until the user reboots her device. Less severe but still disruptive is the exploit related to *C1-154301*, which locks the UE out of the Packet Switched (PS) service so it cannot use IP data but can still make phone calls. Also pervasive is data leak: SR-CRs in this category expose sensitive information, such as the UE's locations, private identities, certificates, ciphering mechanisms, and even security keys. Further, 3GPP has a special WG SA3 that defines security and privacy requirements, architectures, and protocols for 3GPP systems, which are meant to mitigate threats like DoS, man-in-the-middle (MITM) attacks and etc. However, from the SR-CRs in the third category, we found that erroneous specification content potentially results in a system without proper protection against such attacks. For example, *C4-191528* reveals an error in the OAuth token defined in TS 29.510, which could lead to failure in the PLMN verification, rendering the system unable to prevent an impersonation attack. Moreover, we found that some specification weaknesses (e.g., *S3-120336* and *S3-142116*) could cause battery draining and some others can be exploited to make free phone calls or overcharge a target victim UE. These findings demonstrate significant security impacts of 3GPP specification errors.

Causes. Given the serious security consequences of specification weaknesses, it is important to understand their causes, which tend to be clearly documented by each CR in the field **reason for change** as Figure 1 shows.

So, in our research, we manually inspected 1,270 SR-CRs and our findings are summarized in Table 5. As we can see, most issues are related to design errors (70.55%), as expected, but surprisingly, the remaining specification problems (29.45%) are caused by problematic presentation. We elaborate our findings as follows.

The design errors are found to be rather diverse, but most of them are reported in the procedures meant for security protection (such as integrity protection) and the rest in the procedures that need to be safeguarded. As an example, *S3-080841* documents a design flaw in key derivation when a UE moves from one cell to another, which could cause potential key reuse and authentication circumvention. Also our

Table 4: Categories and examples of exploit consequences.

Category	Ratio	Example	CR
DoS/DDoS/Bidding down attack	60.63%	*Cause the UE to be vulnerable to DoS attack by a malicious base station.*	C1-094446
		Cause the UE being out from Packet Switched (PS) services.	C1-154301
Sensitive data leakage	18.69%	*Cause the UE to be tracked and traced.*	S2-2006202
		Cause the UE to expose the ciphering mechanism.	S3-020689
Failure of preventing attacks	15.52%	*Cause preventing impersonation attacks not be supported.*	C4-191528
		Cause the network cannot recognize and tackle man-in-the-middle attack.	C1-091594
Battery draining	2.87%	*Cause the UE to lose power.*	S3-120336
		Cause a lot of UEs being busy and wasting UEs' electricity.	S3-142116
Incorrect charging	2.30%	*Cause the UE to be open to an over charging attack.*	C1-122414
		Cause the UE can have free talk.	C1-160432

Table 5: Summary of causes on SR-CRs.

Category I	Ratio	Category II	Ratio
Design error	70.55% (896/1,270)	Error on security procedures	≈60.00%
		Error on other procedures	≈40.00%
Problematic presentation	29.45% (374/1,270)	Lack of details	68.45%
		Inconsistent specifications	31.55%

Table 6: Categories of misalignment in 3GPP specifications.

Category	Ratio
Violate inconsistency between stages	50.85%
Violate inconsistency between Releases	5.08%
Violate inconsistency between generations	3.40%
Violate inconsistency in a single specification	40.68%

analysis shows that for the procedures serving other telecommunication functionalities than security, their protection can be inadequate, erroneous or oftentimes completely missing, as some security risks may have never been seriously considered during specification development. For example, the *paging* procedure in the NAS protocol is designed to waken an idle UE in response to an incoming call or message; it is found to lack authentication protection, which enables a UE impersonation attack (*C1-135219* in TS 24.301).

In the 896 SR-CRs about design errors, we found that 193 are meant to fix the content issues in the 33 series specifications – a set of documents that focus on 3GPP security aspects as mentioned before (Section 3.2). So they are clearly related to security procedures. The nature of the procedures associated with other SR-CRs, however, cannot be easily determined, due to loose descriptions of CRs, whose connections with specifications are established through nothing but a few keywords like "privacy", "encrypt" and etc. To find out whether these procedures are security-related, we randomly selected 50 SR-CRs and looked up their keywords in related specifications to understand their context. In the end, we found that 26 of them are meant to fix security-related procedures, such as *R5-190434* fixing EAP-AKA based authentication procedure in TS 38.508-1, while the rest are for other procedures that miss necessary security protection. Altogether, we estimate that about 60% of the design errors reported by SR-CRs are inside security procedures.

Among the 29.45% SR-CRs caused by problematic presentation, we found that 68.45% (= 256/374) are due to the lack of details about how to implement security-related functionalities. For instance, SR-CR *S3-171530* points out that TS 33.401 in version 14.2.0 introduces a hash parameter $HASH_{MME}$ in a message to prevent the bidding down attack, but the specification does not give the calculation method to the element, which causes confusion about how to implement it and may lead to a security weakness. Another example is *C1-094810*: TS 24.301 in version 8.3.0 requires to compute two keys on the UE side during authentication, but fails to specify whether the inserted SIM card or the mobile device actually generates the security parameter. Such unclear specifications may result in wrong implementations exposing security parameters.

What is interesting is the rest of SR-CRs (31.55% = 118/374) all about inconsistent descriptions (which is called *misalignment*) in specifications, with 24 (6.42% = 24/374) of them for addressing conflict statements. For instance, from *C1-101068* we discovered that in TS 24.301, S4.4.4.4 requires that any message without security protection shall be ignored after security context is established, but S4.4.2.3 conflicts with the requirement, allowing to accept some messages without protection. This inconsistency could mislead the developer and introduce security flaws exploited by those without the right security context to launch an impersonation attack.

Since the inception of the consortium, 3GPP has always claimed that they strive to keep the consistency of the specifications under the responsibility of different TSGs through a manageable mechanism to handle document updating [12]. For this purpose, 3GPP requires the originator of a CR to thoroughly examine its impact on other specifications. Actually, each CR has a field called **other specs affected** to help keep consistency across specifications (see Figure 1). However, we found that the field has been left empty on most CRs in the 3GPP database. In the meantime, the content of CRs shows that this misalignment problem not only exists but is also serious and pervasive in 3GPP specifications. Following we report our study on this problem.

Misalignment in specifications. Our research shows that the misalignment problem not only appears in a single specification but across specifications. As shown in Table 6, we found that 60 SR-CRs report inconsistent security-relevant content across the specifications at different stages, in different Releases, and for different generations. As mentioned earlier (Section 2), 3GPP uses a stage methodology in developing telecommunication services. Since stage 2 specifies network function requirements, all specifications at stage 3 are expected to support them. However misalignment problems are found to be pervasive across the stages and many CRs are issued to address them. For example, TS 33.401 (a stage 2 specification) requires the UE to modify its CSG list (that decides which cell groups the UE can access) only when receiving commands with security protection; however, a stage 3 specification TS 24.301 violates the requirement, allowing a UE to delete the CSG list when it receives a reject message without the safeguard of integrity check, which could be exploited by a fake network for a DoS attack on victim UEs (see *C1-095554*). Such inconsistent issues are quite common between security requirements specified at stage 2 and implementation aspects at stage 3, being reported by 56.67% (34/60) of SR-CRs about cross-stages misalignment.

3GPP systems are backward compatible: for example, a mobile system developed based upon Release 12 should also be supported in later Releases. This property requires some levels of consistency between two Releases so they can maintain the compatibility. However, from SR-CRs, we found multiple instances of consistency violations, e.g., resetting downlink sequence numbers required in one Release but not in another, which could lead to replay attacks (*C1-101252, C1-213353*). Also, across generations of telecommunication, except for new features, functionalities should be compatible and their descriptions should be consistent. For example, *UECapabilityEntry* in RRC protocol needs be encrypted before transmission to avoid exposure; this protection is introduced by 5G but 3GPP requires the 4G specifications to be consistent with 5G for this protocol, which however had not been addressed until *R2-2002094* was issued 2 months later.

With years of effort claimed by 3GPP, the misalignment issues are still pervasive. Our hypothesis is that the problem has fundamentally been caused by 3GPP's management of CRs, which tend to be reported by individual members, particularly manufacturers, to different WGs. From the discovered SR-CRs, we observed that the misalignment of a CR proposed by one member with other specifications has usually been identified by a different member. For example, *S2-112468* was submitted by Vodafone and updated to TS 23.401 but the misalignment with TS 24.301 caused by the CR was reported by ZTE through another CR *C1-112469* one month later. It is likely that the CR originator, usually a device manufacturer, reports only the problem it encounters in implementing part of the specifications, expecting 3GPP to take care of the consistency issue, which however often falls through the cracks.

Even when related specification content has been identified, and multiple CRs are submitted to ensure consistency, this effort could be impeded by the 3GPP working procedure. Consistency across different specifications requires coordination among multiple 3GPP WGs. This does not seem to work well now. Particularly, these groups have different working schedules, which causes related changes to be updated at different times, rendering the specifications misaligned for a period of time. For example, to secure *UECapabilityEnquiry* in RRC protocol, *S3-192862* was submitted to SA WG3 on 2019/8/19 and a related CR *R2-1909394* was sent to RAN WG2 on 2019/8/16. *S3-192862* moved on quickly in SA WG3 and its modification was updated to TS 33.501 on 2019/09/25, while RAN WG2 took a long time to review *R2-1909394*, which was finally approved and updated to TS 38.331 on 2020/03/31. Therefore, there is a 7-month gap during which TS 33.501 and TS 38.331 were inconsistent.

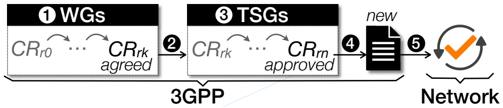


Figure 3: CR-processing procedure and system patching.

Misaligned security-relevant content may bring in security risks, when an SR-CR is published while some related content has not been discovered and updated for a long time, even after the original change has been made to the specification. We will discuss this problem in Section 5.

4.2 CR Management and System Patching

3GPP publishes every CR immediately after its submission, thus exposing the content of an SR-CR to unintended eyes, which could lead to an exploit on a real-world system containing the reported security weakness before it is patched. To understand this risk in today's 3GPP ecosystem, we looked into how SR-CRs are managed. Figure 3 shows the whole procedure from proposal of a CR to the integration of its requested changes into commercial systems. As we can see, after submitted by an individual member, a CR is first reviewed by a WG, which often requires several rounds of revisions before approval. Then it goes through a similar revision process at TSG meetings before its content is updated by the support team to a new specification version. The specification will later be used by telecommunication developers around the world. This procedure indiscreetly discloses security-critical information. To understand its security implications, we measured the length of an *attack window*, conservatively from the publication of an SR-CR on the 3GPP server to the update of its content to the target specification, and precisely until patching of its related systems (which tends to be more difficult to determine). Following we elaborate on our findings.

Dataset. To measure the attack window, again we started with the 1,270 SR-CRs. From them, we first removed those withdrawn by their originators or not accepted by the WG or TSG reviews, which leaves us with 817 CRs. Further, we dropped those proposed before their target specifications were frozen, since most device vendors only implement the specifications when they become stable (frozen), though exceptions do exist (e.g., Qualcomm and Huawei built their devices based upon the Releases yet to be stabilized [13, 49]). Altogether, there are 462 SR-CRs proposed for frozen Releases, including 443 CRs submitted to WGs and 19 CRs directly to TSGs[4].

[4] This happens when a specification is handled directly by a TSG, instead

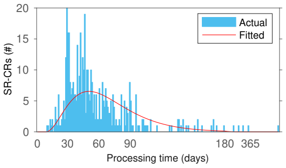


Figure 4: Attack windows in 3GPP.

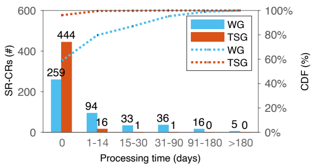


Figure 5: Processing time in WGs and TSGs.

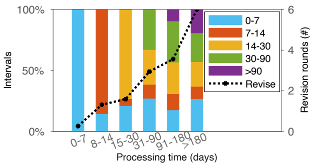


Figure 6: Causes of processing time in WGs.

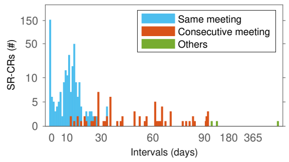


Figure 7: Intervals in WGs considering meetings.

Attack window. Figure 4 illustrates the distribution over the length of our "conserative" attack window (from proposal of an SR-CR to the update of the specification). As we can see, most SR-CRs (90.48% = 418/462) have been processed within three months, 69.48% (321) in two months, and 21.43% (99) in one month. Among the 462 SR-CRs, the minimum attack window is 10 days (*RP-010274*) while the longest one lasts more than two years (*C1-105068*). The expected length is around 58 days, which we calculated by approximating the distribution with *Negative Binomial Distribution* [22]. To understand this distribution, we looked into each stage of the CR-processing procedure to find out how long 3GPP needs to process a CR. As highlighted in Figure 3, we analyzed ① how long the WG takes to process a CR; ② what is the waiting time between a CR agreed by the WG and the review on it started at the TSG level; ③ how long the TSG takes to process a CR; ④ what the delay would be for an approved CR (by the TSG) to be updated to its target specification.

For ① , an SR-CR can quickly be approved by its responsible WG or go through multiple rounds of required revisions. Figure 5 shows the distribution of SR-CR processing time at the WG level. Among the 443 SR-CRs, 58.47% are agreed by WGs directly and 87.13% get approval in one month, which is within the duration of a single WG meeting. The rest 12.87% SR-CRs take a much longer time to approve at the WG level. For example, we found 5 SR-CRs that each had been reviewed by WGs for more than 6 months, and one took 2 years before it was finally agreed by a WG. Such a long delay is caused by multiple rounds of required revisions or a large amount of time invested in revision, which extends the time *interval* between a CR's two submission rounds. Figure 6 shows that the CR-processing time grows when the number of revision rounds or the interval between rounds goes up. For example, before the WG approved *C1-095712*, it had been revised 10 times, with the longest interval being 127 days (between *C1-091323* and *C1-092720*). Figure 7 further explains why some interval is so long: if two consecutive versions of the same CR (due to the required revision) are reviewed at the same WG meeting, on average it takes just 7 days; however, when these versions are discussed at two consecutive WG meetings, 48 days on average and 98 days at most are needed, as a WG meeting may take place every one/two/three months. Interestingly, we found that 3 SR-CRs even waited for more than 90 days before their revisions were reviewed again. This is because each of them was "postponed" by its related WG, and later revived, which causes a huge delay in their processing.

For ② , after an SR-CR has been approved by the WG, it needs to wait to be presented to the responsible TSG so it can be discussed at the most recent TSG meeting. Figure 8 illustrates the distribution of such waiting time. As we can see, most of SR-CRs can be processed within 3 months (104 days at most), since the TSG plenary meeting is held quarterly every year. On average, each SR-CR needs to wait for 22.8 days according to our analysis. Note that such a delay is inevitable given the current CR-processing procedure.

For ③ , the TSG may also require a CR to be revised before approving it. Figure 5 illustrates the distribution of SR-CR processing time at the TSG level. Among the 462 SR-CRs, all except 18 CRs were approved by the TSG directly without any revision. Among these 18 CRs, *CP-090678* was "postponed" by CT TSG 45 and later approved by the next meeting (CT TSG 46), which took 77 days in total. All other SR-CR revisions successfully got through the review during the same meeting. So the expected SR-CR processing time at TSGs is 0.4 days.

Finally, for ④ , a TSG-approved SR-CR still needs to wait for the 3GPP support team to update its content to a new specification version. Figure 9 illustrates the distribution of such waiting time: most such updates (91.13% = 421/462) happen within a month, 99.57% in 2 months, and only 2 SR-CRs wait for more than 2 months (but still within 3 months). In our research, we found that on average, a change to the specification is done within 21.3 days after its CR's approval.

Altogether, we found that the procedure for CR approval and specification amendment is complicated and time consuming: on average, an SR-CR takes 58 days before it is applied to a Release. The length of this conservative attack window (not to mention the further delay before patching of real-world systems), coupled with 3GPP's indiscreet publication of SR-CRs, constitutes a serious (yet overlooked) security risk, allowing the adversary to attack today's telecommunication system using the information of published SR-CRs.

Impact of related SR-CRs. Further extending the attack window is the presence of related SR-CRs. Among the 462 SR-CRs, 53 are related to at least another SR-CR. We found that such a relation falls into three categories. First, 22 of them are meant to address the misalignment issue across different specifications: when one SR-CR requests a change to one specification, additional CRs are issued to fix related content in other specifications. Second, we identified a group of 29

of WGs underneath the TSG.

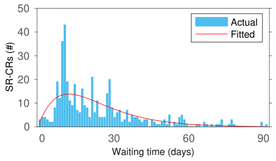


Figure 8: Waiting time to TSGs from WGs.

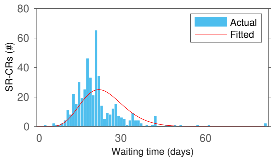


Figure 9: Waiting time to update to specifications.

CRs among which each is connected to at least one another CR in the group since one or more of them are used to patch the problem not fully addressed by the other. For example, *S3-161217* suggests using a hash value $HASH_{MME}$ to protect a message against unauthorized changes; however, no details about how to calculate the hash is given by the CR, which is provided by *S3-171355* proposed almost one year later in May 2017. The story does not end here. The integrity protection was later found to be described in a wrong way in TS 33.401: the specification only requires the message *carrying the hash value* to go through the integrity check, while those *without* the hash completely fall through the cracks. So an additional CR *S3-173080* was proposed to fix it on 2017/11/20.

Furthermore, we found 2 SR-CRs meant to fix similar problems in different services. The Paging procedures for CS fallback and EPS services are similar (but not identical). In November 2013, *C1-135219* was issued to fix a collision risk in the Paging procedure for CS fallback in TS 24.301. One year later, another CR *C1-141405* was proposed to address the same problem in the Page procedure for EPS services.

The presence of related SR-CRs extends the attack window: information exposed by the first published SR-CR can potentially be used to exploit the targeted security weakness and its related problems until all of them have been fixed by follow-up CRs. So we estimate the window for such related SR-CRs from the date the first SR-CR issued to the time the last affected specification updated. Figure 10 shows the attack windows for the 53 SR-CRs based upon their relations. As we can see, 38 of them were addressed in 1 year, with most of them (24) updated to specifications taking 3-6 months, 11 related SR-CRs were patched using 1-2 years, and 4 took more than 2 years to fix. To our surprise, 2 related SR-CRs on a security risk in the *Service Reject* message of TS 24.301, which was first proposed in August 2009 (*C1-093567*), were not fully applied to the specifications until June 2014 (*C1-141834*)! So the security weakness has been exposed to the public, without protection, for almost 5 years (1,761 days).

Real-world studies on precise attack windows. The conservative attack window just estimates a lower bound for the duration in which the security weakness exposed by a CR could be exploited. A more accurate assessment of the risk should take into account the delay introduced by patching implemented systems after related specifications are fixed (⑤ in Figure 3), which often takes a long time. However, finding out this delay is nontrivial, due to the lack of the information about the versions of specifications implemented by commercial cellular networks and UEs (e.g., mobile phones). In our research, we resorted to reported vulnerabilities in telecommunication systems and our experimental analysis to unravel this myth.

Among the weaknesses reported by the 443 SR-CRs (which were all accepted by 3GPP and published after the corresponding specifications were frozen, as described before), 13 have been discovered in commercial systems by prior research [16, 31, 36, 50, 58] after they were fixed in the 3GPP specifications, with 5 of them still observed from some devices in our experiments. Interestingly, *there is no evidence that the researchers ever realized that the implementation problems they uncovered actually come from the specification weaknesses known years before.* Due to the ethical constraints on evaluating our findings on real-world carrier networks and the failure of current simulators to support many functionalities (e.g., handover, emergency), most SR-CRs cannot be verified. In the end, we were only able to validate the weaknesses related to the NAS/RRC protocol basic procedures, such as Attach procedure and Paging procedure. Further we found one additional problem in our experiments, which has *never* been reported in real systems, up to our knowledge. Table 7 presents our findings. Here we conservatively consider that the end of the attack window should be extended to at least the release date of a vulnerable device or the time the problems confirmed in our experiments, or the year the related papers were published (for the network flaws we could not verify).

Compared with the conservative window (Figure 4), this more precise window turns out to be much larger, 0.7 years at least. To our surprise, the largest one even extends over 11.7 years. Specifically, in 2009, *C1-094446* has been applied to TS 24.301 to fix a security weakness that the UE could accept a DETACH REQUEST message without integrity protection, allowing a fake base station to disable the victim UE. However, 11.7 years later, our experiment confirmed this exact problem in a mobile phone (Nexus 6P). Such a large window gives attackers sufficient time to exploit the published vulnerabilities in the real world.

Our experiment analysis. We developed a testing environment to find out whether security weaknesses reported by SR-CRs and fixed in specifications are still out there in today's systems, including both UEs and core networks on the carrier side. Specifically, for UEs, we inspected three mobile devices, including Samsung Galaxy S10, Google Pixel 3, and Nexus 6P. For this purpose, we used an SDR board (LimeSDR USB v1.4) connecting to a computer that runs a simulator (srsRAN [11]) that acts as both the base station and the core network to issue attack messages. Also, we connected these phones to SCAT [10] through the USB bridge to monitor their states, to determine whether these UEs have been attacked successfully. Through our experiments, we found 5 security risks reported by SR-CRs years ago, yet still unpatched on these devices. Actually, for 3 of these risks, each is present

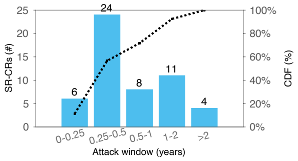


Figure 10: Attack window for the related SR-CRs.

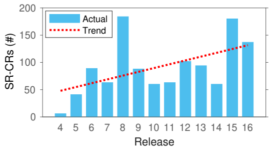


Figure 11: Change of SR-CRs over releases.

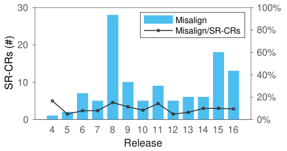


Figure 12: Misalignment SR-CRs over releases.

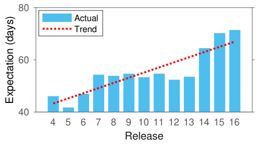


Figure 13: Attack window expectation over releases.

on 2 to 3 devices we evaluated. For example, on 2009/12/17, *C1-095712* was officially applied to TS 24.301, fixing the problem that a UE's CSG list could be modified by an unauthenticated reject message. However, we confirmed that the risk still exists in Samsung Galaxy S10, which reveals a large attack window of at least 11 years! Note that the same problem was reported in 2019 on iPhone 6, though no evidence here shows that the authors had known the SR-CRs and therefore the fundamental cause of the implementation error [58]. We have reported the risks to the manufacturers of these three device and received confirmation from all of them. Other findings are in Table 7.

For the carrier's network, we could not exploit it to verify the presence of weaknesses, due to ethical constraints. So we used a non-intrusive approach, by inspecting the downlink traffic from the core network to the UE using SCAT [10]. In our research, we passively analyzed the traffic of three major commercial carriers (anonymized to protect them) and found that one problem reported by an SR-CR still exists in their core networks. It is missing of $HASH_{MME}$ suggested by *S3-161217*, as mentioned earlier. This protection is still not there in the core networks, since we could not find $HASH_{MME}$ in the response to Attach Request, as required by TS 33.401. As a result, it is under the threat of MITM. We have reported our findings to authorized parties.

4.3 Security Trends of the Ecosystem

Changes of SR-CRs over time. To understand whether the security quality of 3GPP specifications improves over time, and how likely disclosed CRs only involve a subset of security flaws, we analyzed the trend of SR-CRs by looking back at the history. For this purpose, we first found out the number of SR-CRs for each Release. As Figure 11 shows, there is a slow upward trend for the number of SR-CRs from Release 4 to Release 16, which indicates that new Releases tend to have more security weaknesses. Notably, from Figure 11, it is easy to see big bumps of SR-CRs for Release 8 and Release 15, which introduce LTE and 5G respectively, two key stages for the telecommunication evolution. As new techniques emerge, more security risks tend to be discovered from their specifications at the same time.

Second, one way to estimate the unknown security weaknesses still present in a newly published specification is to find out the duration during which the releases continue to receive SR-CRs. To this end, we measure the number of SR-CRs have been reported over time for a specification after it was frozen. From the results (shown on our website), we found that for each Release, SR-CRs have been continuously issued for several years. For example, Release 4 was frozen in 2001 but has been reported for various security flaws for 12 years. From Release 4 to 11, which are no longer receiving SR-CRs, they had been modified for 7 years on average after their specifications were frozen. So we have reason to believe that there still are many unveiled security weaknesses in Release 12 and after, and the effort to improve their security quality will last for a long time.

Changes of inconsistent specifications over time. Inconsistent descriptions in specifications are pervasive and may further result in serious security consequences, as analyzed in Section 4.1, even given 3GPP's effort to address this issue. Figure 12 shows the number of SR-CRs reporting misaligned specifications over different Releases and the percentage of such SR-CRs in every Release's total SR-CRs. From the figure, we observe that the number of misalignment-related SR-CRs varies over different Releases with two peaks at Release 8 and 15, but are relatively steady in terms of percentage. Again, the bumps (two peaks) in the figure are likely due to the sudden rise inconsistent issues that come with publications of the complicated documents for revolutionary technologies. Meanwhile, the steadiness of the percentage curve in general gives no sign that the pain of security-relevant misalignment would get better in the near future.

Changes of attack windows over time. Our research reveals long attack windows for SR-CRs because of the CR management (Section 4.2). In Figure 13, we show the expected lengths of such windows based on the fitting results by applying Negative Binomial Distribution [22] for each Release. From the figure, we observe the uptrend of the window sizes across Releases, from 43 days for Release 4 to 71 days for Release 16. This trend not only indicates that the attack window may continue to grow for future Releases, but reveals that the security implications of attack windows become even more serious over time, with the published system weaknesses exposed to the adversary for a longer time.

5 Discussion

5.1 Lessons Learnt

Lesson 1: Misalignment is an important security risk. 3GPP claims that they try to ensure the consistency of specification by their management [12], requiring the member submitting a CR to indicate other related specification content through a CR field (**other specs affected**). However, our research shows that the misalignment problem is still pervasive in 3GPP specifications, potentially with serious security consequences: inconsistent security-relevant descriptions in specifications indicate the presence of erroneous content that once followed during system implementation could introduce security weaknesses (e.g., *C1-095554*); also in the case that a vulnerability described by a published CR for a specification still exists in a different specification, the telecommunication system implementing the latter is posed to grave danger, with its security weaknesses completely exposed (Section 4.1).

Fundamentally addressing this problem requires technical support. We believe that cross-references for at least security-related content should be in place for 3GPP specifications. Facilitating this effort are NLP/ML techniques that assist protocol development, in terms of technical content indexing and misalignment discovery, which should be studied in the future.

Lesson 2: Attack window should be controlled. From the publicity of an SR-CR to its mend in the target specification, the 3GPP ecosystem shows a 58-day delay on average. This practice even violates 3GPP's own responsible vulnerability disclosure rule, as described in the 3GPP Coordinated Vulnerability Disclosure requirement ("*not to share knowledge of the vulnerability with third parties until 3GPP has resolved it*") [4]. Hence, we suggest that 3GPP handle SR-CRs following the same responsible disclosure rule: publishing the vulnerability information of SR-CRs *no earlier than* the security issues reported by the CRs have been updated to the target specifications. For this purpose, SR-CRs should first be identified so they can be handled in a more responsible way. One possible way to do so is asking the originator to label the security relevance of a CR. In the case that 3GPP wants to double-check unlabeled CRs, automatic CR classifiers like CREEK could be leveraged.

Even after the specification updates, we observe a long delay (up to 11 years) before the real-world system is patched if this has ever been done by vendors. So it remains a challenge to motivate vendors to follow specifications, reacting timely to the released security mends. A possible solution is, for 3GPP, to issue well-designed conformance test cases, particularly for its security updates, which enforces vendors' implementation compliant with specifications [45]. Recent studies [35, 43] show the effectiveness of the conformance test in finding the security problems of mobile devices.

Other lessons. Our measurement study reveals the dominance of DoS risks among all security consequences of SR-CRs (60.63%), in line with the findings reported by prior researches, which are mostly DoS related vulnerabilities [16, 31, 36, 50, 58]. This indicates that DoS will continue to be a major threat to carrier networks. Also discovered in our research, there are 70.55% SR-CRs about design errors (Section 4.1). So we believe that an important direction is to facilitate automated construction of a protocol model from specifications, for the purpose of formal security verification.

5.2 Limitation and Discussion

Finding the exact and also complete set of SR-CRs from 400K CRs is extremely difficult, due to convoluted descriptions of these CRs that are often hard to decipher even to human experts. CREEK is the first step toward the full discovery of these SR-CRs, but the capability of our current design and implementation is still limited. The best we could do now is to capture a reasonable number of SR-CRs (1,270) with a relatively high precision (91.6%). The real set of the CRs with security-relevant content is larger.

Among all 1,270 SR-CRs discovered in our research, 453 have not yet been accepted. In-depth analysis of the ecosystem of these SR-CRs is an important issue that has not been covered by our study. It is known that 3GPP *deliberately* leaves some security vulnerabilities (including those in SR-CRs and the problems reported by research papers [41, 57]) unfixed in the specifications or allows protection to be optional, due to various reasons, e.g., performance impacts or implementation complexity [19]. Future research should revisit these problems, both to seek more effective solutions and provoke a public discussion so the presence of these ticking bombs will not be just swept under the rug. Furthermore, considering the chaos that several SR-CRs fixing the same weakness may receive different decisions (accept or not) over time, more works need to be done to understand what kind of vulnerabilities are still unfixed in 3GPP specifications.

6 Related Work

Cellular network vulnerabilities. Previous studies have revealed various security risks in cellular network systems, including both implementation errors and design issues. For example, previous works [18, 27, 35, 36, 50, 58] report both commercial mobile phone and core network implementation vulnerabilities, containing DoS, location tracking, spoofing attack and others. In the meantime, other works focus on discovering security weaknesses in the system design, such as weak cryptographic algorithms for protecting the user plain data [47, 48] and problematic security-critical procedures [31, 32, 34, 38]. Different from these studies, our research aims at understanding the security risks in the 3GPP documentation and the specification design process through analyzing CRs and their management process.

SoK-style papers. There are some prior efforts to survey or systemize the knowledge of key security issues in the cellular network system. Bertino et al. [14] summarize emerging systematic methods for analyzing cellular network security and discussed their limitations. Based on their discovery, the authors further propose an initial security and privacy roadmap for 5G [15]. Rupprecht et al. [46] systematically categorize well-known attacks and defenses from 2G to 5G and map the attacks to proposed defense mechanisms and suggestions for 5G specifications, which helps identify open research questions and challenges for the development of the next-generation cellular network. Unlike these surveys and

SoK researches, we analyzed 3GPP CRs, the records of 3GPP standards' modifications, and discovered not only the weaknesses in the 3GPP specifications but also the risks in the 3GPP ecosystem, such as long-standing attack window.

Security analysis on standard documents. A previous study [55] analyzes security consideration sections (SCSs) and other security-related descriptions in RFCs. Although this work also touches the specification ecosystem (for RFC), it just looks for the mandatory requirement for SCSs in the RFC guideline, and if such section is already present in an RFC, measures the topics the section covers and other information. By comparison, we studied security-related CRs, which are unique for the 3GPP specification development. More importantly, we dived into security implications of these CRs and their management procedure, which leads to the discovery of the security risks in the 3GPP ecosystem never reported before.

7 Conclusion

In this paper, we developed a novel CREEK to recover 1,270 high-confidence SR-CRs. Our measurements on them revealed serious security consequences of specifications and their causes, including design errors and presentation issues, particularly the pervasiveness of misalignment in security-relevant content. Also important is the discovery of a security weakness inherent to the 3GPP ecosystem, which publishes an SR-CR long before the specification has been fixed and related systems have been patched. This opens an attack window, which can be as long as 11 years. Interestingly, we found that some recently reported vulnerabilities are actually related to the SR-CRs published years ago. And we identified 6 are still there today, including 1 weaknesses existing in major carriers. With the trend of SR-CRs not showing any sign of abating, we propose measures to improve the security of the 3GPP ecosystem.

8 Acknowledgment

We would like to thank the anonymous reviewers for their insightful comments, particularly our shepherd Yongdae Kim for the guidance for preparing the final version.

References

[1] 3GPP. https://www.3gpp.org/.

[2] Alliance for Telecommunications Industry Solutions. https://www.atis.org/.

[3] China Communications Standards Association. http://www.ccsa.org.cn/.

[4] Coordinated vulnerability disclosure (cvd). https://www.3gpp.org/specifications/coordinated-vulnerability-disclosure-cvd.

[5] CTIA. https://www.ctia.org/.

[6] Ericsson. https://www.ericsson.com.

[7] GSM Association. https://www.gsma.com/.

[8] Huawei Technologies. https://www.huawei.com.

[9] Qualcomm Technologies International. https://www.qualcomm.com.

[10] SCAT. https://github.com/fgsect/scat.

[11] srsRAN. https://github.com/srsran/srsRAN.

[12] 3GPP. Technical Specification Group working methods. https://www.etsi.org/deliver/etsi_tr/121900_121999/121900/.

[13] Andrew Liptak. Huawei announces its first 5G chip for mobile devices. https://www.theverge.com/2018/2/25/17050296/huawei-balong-5g01-chip-5g-networks-tech-mwc-2018.

[14] Elisa Bertino. It takes a village to secure cellular networks. *IEEE Security & Privacy*, 17(5):96–95, 2019.

[15] Elisa Bertino, Syed Rafiul Hussain, and Omar Chowdhury. 5g security and privacy: A research roadmap. *arXiv preprint arXiv:2003.13604*, 2020.

[16] 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 *2021 IEEE Symposium on Security and Privacy (SP)*, pages 1197–1214. IEEE, 2021.

[17] Florent Chiaroni, Mohamed-Cherif Rahal, Nicolas Hueber, and Frédéric Dufaux. Learning with a generative adversarial network from a positive unlabeled dataset for image classification. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 1368–1372. IEEE, 2018.

[18] Merlin Chlosta, David Rupprecht, and Thorsten Holz. On the challenges of automata reconstruction in lte networks. In *WiSec*, 2021.

[19] David Rupprecht. 5G Release-16 Without Mandatory Full-Rate Integrity Protection? https://davidrupprecht.github.io/nianullblog.

[20] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[21] Marthinus C Du Plessis, Gang Niu, and Masashi Sugiyama. Analysis of learning from positive and unlabeled data. *Advances in neural information processing systems*, 27, 2014.

[22] P Fisher et al. Negative binomial distribution. *Annals of Eugenics*, 11:182–787, 1941.

[23] Tieliang Gong, Guangtao Wang, Jieping Ye, Zongben Xu, and Ming Lin. Margin based pu learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[24] Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. Don't stop pretraining: adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964*, 2020.

[25] Dan Hendrycks and Kevin Gimpel. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. 2016.

[26] Geoffrey Hinton, Nitsh Srivastava, and Kevin Swersky. Neural networks for machine learning. *Coursera, video lectures*, 264(1):2146–2153, 2012.

[27] Byeongdo Hong, Sangwook Bae, and Yongdae Kim. Guti reallocation demystified: Cellular location tracking with changing temporary identifier. In *NDSS*, 2018.

[28] Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*, 2011.

[29] Wenpeng Hu, Ran Le, Bing Liu, Feng Ji, Jinwen Ma, Dongyan Zhao, and Rui Yan. Predictive adversarial learning from positive and unlabeled data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7806–7814, 2021.

[30] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015.

[31] Syed Hussain, Omar Chowdhury, Shagufta Mehnaz, and Elisa Bertino. Lteinspector: A systematic approach for adversarial testing of 4g lte. In *Network and Distributed Systems Security (NDSS) Symposium 2018*, 2018.

[32] 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 *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 669–684, 2019.

[33] Shantanu Jain, Martha White, and Predrag Radivojac. Recovering true classifier performance in positive-unlabeled learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[34] Imtiaz Karim, Syed Rafiul Hussain, and Elisa Bertino. Prochecker: An automated security and privacy analysis framework for 4g lte protocol implementations. In *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS)*, pages 773–785. IEEE, 2021.

[35] Eunsoo Kim, Dongkwan Kim, CheolJun Park, Insu Yun, and Yongdae Kim. Basespec: Comparative analysis of baseband software and cellular specifications for l3 protocols. In *Symposium on Network and Distributed System Security (NDSS)(San Diego, CA, USA). ISOC*, 2021.

[36] Hongil Kim, Jiho Lee, Eunkyu Lee, and Yongdae Kim. Touching the untouchables: Dynamic security analysis of the lte control plane. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 1153–1168. IEEE, 2019.

[37] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[38] Denis Foo Kune, John Koelndorfer, Nicholas Hopper, and Yongdae Kim. Location leaks on the gsm air interface. *ISOC NDSS (Feb 2012)*, 2012.

[39] C Linton. Freeman, elementary applied statistics. *New York: John C. Wiley and Sons, Inc*, pages 71–78, 1965.

[40] 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. *arXiv preprint arXiv:1907.11692*, 2019.

[41] Norbert Ludant and Guevara Noubir. Sigunder: a stealthy 5g low power attack and defenses. In *Proceedings of the 14th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pages 250–260, 2021.

[42] Subhabrata Mukherjee and Ahmed Awadallah. Uncertainty-aware self-training for few-shot text classification. *Advances in Neural Information Processing Systems*, 33:21199–21212, 2020.

[43] C Park, Sangwook Bae, B Oh, Jiho Lee, Eunkyu Lee, Insu Yun, and Yongdae Kim. Doltest: In-depth downlink negative testing framework for lte devices. In *USENIX Security Symposium*, 2022.

[44] William H Press and Glennys R Farrar. Recursive stratified sampling for multidimensional monte carlo integration. *Computers in Physics*, 4(2):190–195, 1990.

[45] Muhammad Taqi Raza and Songwu Lu. A systematic way to lte testing. In *The 25th Annual International Conference on Mobile Computing and Networking*, MobiCom '19, 2019.

[46] David Rupprecht, Adrian Dabrowski, Thorsten Holz, Edgar Weippl, and Christina Pöpper. On security research towards future mobile network generations. *IEEE Communications Surveys & Tutorials*, 20(3):2518–2542, 2018.

[47] David Rupprecht, Katharina Kohls, Thorsten Holz, and Christina Pöpper. Call me maybe: Eavesdropping encrypted {LTE} calls with revolte. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 73–88, 2020.

[48] David Rupprecht, Katharina Kohls, Thorsten Holz, and Christina Pöpper. Imp4gt: Impersonation attacks in 4g networks. In *NDSS*, 2020.

[49] Sean Hollister. Qualcomm announces the Snapdragon 855 processor for 5G phones. https://www.theverge.com/2018/12/4/18125853/qualcomm-snapdragon-855-mobile-processor-announcement.

[50] Altaf Shaik, Ravishankar Borgaonkar, N Asokan, Valtteri Niemi, and Jean-Pierre Seifert. Practical attacks against privacy and availability in 4g/lte mobile communication systems. 2016.

[51] Claude Elwood Shannon. A mathematical theory of communication. *ACM SIGMOBILE mobile computing and communications review*, 5(1):3–55, 2001.

[52] Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2):227–244, 2000.

[53] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[54] International Telecommunication Union. ITU-T Recommendation I.130: Method for the characterization of telecommunication services supported by an ISDN and network capabilities of an ISDN. 1989.

[55] Justin Whitaker, Sathvik Prasad, Bradley Reaves, and William Enck. Thou shalt discuss security: Quantifying the impacts of instructions to rfc authors. In *Proceedings of the 5th ACM Workshop on Security Standardisation Research Workshop*, pages 57–68, 2019.

[56] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45, 2020.

[57] 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 *28th USENIX Security Symposium (USENIX Security 19)*, pages 55–72, 2019.

[58] Chuan Yu and Shuhui Chen. On effects of mobility management signalling based dos attacks against lte terminals. In *2019 IEEE 38th International Performance Computing and Communications Conference (IPCCC)*, pages 1–8. IEEE, 2019.

[59] Chuang Zhang, Dexin Ren, Tongliang Liu, Jian Yang, and Chen Gong. Positive and unlabeled learning with label disambiguation. In *IJCAI*, pages 4250–4256, 2019.

[60] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27, 2015.

APPENDIX

A Implementation Details

A.1 Fine-tuning BERT

We use 3GPP specifications for BERT fine-tuning. We transformed these documents into "docx" format by LibreOffice software and stored in plain text ("txt" format) through python-docx library. Totally, we collected 1,318,364 sentences in 1546 specifications exclude menus, titles and failures during our processing.

For fine-tuning BERT on 3GPP corpus, we establish two objectives: *Masked Language Model (MLM)* and *Security Specification Classification (SSC)*. The MLM objective is to train our BERT to predict randomly masked words in a sentence. According to the experience reported in [40], we randomly select 15% words in each sentence and mask them by replacing with a special token [MASK]. We use the cross entropy loss for MLM objective. The SSC objective is to train our BERT to judge whether a given text belongs to security specifications. We use binary cross entropy function as the loss function for SSC.

Table 7: Summary of SR-CRs attack windows in commercial system

Note1: The more detailed vulnerability information can be found in our website. Note2: To protect the basebands and network operators, the three works [31, 36, 50] keep them anonymous. Time$_1$: specification update date for a SR-CR (yyyy/mm/dd). Time$_2$: the release date of a vulnerable device / the time the problems confirmed in our experiments / the year the related papers were published. Attack window: minimum estimated number of years.

id	CR	Time1	Time2	Attack Window	System	Reporter
1	C1-101068	2010/03/31	2019	8.7 years	Two commercial carrier networks	Kim et.al [36]
2	S3-091802	2009/12/18	2018	8.0 years	Four major U.S. network operators	Hussain et.al [31]
3	C1-101252	2010/03/31	2018	7.7 years	Four major U.S. network operators	Hussain et.al [31]
4	C1-135219	2013/12/20	2021	7.0 years	China Unicom	Chen et.al [16]
5	C1-141405	2014/06/27	2021	6.5 years	China Unicom	Chen et.al [16]
6	C1-094446	2009/12/17	2015/09/29	5.7 years	Google Nexus 6P	Chen et.al [16]
			2021/10/01	11.7 years	Google Nexus 6P	Our paper
7	C1-095712	2009/12/17	2015/09/10	5.7 years	iPhone 6sp	Yu et.al [58]
			2021/10/01	11.7 years	Samsung Galaxy S10, Google Pixel 3, Google Nexus 6P	Our paper
8	C1-112809	2011/09/28	2017/06/16	5.7 years	Honor 9	Yu et.al [58]
			2016/12/01	5.1 years	M5 Note	Yu et.al [58]
			2015/09/10	3.9 years	iPhone 6sp	Yu et.al [58]
			2021/10/01	10.0 years	Google Pixel 3	Our paper
9	R2-103442	2010/06/18	2016	5.5 years	four major basebands	Shaik et.al [50]
10	C1-132662	2013/06/27	2015/09/10	2.2 years	iPhone 6sp	Yu et.al [58]
			2021/10/01	8.2 years	Samsung Galaxy S10, Google Pixel 3, Google Nexus 6P	Our paper
11	C1-172658	2017/06/16	2019	1.5 years	two commercial carrier networks	Kim et.al [36]
12	C1-154699	2015/12/18	2017/06/16	1.4 years	Honor 9	Yu et.al [58]
			2016/12/01	0.9 years	M5 Note	Yu et.al [58]
13	C1-161448	2016/03/18	2017/06/16	1.2 years	Honor 9	Yu et.al [58]
			2016/12/01	0.7 years	M5 Note	Yu et.al [58]
			2021/10/01	5.5 years	Samsung Galaxy S10, Google Pixel 3	Our paper
14	S3-161217	2016/09/30	2021/10/01	5.0 years	Three major commercial carriers	Our paper

The positive training text of SSC comes from 39,608 sentences in 71 specifications. Referring [24], we extend each sentence to a predefined maximum text length (512 words) by the following sentences. The same extension is done on the negative text in the rest specifications. To balance the ratio of positive and negative training text, we randomly sample the same number of positive and negative samples in each minibatch.

We let our BERT learn the MLM objective prior to SSC objective by setting the weight of MLM objective as 1 and weight of SSC objective as $1e^{-3}$. Following [40], our BERT is optimized with Adam optimizer [37] using the following parameters: $\beta 1 = 0.9$, $\beta 2 = 0.999$, $eps = 1e^{-6}$ and L2 weight decay of 0.01. The learning rate is warmed up over the first 10,000 steps to a peak value of $1e^{-4}$, and then linearly decayed. BERT trains with a dropout of 0.1 on all layers and attention weights, and a GELU activation function [25]. Models are pretrained for 1,000,000 iterations, with minibatch of batch size 256 and input text with maximum length 512 words.

A.2 Self-training improvement

After training a classifier $\hat{C}$ through our PU learning algorithm, we leverage self-training on unlabeled data to strengthen the distinguishability of our classifier. Specifically, the self-training process runs in teacher-student iterations (Section 2.2). We want the student model learn to produce high confident prediction on those instances that teacher model has less doubt on. For measuring the prediction uncertainty of our model, we exploited BALD with the help of the dropout distribution of the model parameters [53].

Specifically, BALD selects the unlabeled samples $\mathbf{x_u}$ that maximize information gain between the current predictions and the posterior predictions of $\mathbf{x_u}$ after incorporating the labeled samples into the model:

$$\mathcal{B}(y_u, W|\mathbf{x_u}, D'_u) = \mathbf{H}[y_u|\mathbf{x_u}, D'_u] - \mathbf{E}_{p(W|D'_u)}[\mathbf{H}[y_u|\mathbf{x_u}, W]] \tag{10}$$

where $\mathbf{H}[y_u|\mathbf{x_u}, W]$ is the entropy of y_u given $\mathbf{x_u}$ under model parameters W, and D'_u is the pseudo-labeled unknown dataset. Based on dropout distribution, an approximation of Eq. 10 can be obtained by Monte-Carlo integration [44] as:

$$\begin{aligned}&\widehat{\mathcal{B}}(y_u, W|\mathbf{x_u}, D'_u)\\&= -\Sigma_c(\tfrac{1}{T}\Sigma_t \hat{p}^t_c)\log(\tfrac{1}{T}\Sigma_t \hat{p}^t_c) + \tfrac{1}{T}\Sigma_{t,c}\hat{p}^t_c \log(\hat{p}^t_c)\end{aligned} \tag{11}$$

where, $\hat{p}^t_c = p(g^{\widetilde{W}_t}(\mathbf{x_u}) = c)$, $\widetilde{W}_t \in \{\widetilde{W}_1, ..., \widetilde{W}_T\}$. $\hat{p}^t_c$ is the probability that a model $g^{\widetilde{W}_t}$ with dropout sampled weights $\widetilde{W}_t$ judges $\mathbf{x_u}$ belongs to c. Using this empirical approximation of BALD, we can select those less uncertain instance $\mathbf{x_u}$ with high $p^{easy}_c(\mathbf{x_u}) = \frac{1-\widehat{\mathcal{B}}(y_u, W|\mathbf{x_u}, D'_u)}{\Sigma_{\mathbf{x_u}\in\{\mathbf{x_u}:y_u=c\}} 1-\widehat{\mathcal{B}}(y_u, W|\mathbf{x_u}, D'_u)}$. After selecting a set of uncertain instances, we also use the predictive variance [42] $Var(y) = Var[(y|W, \mathbf{x_u})] + \mathbf{E}[Var(y|W, \mathbf{x_u})]$ to weight those selected instances, as Eq. 11 only considers the prediction mean $(\frac{1}{T}\Sigma_t \hat{p}^t_c)$). Gathering them all, we optimize the following objective on unlabeled instances:

$$\min_W \mathbf{E}_{x_u \sim p^{easy}_c(\mathbf{x_u})} \mathbf{E}_{y \sim g^{W^*}(\mathbf{x_u})} [\log p(y|g^W(\mathbf{x_u})) \cdot Var(y)] \tag{12}$$

where $g^{W^*}(\cdot)$ is the teacher model (optimal model in the last teacher-student iteration). Note that Eq. 12 equals to weighting the BALD selected instances with $\log(\frac{-1}{Var(y)})$. This weighting scheme emphasizes those instances with low variance results predicted by T dropout sampled models.