Preventing SIM Box Fraud Using Device Model Fingerprinting

Beomseok Oh*
KAIST
beomseoko@kaist.ac.kr

Junho Ahn*
KAIST
dwg226@kaist.ac.kr

Sangwook Bae
KAIST
hoops@kaist.ac.kr

Mincheol Son
KAIST
mcson@kaist.ac.kr

Yonghwa Lee
KAIST
yhlee0078@kaist.ac.kr

Minsuk Kang
KAIST
minsukk@kaist.ac.kr

Yongdae Kim
KAIST
yongdaek@kaist.ac.kr

***Abstract*—SIM boxes have been playing a critical role in the underground ecosystem of international-scale frauds that steal billions of dollars from individual victims and mobile network operators across the globe. Many mitigation schemes have been proposed for these frauds, mainly aiming to detect fraud call sessions; however, one direct approach to this problem—the prevention of the SIM box devices from network use—has not drawn much attention despite its highly anticipated benefit. This is exactly what we aim to achieve in this paper. We propose a simple access control logic that detects when unauthorized SIM boxes use cellular networks for communication. At the heart of our defense proposal is the precise fingerprinting of device models (*e.g.*, distinguishing an iPhone 13 from any other smartphone models on the market) and device types (*i.e.*, smartphones and IoT devices) without relying on international mobile equipment identity, which can be spoofed easily. We empirically show that fingerprints, which were constructed from network-layer auxiliary information with more than 31K features, are mostly distinct among 85 smartphones and thus can be used to prevent the vast majority of illegal SIM boxes from making unauthorized voice calls. Our proposal, as the very first practical, reliable unauthorized cellular device model detection scheme, greatly simplifies the mitigation against SIM box frauds.**

I. Introduction

Together with telecommunication fraud [25], [31], [48], bypass fraud has plagued the global mobile networks in the last two decades, causing significant financial damage to mobile network operators (MNOs) [47], [49]. *SIM boxes* are at the center of these bypass frauds as they interconnect mobile cellular networks with low-cost voice-over-IP (VoIP) sessions in ways that MNOs would not typically approve.

Most notably, interconnect bypass frauds use SIM boxes deployed in a country in which a callee is located to bypass international call billing while utilizing low-cost VoIP connections to the callee. The primary reason that SIM boxes are useful for such attacks is that they provide a means of circumventing the billing system at the local MNOs.

*These two authors equally contributed.

Network and Distributed System Security (NDSS) Symposium 2023
27 February - 3 March 2023, San Diego, CA, USA
ISBN 1-891562-83-5
https://dx.doi.org/10.14722/ndss.2023.24416
www.ndss-symposium.org

Recently, we have witnessed a surge of an emerging type of criminal activities with a renewed interest in SIM boxes [42], [52], [53]. Organized crime groups have quickly learned that with SIM boxes, they can mount financial scamming operations targeting people in a certain nation while staying outside the legal boundaries of their targets. Scam calls made through SIM boxes show local phone numbers on the target's mobiles, rendering these calls much less suspicious than traditional VoIP-based scam calls. These financial scams typically have targets in developed countries (where the expected financial gain is large) while their operations take place in less regulated, attacker-chosen countries anywhere on Earth [9]. Over 256,000 phone frauds, worth approximately \$18.6 billion, occurred in China in 2020 [17], [30]. Korea also has suffered a significant loss of approximately \$0.6 billion due to scam calls [37]. For simplicity, we will refer to those frauds using SIM boxes as "*SIM box frauds*" throughout this paper.

Security researchers have proposed a number of interesting countermeasures that detect calls that traverse SIM boxes based on the call data [33], [39] and voice call quality [13], [45]. Specifically, the countermeasures first accept all (including scam) calls to monitor them and use additional information obtained from the session. Some [13], [45] utilize packet loss in the call, while others [33], [39] utilize call detail records (CDRs) for the detection. Note that these solutions mainly utilize the properties of the individual SIM or call session, so that detection is performed after the calls are made.

Meanwhile, one basic approach that has not drawn much attention is to detect the SIM box (the device itself) in the network layer when it accesses the cellular network. For this, a naïve approach is to use the International Mobile Equipment Identity (IMEI), a primary device identifier that has been designed and used for device model identification for decades. Specifically, the cellular network can check the device model based on the reported IMEI and reject its access if the device is a SIM box. However, it is well-known that IMEIs are easily spoofed with little to no effort (*e.g.*, many SIM boxes provide IMEI spoofing as a feature [22]).

Therefore, instead of solely using this untrustworthy IMEI, we propose to utilize the information about the device capabilities from the contents of *control-plane* messages exchanged at the time of initial attachment procedures to identify the device model. More precisely, we utilize the two control-plane messages from RRC (`UECapabilityInformation`) and NAS (`ATTACH Request`). Both messages report the device capabilities, related to the setup of the radio connection and service establishment, to the base station and core network. Note that those capabilities are closely related to the hardware support of the device itself and the software support of the embedded baseband; thus, the capability information of a device can serve as its fingerprint. We show that individual smartphone models are accurately *fingerprintable*.

To this end, we present a systematic way of constructing a device model fingerprint. Instead of utilizing the content of control-plane messages blindly, our fingerprint construction method starts from the specification analysis. *Using the specifications* as a guide, we build a filter containing user-specific, session-specific, or previous-connection-specific fields, which cannot represent the characteristics of a device itself. This procedure may seem inefficient as it requires manual effort; however, it is only *a one-time task that occurs at specification updates*. Notably, as the typical time intervals between specification updates are a few months, this manual task is not required frequently. With the filter resulting from the manual analysis, the device fingerprint is constructed by *automatically pruning out* the features in that filter. Note that *automatic comparative analysis* is used to double-check the correctness of our filter, which helps to reflect non-device specific features missed by manual analysis. As the remaining procedures are designed to be fully automated, our fingerprint construction method can easily be scaled out.

We believe that we are the first to present empirical evidence that smartphones are indeed *fingerprintable* when systematically pruning and utilizing the large feature space (*e.g.*, 31,118 features as of July 2022) in the control-plane messages. Our large-scale experiment results show that most smartphone fingerprints are unique. Our dataset contains 279 traces of 102 different mobile devices with various options, including 85 distinct smartphone models, 6 LTE-compatible SIM boxes, and 11 other IoT devices on the market. Our in-depth analysis shows that smartphone manufacturers and baseband vendors independently and jointly produce a variety of unique device capabilities and configurations, resulting in distinct fingerprints for smartphones (§V).

Based on the effectiveness of the designed fingerprints, we present a fingerprint-based access-control mechanism for preventing SIM box fraud that can be deployed in today's cellular networks. We make specific recommendations for MNOs' local policies for IoT devices, including SIM boxes. The suggested access control list (ACL) successfully prevents the majority of SIM box fraud use cases (§VI). In addition, we present practical deployment considerations of the proposed SIM box fraud prevention system for commercial networks and analyze the system overhead for each fingerprint construction step. We demonstrate that the steps can be mostly automated and that manual procedures do not have much overhead (§VII).

We conclude that, *without the supplier's support or using the same chipset of smartphones*, mimicking the control-plane messages to smartphones has limitations for fraudsters. Impersonation of IoT devices may be still possible due to their simpler capabilities, yet the proposed system makes such attacks impractical (§VIII). We will open our entire dataset/codes to the public [1].

Contributions. In summary, our contributions are as follows:

- We present the first empirical study of cellular device model fingerprinting with a large-scale dataset.
- We conduct concrete feature analysis and consider user-configurable options. Through feature analysis, features that are user-specific, session-specific, or related to previous connections are eliminated as these interfere with fingerprinting. In addition, we demonstrate that device model fingerprinting would fail if user configurations were not considered.
- We suggest a SIM box fraud prevention system that utilizes fingerprint-based access policies. This system is the first to use control-plane messages for prevention.
- We analyze (a) the overhead of the proposed system in the various scenarios and (b) the required changes for system deployment on commercial networks.

II. Cellular Networks and SIM Boxes

We first offer a short overview of how MNOs in cellular networks manage heterogeneous devices (*e.g.*, smartphones, wearables, IoTs, and SIM boxes) (§II-A). We then outline what SIM boxes do in cellular networks and in what use cases they are employed in practice (§II-B).

A. Device Managements in Cellular Networks

LTE architecture. The LTE network includes three components: user equipment (UE), an evolved Node B (eNB), and an evolved packet core (EPC). The UE is an end device that provides data and voice services to an end user. The eNB is a base station that provides a wireless connection to UEs using the radio resource control (RRC) protocol [6]. The EPC is the core of the network that contains a mobility management entity (MME) for user authentication and session/identity management. The MME communicates with the UE via the non-access-stratum (NAS) protocol [5] to perform these management functions. The MME is responsible for authenticating users and managing the keys, sessions, and identities. On the UE side, the baseband (also called baseband modem) processes and manages the NAS and RRC control-plane protocols.

International Mobile Equipment Identity (IMEI). Mobile devices in cellular networks are assigned multiple identifiers at different layers. One of them is the IMEI, an identity specifying the device handset itself [29]. This 15-digit identifier provides a globally unique ID for each device handset. In particular, the first 8 digits of an IMEI, called a type allocation code (TAC), indicate the precise model of an individual device bearing it.

The TAC system in IMEI is crucial for a range of business and operational applications in the mobile ecosystem, such as marketing (for customizing offers and upgrades to different device models), procurement (for understanding the relationship between device models and revenue), service differentiation (for providing different quality of services to different types of devices), and network planning (for optimizing network performance and efficiency) [29].

Device identification procedure. The core network requests IMEI of the UE in two ways. The first is to utilize `NAS Identity Request and Response` messages. When the core network sends a `NAS Identity Request` message requesting an IMEI, the UE sends its IMEI in response. Another way is to utilize `NAS Security Mode Command/Complete`. When the core network sends `NAS Security Mode Command` message, it can optionally request the IMEI-Software Version. Once requested, the UE sends a `NAS Security Mode Complete` message with its IMEI.

Lack of authentication. Despite the importance of IMEI in cellular networks, one can forge the IMEI of UE and deceive the network. Because there is no mechanism to authenticate the reported IMEI on the network, one can change the IMEI of a commercial smartphone by using file system modification tools [21], [57]. SIM box manufacturers also typically provide a method for users to change various device information, including the IMEI. Thus, an adversary may easily bypass IMEI-based access control in the cellular network.

Reporting UE capabilities. When the UE tries to access the LTE network, it reports its capabilities through the two control-plane messages: `ATTACH Request` and `UECapabilityInformation`. `ATTACH Request` contains the functionalities that the device supports, including security algorithms, speech codecs, supporting networks, and telephony features. While the information in `ATTACH Request` is related to the core network functions, `UECapabilityInformation`, an RRC layer message, lets the eNB know radio-access related capabilities of the device. Such information includes the supporting radio frequency, carrier aggregation (CA) configuration, and radio resource scheduling capabilities of the device. Those capabilities are highly related to the hardware support of the device and the software support of the embedded baseband. Thus, if the reported information does not match with the device, the communication between UE and network might be failed.

B. SIM Box and Illegal Use Cases

What is a SIM box? A SIM box is a VoIP gateway device that contains a number of SIM cards, which are used to connect to local cellular networks. As shown in Fig. 1, multiple (eight slots in red rectangles) SIM cards of different MNOs can be inserted into a SIM box. Large SIM boxes hold more than hundreds of SIM cards in one device. The sliver plates (yellow rectangle) are baseband chipsets, each connected to one SIM slot and one antenna port (orange rectangle). Note that each group acts as a UE and connects to cellular networks.

How does a SIM box work? A SIM box serves as a VoIP gateway, transferring VoIP calls to cellular calls. When an adversary tries to make a call to a victim using a SIM box, the VoIP session is first established between the victim and the SIM box. Then, it makes a mobile network call (*e.g.*, VoLTE, 3G call) to the victim using its embedded USIM. Thus, adversaries can make cellular voice calls far away from an installed SIM box as long as the SIM box and adversaries are connected through the internet. Consequently, adversaries with SIM boxes can use local phone numbers even when they are outside of the mobile network service area.

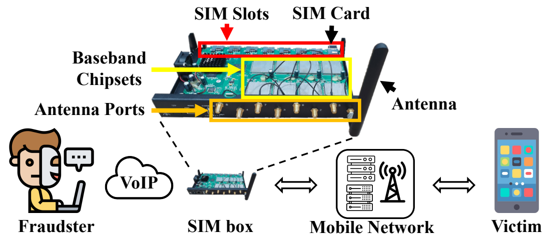

Fig. 1: SIM box structure and SIM box fraud procedure.

Interconnect bypass fraud. Typical international calls are routed through the regulated interconnection between callers and callees. In interconnect bypass frauds, international calls bypass these regulated interconnect (thus skipping billing for overseas calls) and route through the low-cost IP voice calls to the destination network. According to a recent survey [18], interconnect bypass frauds using SIM boxes constitute the fourth largest fraud type in the telecommunication business, with the loss increasing from $2.7 billion in 2019 to $3.1 billion in 2021.

Voice scam fraud. SIM boxes are also used to carry out voice scams or phishing attacks. By impersonating someone else, such as a close family member, colleague, or law enforcement officer, perpetrators mount financial scams. Perpetrators are usually organized crime groups and the callers are typically stationed outside the legal boundaries in which the victims are located. The main benefit of using SIM boxes for such scams is that the scammers can present local phone numbers (not suspicious VoIP numbers) on the victims' caller ID screen. Using local phone numbers is expected to increase the call success rates of the scammers. For instance, a bank manager in Hong Kong in 2020 transferred $35 million to a voice scammer who impersonated his boss [15]. If the call had reached the victim with a traditional VoIP caller ID, the bank manager would have had a suspicion and avoided the scam.

III. Threat Model and Defense Objectives

Threat model. We consider SIM box frauds, including both interconnect bypass frauds and voice scam fraud. The primary goal of these SIM box frauds is to make successful voice calls through local MNOs by operating their SIM boxes. We assume that adversaries have legitimate means of accessing commercial MNOs' networks with lawfully issued local SIMs. We do not consider supply-chain attackers that directly or indirectly influence device manufacturers or baseband vendors. Thus, our adversaries cannot make arbitrary changes to the device and baseband implementations or the messages generated by these systems.

Defense objectives. Our goal is to prevent unauthorized SIM boxes from making calls on cellular networks. For strong prevention of such an unauthorized call access, we propose an access control policy for MNOs to deploy and enforce for any device attached to them. We aim to present a precise and delicate access control logic for unauthorized SIM boxes so that MNOs block illegal use cases of SIM boxed calls but not the legal case; *e.g.* telemarketing. With our access policy, SIM boxes for lawful applications should be pre-registered (*e.g.*, self-registration [16]) at each MNO so that they are permitted to have voice call access.

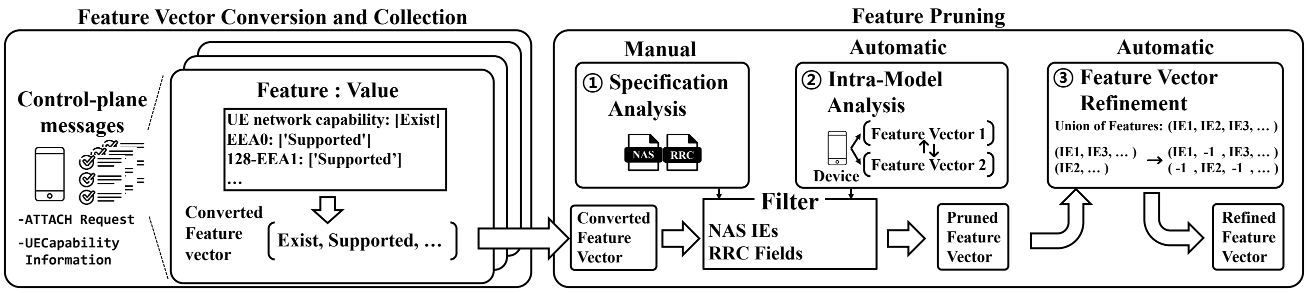

Fig. 2: Overview of fingerprint construction.

Our SIM box unit detection scheme is *orthogonal* to existing SIM boxed call detection schemes [10], [11], [13], [23], [32], [33], [38], [39], [45], [50] and they can be used together to complement each other. For example, our scheme can potentially stop many fraud calls made by SIM boxes even before they are detected and handled by other defense schemes.

Key insights. Strict access control at the entry point (*i.e.*, UE attachment) of a cellular network has been considered difficult because the de facto standard device model identification, IMEI, is known to be easily spoofable. Thus, any security policy based on IMEIs risks missing malicious access to the system and, worse yet, may give a false sense of security to operators, regulators, and end users.

To address this issue and design reliable and practical access control for cellular networks, we propose to utilize the auxiliary information found in the control-plane messages when any mobile device attaches to the network. One critical property we desire to leverage is that the auxiliary information consists of a wide variety of features depending on each device model (*e.g.* Galaxy S20 Ultra or iPhone 13). Through our large-scale empirical study, we confirm that the 102 different device models tested are indeed *fingerprintable* with the information.

IV. Constructing Device Model Fingerprints

One critical part is to construct the fingerprints that reflect the differences between device models from control-plane messages. When device models are uniquely fingerprinted, we can use these fingerprints to identify device models instead of IMEIs, which are easily spoofable. More specifically, our goal is to identify a device model, such as Galaxy S4 or iPhone 13 by using the fingerprint.

Fig. 2 shows an overall flow of constructing a device model fingerprint. First, we collect two control-plane messages from the device and convert them into a key-value list (§IV-B). Next, we collect feature vectors in the same manner while varying the device configurations (§IV-C). Then, we conduct a pruning process using the filter mainly built by manual specification analysis and automatic intra-model analysis (§IV-D). Note that the specification analysis is manual, yet a one-time task for each update. Lastly, feature vectors are automatically refined (§IV-D). Finally, we define the set of refined feature vectors as a fingerprint.

A. Preliminaries: From Standards to Control-plane Messages

While every device on the market must strictly adhere to the 3GPP standards, UEs have diversities in their implementations according to the device and baseband manufactur-

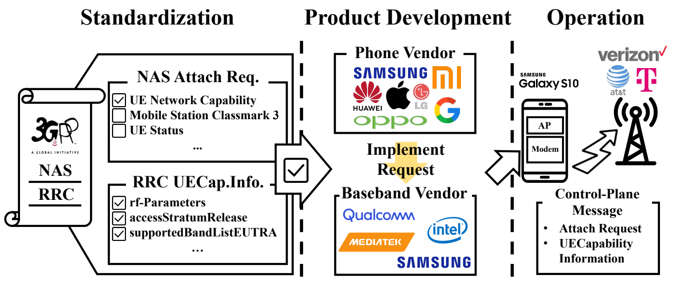

Fig. 3: Illustration of how cellular functionalities are defined, selected, and observed in modern cellular networks.

ers. These differences exist because implementation of many optional functionalities are left to the device manufacturers, allowing highly diverse sets of optional functionalities in the device implementations. In addition, each device has different capabilities due to the differences in hardware specifications and the specific requirements of each device manufacturer. Consequently, for each baseband vendor or UE, device capabilities such as supported radio frequencies and available radio technologies could be implemented differently. Also, the differences across different device models could be reflected in the control-plane messages that contain devices' capabilities. These messages are sent to the core and base station, as both require devices' capabilities to provide cellular services properly. Therefore, one can obtain such information from the control-plane messages. Fig. 3 shows the overall flow from standards to control-plane messages.

B. Converting Capability Information into Feature Vectors

We utilize capability information from the control-plane messages that mobile devices transmit to the cellular network during service registration called the `ATTACH` procedure. To be specific, inspired by the prior works [34], [51], we utilize two messages in NAS and RRC protocols: `ATTACH Request` and `UECapabilityInformation`. UE reports its capability information by using those control-plane messages.

Converting contents of messages. We first convert the contents of the control-plane messages into a key-value (Feature: Value) representation, as exemplified in Fig. 2. In the message conversion, we leverage the fact that the control-plane messages follow a specific format defined by standards [5], [6]. First, the `ATTACH Request` message follows a format defined by the NAS specification [5]. Using this format, we set the name of the information element (IE) as a key and its data as a value. Similarly, the `UECapabilityInformation` message is in ASN.1 format [6], and we set the field name as a key and

TABLE I: Considered configurations in feature collection.

	Configurations	Options
Setting option	Preferred network	5G-SA, 5G-NSA, or LTE
Engineering mode	Band selection	Automatic / LTE-only / Band (1, 3, 7)
	Service domain	CS/PS, PS only

the corresponding data as a value. As this conversion directly uses the format of each message, some fields and IEs may not be assigned values. For those cases, we set a value as "Exist". Note that the whole message is converted into a key-value list, *i.e.*, no IEs and fields are dropped during the conversion.

Feature and Feature Vector. In this work, we define a *feature* as a key in the converted representation. We also define a feature vector as a vector of values in the key-value list, resulting from the feature vector conversion above. As shown in Fig. 2, `UENetworkCapability` and `EEA0` are the features with corresponding values of `Exist` and `Supported`, respectively. Also, the corresponding *feature vector* is (`Exist`, `Supported`, $\cdots$). Note that some features have the same names as other features in the feature vector, because both RRC and NAS message formats allow repetition in a nested structure. For example, a `BandInfoEUTRA` field in a RRC specification [6] provides information about the supporting band. The field occurs multiple times in `UECapabilityInformation`, according to the number of supporting radio bands.

C. Feature Vector Collection with Various Options

Existing works [34], [51] suggest collecting control-plane features of different device models only with *default options*. We show that, however, the feature vectors belonging to the same device model may differ by their options. In other words, the feature vector of the same model may vary once end users customize their mobile devices with different settings. In practice, one can configure a device to use a specific radio band or the preferred network. Our experiments also show that more than half of the tested devices changed their feature vectors due to configuration changes (§V-B). Thus, fingerprinting while ignoring the effect of end-user configurations yields poor performance, and we cannot possibly call the feature sets fingerprints of a device model.

To address this issue, we conduct the feature vector conversion for each device while varying its configurations. Among various optional configurations, we focus on the network-related configurations that an end user can modify, because most features in the used control-plane messages are related to the network capabilities. In general, end-users can customize device configurations in two ways: the *settings tab* and *engineering mode*. First, one can select the preferred radio technology (*e.g.*, 5G-SA, 5G-NSA, or LTE) through the settings tab in both the iOS and Android mobile operating systems. Second, tech-savvy users may change the configurations of their devices through engineering mode, which provides various additional configurations. Among them, we choose network-related configurations, as we focus on the control-plane messages containing network capabilities. We covered most of the configurations that can be modified using the settings tab or engineering mode. Tab. I shows the configurations we covered, including network mode setting, band selection, and service domain selection. We did not cover the tiny number of configurations that might disturb the network connection.

TABLE II: Examples of fields/IEs removed by spec. analysis.

Properties	Examples	
User Specific	EPS mobile identity	TMSI based NRI container
Session Specific	EPS attach type	ESM message container
Previous Connection	Last visited registered TAI	Old location area identification

D. Systematic Pruning of Features

After converting the feature vectors from the messages collected from all test devices with available options, we conduct a systematic pruning of these features. This aspect separates our work the most from the existing cellular device model fingerprinting works [34], [51], in which researchers with domain knowledge provide anecdotal evidence with small-scale datasets and handpicked-feature vectors. We build a semi-automated pruning process for the large feature space in a systematic manner. Systematic feature vector pruning is critical for two reasons:

1) Not all available features in the large feature space are *device model specific*. For example, some features are too fine-grained as they carry ephemeral session information (*e.g.*, key identifiers, sequence number, transaction identifier). Some are permanent but carry individual user-specific information (*e.g.*, IMSI, TMSI). Some are also too coarse-grained (*e.g.*, all devices share the same features).
2) The features that are specific to device models (thus, useful for our fingerprinting) *change over time*, and thus, the pruning should be repeated frequently to adapt to the changes. Therefore, without a systematic approach, considerable manual effort is required on characterizing and pruning procedures whenever new devices are introduced to the market.

Hence, we present a simple yet highly effective approach for pruning the features for device model fingerprinting, which can ultimately be used for SIM box fraud prevention in real-world cellular networks. The key ideas are 1) to build a filter, a list of features that are not device model- and configuration-specific, and 2) to construct a refined feature vector by removing the features listed in the filter. In particular, we take two approaches to build the filter: *manual* inspection of the specification and *automated* comparative analysis of the collected feature vectors in the previous procedures (§IV-B and §IV-C).

Stage-①: Specification analysis. This stage aims to find out the features whose definitions in the specifications are not device model specific. Thus, those features should not be used to construct device model fingerprints. For this, we conduct a manual inspection on the two specification documents [5], [6]. Due to the enormous volume of the specifications, we set rules for this manual analysis stage. First, we mainly focus on the IEs and fields composing the target messages (*i.e.*, `ATTACH Request` and `UECapabilityInformation`). This part occupies 162 pages from four standards in total [3], [4], [5], [6]. Second, we only select the features that meet one of the following three properties; features are (1) individual user-specific (*e.g.*, identifiers such as IMSI or TMSI), (2) session-specific (*e.g.*, MAC, sequence number), or (3) associated with previous connections (*e.g.*, type of security context). Tab. II shows the examples of fields and IEs in the filter, which are found by specification analysis.

Consequently, the filter has 31 features consisting of 22 IEs and 9 fields in the collected control-plane messages. Although the number of features in the filter seems small, it greatly helps to prune a large number of features in the feature vector. Note that some IEs and fields contain other IEs and fields due to the nested structure of the messages. Also, some IEs and fields are repeated in the control-plane messages. Therefore, the elimination of such IEs and fields results in pruning the multiple features in the feature vector.

Stage-②: Intra-model analysis. Next stage aims to find out the features whose values are not consistent *within* a device model with the same configuration. These inconsistent features should not be used to construct device model fingerprints. Also, those features are difficult to determine with specification analysis alone; so empirical analysis is required. For this, we conduct a comparative analysis on the same device models.

We conduct the conversion procedure multiple times for the same device model with the same configuration. In addition, we collect the control messages in two settings: 1) attaching the device with different bands and 2) using different SIM card (but with the same public land mobile network (PLMN)). Note that all other configurations of test devices are fixed. With the converted feature vectors, we conduct a comparative analysis to determine which features should be added to the filter.

As a result, we found that some nested features are mixed in order even in the messages from the same device, resulting in different feature vectors. For example, we observe that the nested features in `supportedBandListEUTRA`, containing supporting radio bands information, show differences. Specifically, the values of the nested features are identical themselves, but they are listed in a different order, resulting in distinct feature vectors. Also, the features nested within `supportedBandCombination` are listed differently over the same device, resulting in different feature vectors. To consider this issue, we exclude those nested features.

This stage completes the feature pruning procedure along with the first stage. First, this empirical analysis helps reveal missed features that should not be used for the fingerprint in the first stage. For example, the abovementioned findings are difficult to acquire only by inspecting the specifications. Furthermore, considering that there exist way *too many features* in the modern 4G/5G cellular network control-plane messages, human errors could occur in the manual analysis of the specifications. Second, among many such features, which of them are indeed useful for device model fingerprinting does not always well align with our *domain knowledge*, rendering the pruning process difficult (not only tedious) even to the domain experts. For example, some values of features do not follow the 3GPP specifications due to differences in implementation. As a result of both stages, we were able to effectively remove unneeded features for generating device model fingerprints and greatly reduce the total 31,118 features to 922 features.

Stage-③: Feature vector refinement. Lastly, we formalize the feature vector after the pruning procedure. As we directly construct the feature vector from the collected control-plane messages, the length of feature vectors might vary over the devices. Note that the features in the control-plane messages vary according to the device implementation. Therefore, to construct a fingerprint, we refine the feature vector using the union of all features. Consequently, all feature vectors are represented with the same length and order of the features, as shown in the end of Fig. 2.

Constructing device model fingerprints. Finally, we define the term *fingerprint* as a set of multiple refined feature vectors from a device with various configuration options. In addition, we consider a device to be *fingerprintable*, or *has a unique fingerprint* if its fingerprint does not overlap with those of other devices. In other words, fingerprintable devices do not share a common feature vector with others.

V. Empirical Study on Device Model Fingerprints

We first evaluate whether the constructed fingerprint can be used to represent device models and prevent SIM boxes from connecting to the network. For this, we create *fingerprints* of various cellular device models and answer the following questions:

- **[Q1]** Do smartphones have unique fingerprints?
- **[Q2]** What makes smartphones fingerprintable?
- **[Q3]** Are SIM boxes fingerprintable?

Through large-scale experiments, we make a number of important observations, which are summarized as follows:

- **[O1]** Smartphones have *practically unique fingerprints*; thus their fingerprints can be practically used to present device models (§V-B, §V-C, §V-D).
- **[O2]** Smartphones are *unambiguously separated* from other device types by their fingerprints (§V-E).
- **[O3]** Device models other than smartphones also have highly distinguishing control-plane features but they may not be unique (§V-E).

A. *Experimental Setup*

Test devices. For large-scale analysis, we use 102 commercial cellular devices in total. Our test devices include smartphones from 5 baseband vendors (*i.e.*, Qualcomm, Samsung, HiSilicon, MediaTek, and Intel) and 8 major phone vendors (*i.e.*, Samsung, Apple, Huawei, Xiaomi, LG, Google, ZTE, and Oppo). We also test 6 SIM boxes from 5 different vendors. All SIM boxes work in LTE networks, and some even support Voice over LTE (VoLTE) as well. Note that LTE-compatible SIM boxes have been introduced to the market recently. They house 4 different baseband chipsets, and all of the modules are provided by one vendor, Quectel. Note that all the LTE-compatible SIM boxes from various vendors use the baseband module from Quectel, according to a recent survey [35]. We also test *IoT* devices, such as LTE routers, wearable devices (*e.g.*, Galaxy Gear), tablets (*e.g.*, iPad), and USB dongles. In our work, we use the term “IoT device” to refer to a cellular device that is not a smartphone. Please refer to Tab. XIII in the Appendix for the complete list of test devices.

Datasets. We construct 8 datasets according to the configuration and device type (Tab. III). Each dataset contains the feature vectors of the device converted from the control-plane messages. The numbers in parentheses in Tab. III are the total number of the feature vectors in each dataset. For a baseline evaluation, we constructed the P dataset using smartphones configured with the default options. To see the effect of

TABLE III: Dataset summary.

Dataset	# of devices (log)	Description	Usage
P	85 (85)	Phones set to default options	All
$P_{network}$	35 (39)†	Preferred network is configured	§V-B
P_{band}	16 (48)*	Force to use a specific radio band (1/3/7)	§V-B
$P_{service}$	16 (16)	Force to use service domain as PS only	§V-B
P_{update} ◇	71 (71)	Phones with each firmware updated	§V-B
S	6 (9)‡	SIM boxes force to use LTE and Auto	§V-E
I	11 (11)	IoT devices	§V-E
T	30 (30)	Test set of phones for evaluation	§V-F

† 35 devices are configured to use LTE network only. Additionally, we could configure 6 different network mode configuration for one device supporting 5G-SA/NSA network.
∗ We configure 3 kinds of radio bands for each of 16 devices allowing band selection.
‡ We could configure the 3 SIM boxes to use both LTE only/Auto, while 2 SIM boxes only used Auto and 1 only used LTE only.
◇ All phones have the latest firmware version as of Feb. 2022

TABLE IV: Cohort analysis.

Duplicated Devices		Differences
Galaxy S9 (B)	Galaxy S9+ (B)	Display, Battery, Camera
Xiaomi MI8	Xiaomi MIMIX2S	Released within 2 month
Galaxy S20†	Galaxy Note20 ultra†	Released within 6 month
Galaxy Note 9*	Galaxy S9+ (B)*	Released within 6 month
LG K50	LG X6*	Released simultaneously
Galaxy S10 (A)*	Galaxy S10e*	Released simultaneously
MI 5S*	MI5S+	Released simultaneously
iPhone12 Pro	iPhone12 mini*	Released within a month

† Both devices shows the same feature vectors only if when both are configured to use LTE network only or specific bands.
* The device with the updated firmware (dataset P_{update}). For instance, fingerprints of Galaxy Note9 and Galaxy S9+ B overlap when both are updated.

various non-default options to the fingerprinting, we construct datasets for four variants ($P_{network}$, P_{band}, $P_{service}$, P_{update}) by configuring smartphones with the three types of options described in Tab. I and by updating their firmware. Note that, we use subsets of all the devices listed in Tab. XIII to construct these datasets because some of the devices are restricted from using engineering mode. The devices composing each dataset are shown on our webpage [1].

Data collection. We build a testbed using srsLTE [28], an open-source LTE stack, to collect the control-plane messages. The EPC/eNB in our testbed is modified from the open-source LTE stack to transmit `UECapabilityEnquiry` for collecting `UECapabilityInformation` from the devices. The eNB works on top of a USRP B210 [56] as a software-defined radio (SDR). The devices are equipped with a programmable SIM card [54], whose PLMN is set to that of the local MNO.

B. Do smartphones have unique fingerprints?

With our large-scale datasets, we first investigate whether the smartphones are fingerprintable, by counting the number of distinct fingerprints (*i.e.*, feature vector) in dataset P. By comparing each feature vector with others, we observe that 83 fingerprints are distinctive among a total of 85 devices. In other words, 97.6% of the phones can be successfully identified.

This baseline-evaluation result seems unsatisfactory at first because some smartphones do *not* have unique fingerprints. After a brief examination of the exceptions, however, we conclude that all 85 smartphones are practically *fingerprintable* because all exceptions (pairs of smartphones with the same fingerprints) can be practically regarded as the same models for use of SIM box detection. The first two rows of Tab. IV show the exceptions from P. We find out that the smartphones in each pair are manufactured by the same vendor and also have the same baseband model. Furthermore, they are released to the market in a short period of time.

While they have minor differences in their hardware specifications (*e.g.*, displays, batteries), these differences are not related to the radio capabilities utilized for fingerprint construction. In this paper, we define two smartphones as "cohorts" if they have 1) the same baseband, 2) the same vendor, and 3) comparable release dates. Except for cohorts, the device models in P have unique fingerprints (*i.e.*, they are *fingerprintable*).

Why is it necessary to consider the device configuration? As outlined in §IV-C, understanding and utilizing the effect of end-user customization on a device is imperative for building fingerprints of smartphone models in practice. To examine its importance, we first analyze the difference between the fingerprints of devices with and without applied configurations. We utilize three datasets $P_{network}$, P_{band}, and $P_{service}$ to analyze each configuration option. Additionally, we consider the firmware of the device with a dataset P_{update}.

First, we investigate the number of devices whose feature vectors changed as a result of configuration changes. For all four datasets, we find that over half of the devices show different feature vectors compared with their default ones (Tab. V). Also, we investigate the number of features affected by configuration changes. Particularly, when the devices are set to use the LTE network only, 82 features changed in average, mainly because GSM/3G-related features no longer needed to be supported. In addition, firmware updates also resulted in changes to the feature vectors of 42 devices among the 71 devices in P_{update}. This is because the purpose of updates is to patch vulnerabilities or add new functionalities, which is reflected in the control-plane messages. For instance, a group of security researchers demonstrated that an algorithm GEA1 is insecure [14], prompting a standard [8] change in 2021 [2]. In this regard, 20 phones in P_{update} disabled the algorithm after the update. These results show that device model identification fails without understanding the effect of the user configuration; thus, end-user configurations must be taken into account when constructing fingerprints.

Second, we examine whether the fingerprints are still distinct even considering the configurations. For the evaluation, we first construct the fingerprint by adding all the feature vectors we collected ($P_{network}$, P_{band}, $P_{service}$, P_{update}) while varying the configurations from P. We then investigate the number of devices whose fingerprints have no intersections with others, similar to the baseline evaluation. As a result, we find that 77 of the 85 device fingerprints are different from the others. The fingerprints of six additional pairs of phones overlapped compared with the baseline evaluation; however, each pair is a cohort, as shown in Tab. IV.

In summary, most phones have distinct fingerprints, and this property is maintained even after expanding fingerprints to a set of feature vectors collected from devices with various configurations. Furthermore, all pairs of devices with overlapped fingerprints are cohorts (*i.e.* same baseband, same manufacturer, and similar release date).

TABLE V: Changes in features based on configuration.

Dataset	# of changed features	# of devices showing different feature vector	# of total devices
$P_{network}$	81.7	33	35
P_{band}	70.5	16	16
$P_{service}$	25.3	9	16
P_{update}	35.4	42	71

C. What makes smartphones fingerprintable?

We further investigate the fingerprints of smartphones to see how features contribute to achieving accurate device model fingerprinting. For this, we examine the effects of baseband vendors and phone vendors on fingerprints. Note that our large-scale dataset with 85 smartphones enabled us to conduct such detailed analysis and provide new insight on device model fingerprinting.

How do baseband vendors affect fingerprints? As the basebands play a key role in cellular communication (especially in the control-plane), the control-plane messages would be highly affected by the manufacturer. From our large-scale dataset, we find that baseband vendors influence fingerprints in two ways.

First, *each baseband vendor employs unique configurations for their baseband.* For example, they use their own battery-saving configuration. Specifically, each baseband vendor uses unique values for `SPLIT PG CYCLE CODE`; Qualcomm, Samsung, and HiSilicon basebands use values of 10, 16, and 32, respectively. Also, MediaTek and Intel basebands use value 8 (if that feature exists). Those configurations are adopted in smartphones consistently, regardless of the phone manufacturer or chipset number.

Second, *baseband vendors implement and support the functionality in the 3GPP specifications selectively.* The 3GPP specifications contain a number of communication technologies that are not mandatory. Due to the freedom in the implementation, the capabilities of each baseband vary, and these differences show up in its control-plane messages. We observed several examples through careful examination of our dataset. For example, the positioning technology [26] is supported in all basebands from Mediatek and some recent basebands from Qualcomm, while most basebands from other vendors do not support it. Likewise, only the devices equipped with the MediaTek basebands can support `TM5`, a modulation scheme-related capability, and Qualcomm basebands do not support a null integrity algorithm (EIA0).

How do phone vendors affect fingerprints? Vendors also affect fingerprints. When a phone vendor develops a phone that uses a given baseband, the manufacturer can choose whether or not to support certain capabilities of the device as needed. With the help of our large-scale dataset, we observe that *phone vendors can be distinguished even from devices having the same baseband chipset model.*

As a case study, we choose 6 devices equipped with two basebands (*i.e.*, Qualcomm Snapdragon 835 and 845) from three phone vendors (*i.e.*, Samsung, LG, and Xiaomi) and run a comparative analysis. Among the devices, only the LG devices did not support EEA3 and EIA3, the encryption and integrity algorithms. This observation implies that, while the baseband in those devices could support capabilities (because the devices having the same baseband from other phone vendors can support), the phone vendor may choose to support. By further investigation of 10 other devices listed in Tab. XIII with Qualcomm baseband that LG manufactured, we confirm that those algorithms are not supported except for one UE (*i.e.*, V50); V50 supports the latest protocol version of RRC (release 15). These studies indicate that phone vendors generate differences in the capabilities of their devices, resulting in distinct fingerprints.

D. Will the new device have a distinct fingerprint?

Despite our investigation of a large number of mobile devices, one may question the uniqueness of the fingerprint of a newly introduced device. However, the uniqueness of the fingerprint of a new device will be guaranteed for two intuitive and solid reasons. First, the 3GPP specifications are continually updated, and each update produces new features that can be part of fingerprints; thus, it contributes to the uniqueness of the fingerprints. Through specifications analysis, we find that a total of 73.3 fields and 21 IEs are added per update on average (Tab. IX). As new basebands have a high possibility of supporting those new features while old basebands do not, the fingerprints will be distinct. Second, a newly introduced device embeds improved hardware compared to the previous model, resulting in better capabilities, such as the support of more bands. Brand new devices usually embed the latest baseband and peripheral hardware, thus supporting brand new technology in the latest 3GPP specification (*i.e.*, 5G-NSA).

To confirm both intuitions, we conduct a comparative analysis of 7 and 6 phones from the Galaxy S series and the Apple iPhones, respectively. Note that all devices except the iPhone 8 and iPhone XS equip Qualcomm basebands.

Tab. VI shows the numbers of features that newly appeared, compared to the previous device. We find that next-generation devices always contain at least 3 (and up to 162) newly emerging features in both series, supporting the first intuition. For instance, for the first devices supporting release 15, 5G-related features (*e.g.*, 5G-EA0 and 5G-IA0) are first introduced among the series. Furthermore, even if two devices support the same specification version, the more recent device contains new features, because a more recent device has improved hardware compared to the previous one, as explained in the second intuition. For example, even though the Galaxy S7 (B) and S8 support the same specification version (rel. 11), the Galaxy S8 supports more bands, resulting in a difference in the related features compared to the Galaxy S7 (B). These findings imply that new phones will have unique fingerprints, unless a manufacturer sells the same phone (same hardware and software) under a different name.

E. Are SIM boxes fingerprintable?

So far, we have only evaluated the uniqueness of fingerprints on smartphones. Now, we discuss whether the fingerprint generated from the control-plane messages of SIM boxes is distinguishable from those of smartphones and other IoT devices.

Are phones distinguishable from IoT devices? We first address whether smartphone fingerprints remain unique from those of SIM boxes and other IoT devices. To answer this

TABLE VI: Number of new features along series of devices.

Galaxy phones	RRC release	# of new features	Example of new features
Galaxy S5 (A)	10	-	-
Galaxy S7 (B)	11	22	ProSe, rf-Parameters-v1130
Galaxy S8	11	45	rf-Parameters-v1180
Galaxy S9 (B)	12	3	pdcp-SN-Extension-r11
Galaxy S10 (B)	14	162	otdoa-UE-Assisted-r10
Galaxy S20	15	99	5G-EA0, 5G-IA0
Galaxy S22+	15	5	eutra-CGI-Reporting-ENDC-r15

Apple phones	RRC release	# of new features	Example of new features
iPhone 6	10	-	-
iPhone 7	11	17	rf-Parameters-v1130
iPhone 8	11	41	Handover between FDD and TDD
iPhone XS	12	19	rf-Parameters-v1310
iPhone 12 pro	15	124	5G-EA0, 5G-IA0
iPhone 13	15	5	mbms-Parameters-r11

question, we use seven datasets: five smartphone datasets and two datasets of SIM boxes and remaining IoT devices. In total, we use 279 feature vectors from 102 devices including 85 phones, 6 SIM boxes and 11 other IoT devices. We then count the overlapped fingerprints that share one or more feature vectors of IoTs and SIM boxes. As a result, we observe that no fingerprints of smartphones overlapped with any other type of device. This finding implies that *smartphones are unambiguously separated from other device types by their fingerprints*.

Especially, we further examine the effective features that highly contribute to classifying smartphones and SIM boxes. Although SIM boxes and smartphones provide the same services (*e.g.*, voice and data services), their capabilities show differences in some features. The differences appear due to the limitations of the SIM boxes in terms of hardware and software. First, in terms of hardware, typical SIM boxes only have one antenna per baseband, as shown in Fig. 1. This hardware characteristic limits the use of some functionalities requiring multiple antennas, such as carrier aggregation (CA), a radio technology utilizing multiple frequencies simultaneously. Thus, CA-related features do not appear on the control-plane messages of SIM boxes, whereas most smartphones do. In addition, as most IoT devices (including SIM boxes) embed low-cost basebands, their supporting protocol version is relatively low or configured to use low-performance operations (*e.g.*, `ue-Category` is set to a low value).

Do SIM boxes have distinct fingerprints? One remaining question is whether SIM boxes have unique fingerprints compared to the other IoT devices. Based on our observations, we conclude that IoT device models also have highly distinguishing control-plane features but they may *not* be unique.

First, we count the numbers of distinct fingerprints in I and S as in §V-B, and found that two SIM boxes from different manufacturers show the same fingerprints. Moreover, we observe that identical feature vectors are easily generated by modifying the configurations of the baseband chipset using commands available in the product manual (§VIII). In addition, we argue that GSMA's TAC allocation rule somewhat contributes to the collision with fingerprints of IoT devices. Unlike smartphones whose TACs are allocated to the device model, IoT devices use the baseband's IMEI directly [40]. In other words, although both baseband and phone vendors generate differences in the fingerprint of the smartphone, IoT's fingerprints are highly coupled with the baseband vendors. As a result, those observations imply that IoT devices are more likely to have overlapped fingerprints, unlike smartphones.

TABLE VII: Exception cases in open-world evaluation.

Test Device	Expected result	Actual result	Cause
T Galaxy S7	Galaxy S7 (B)	Unknown	Firmware
T Galaxy Note 8	Galaxy Note 8 (A)	Unknown	Firmware
T Galaxy S10 (A)	Galaxy S10 (A)	Unknown	Firmware

F. How do the fingerprints handle unknown devices?

We next evaluate device model fingerprints in an open-world scenario to demonstrate whether a partial database could also handle random devices, including unknown ones.

Datasets. We construct two sets: a fingerprint database and a test dataset. The fingerprint database consists of all datasets in Tab. III except T, containing 279 traces from 102 devices in total (Tab. XIII). We also prepare T consisting of the fingerprints from 30 other smartphones not included in the database (Tab. XI). This test dataset contains 15 known phone models (which are the same model or cohort as one of the 102 devices) and 15 unknown models (which are not the same model as any of the 102 devices).

Evaluation process. We consider that the models are correctly identified in the following two cases: 1) an unknown model is identified as "unknown" (*i.e.*, no fingerprint in the database matches with the test device's fingerprint), and 2) a known model is identified as it is (*i.e.*, fingerprint of the same model in the database matches with the test device's fingerprint).

Evaluation results. We confirm that all unknown models are identified correctly. This finding supports **[O1]**, which implies that smartphones have practically unique fingerprints. Also, we could identify 12 out of 15 known models correctly while three exceptions occurred due to differences in firmware, as discussed in §V-B. Tab. VII shows the list of the three exceptions. This is because our database does not cover the new firmware of those three models. However, when deploying to a commercial network, these exceptions could be covered by the methods described in §VII-C. In conclusion, our method for device model fingerprinting can practically handle unknown devices, regardless of whether the database contains the fingerprints of those device models.

VI. SIM Box Fraud Prevention System

With the effective fingerprints we collect, we now design a SIM box fraud prevention system. Recall that this system is the first to *prevent* (*i.e.*, not detect after frauds begin) unauthorized SIM-boxed voice calls in cellular networks. The system employs an access control list (ACL) together with strict operational policies on IoTs (§VI-A). The security of the ACL is formally analyzed through UPPAAL (§VI-B).

A. ACL for Voice Calls

For a SIM box fraud prevention system, we propose an ACL for voice call in cellular networks. We focus only on the voice call because it is the main system resource exploited by fraudsters in our threat model.

We utilize three orthogonal fields obtained from the device attaching to the network: (i) a device-reported IMEI, (ii) a reported fingerprint (or control-plane features), and (iii) a subscribed plan (or simply a plan). The first two should be familiar

TABLE VIII: Proposed access control list for voice call access to cellular networks.

	Case	Reported IMEI	Fingerprint	Plans	Decision	Reason
Phase 1	1	Phone A	F_{PhoneA}	Phone	Accept	Correct IMEI
	2	Phone A	F_{PhoneB}	Phone	Reject	IMEI is spoofed
	3	Phone A	F_{IoTA} (= F_{IoTB})	Phone	Reject†	IMEI is spoofed
	4	Phone A	$F_{Unknown}$	Phone	Reject†	IMEI is spoofed
	5	IoT A (registered)	F_{PhoneA}	Any	Reject	IMEI is spoofed
	6	IoT A (registered)	F_{IoTA} (= F_{IoTB})	Any	Accept†	Correct, registered IMEI*
	7	IoT A (registered)	$F_{Unknown}$	Any	Reject†	IMEI is spoofed
	8	IoT B (non-registered)	F_{PhoneA}	Any	Reject	IMEI is spoofed
Phase 2	9	IoT B (non-registered)	F_{IoTA} (= F_{IoTB})	Phone	Reject†	Phone plan is not allowed for non-registered IoT devices*
	10	IoT B (non-registered)	F_{IoTA} (= F_{IoTB})	IoT	Accept†	IoT plan does not allow the voice service**
	11	IoT B (non-registered)	$F_{Unknown}$	Phone	Reject†	Phone plan is not allowed for non-registered IoT devices*
	12	IoT B (non-registered)	$F_{Unknown}$	IoT	Accept†	IoT plan does not allow the voice service**

† It may contain SIM box fraud.
* If IoT B is a SIM box, it can act like IoT A with a phone plan. Still, fraudsters require detailed information about the IoT A having the same fingerprint.
** If IoT B is a SIM box, it can be itself while using an IoT plan. Fraud attempt is limited due to the limitations in IoT plans though.

to readers by now. The third denotes the types of service plans that are typically chosen by end users when purchasing SIMs from MNOs. For the purpose of the ACL for voice calls, we use a coarse-grained distinction of service plans: *Phone* plans (which allow both voice calls and mobile data) or *IoT* plans (which only allow mobile data). Additionally, the system employs a fingerprint database, which stores the fingerprints of various device models. If the database contains the fingerprint of a certain model, the system can identify the model by matching the fingerprint in the database.

Tab. VIII shows the ACL we propose. To the best of our knowledge, it is the first access control policy for voice access in cellular networks that uses control-plane fingerprints. Note that *without control-plane fingerprints, one cannot enforce any robust access control because IMEI can be easily spoofed.*

For easier understanding of the ACL, we explain it in two phases. Phase 1 utilizes a fingerprint and reported IMEI to verify that the IMEI value is spoofed. When the system cannot decide the validity of the IMEI in Phase 1, the device is sent to Phase 2 for further decision using the subscribed plan.

Phase 1, validating IMEI. The main goal of Phase 1 is to validate the reported IMEIs of devices — *i.e.*, whether they are spoofed, valid, or undecided. Once the device reports the IMEI of a phone or a registered IoT, the reported fingerprints can be used to validate its IMEI. For validation, Phase 1 utilizes two fingerprints: (F1) the fingerprint of the model corresponding to the TAC value in the reported IMEI, and (F2) the reported fingerprint. Note that F1 is obtained from the fingerprint database of the system and may not be found if the database does not cover the model corresponding to the reported IMEI. Then, **[O1]** and **[O2]** imply that we can use F1 and F2 to make a concrete decision in the following cases.

1) If F1 and F2 match, we consider it a non-fraud case (Cases 1 and 6).
2) If F1 is found in the database but it does not match F2, we consider it a fraud case (Cases 2–5 and 7).
3) If F2 matches one of the smartphone fingerprints in the database but its IMEI reports otherwise, we consider it a fraud case (Case 8).

One important property is that Phase 1 ensures *zero false rejects* (*i.e.*, no legitimate user is rejected) assuming that our system contains all fingerprints of the smartphone models in the network. The implication when this assumption does not hold is discussed in §IX.

Phase 2, checking subscription plans. Phase 2 handles the devices whose IMEIs cannot not be concretely validated in Phase 1. Note that these are the devices that report the IMEI of a non-registered IoT devices. Unlike smartphones, the model of IoT devices cannot be determined strictly with fingerprints due to **[O3]**. In other words, the possibility of SIM box usage cannot be completely ruled out in these cases. To this end, we rely on a couple of new *operational* policy rules, which should be deployed at each MNO for better SIM boxes handling. To be specific, the operational policies confirm unauthorized access to the system by examining the subscription plans along with the IMEI and control-plane features (Cases 9–12 in Tab. VIII). Again, we use a coarse-grained categorization of plans: phone plans and IoT plans.

We envision that MNOs adopt the following simple operational policy: "Phone plans should *not* be allowed for non-registered IoT devices." This policy indeed forces all lawful, authorized SIM box applications to register with the MNOs (*e.g.*, the Bring Your Own Device policy at T-Mobile [16]) so that they can have voice call access. Along with this operational policy, Phase 2 concludes to accept devices with two criteria: (Cases 9–12 in Tab. VIII):

1) If the identified device is deemed to be an IoT device and subscribes to an IoT plan, accept the device (Cases 10, 12).
2) If the identified device is deemed to be an IoT device and subscribes to a phone plan, reject its access (Cases 9, 11).

B. Security Analysis on the ACL

We next conduct security analysis on the ACL to ensure its completeness. In particular, we examine the existence of *false reject* (a legitimate user is rejected) and *false accept* (an unauthorized SIM box is accepted). Note that the system should have zero false rejects as the MNO cares about the service availability for the users first. Simultaneously, for the purpose of the system, it should not allow false accept cases.

Analysis procedure. To verify wrong decisions of the ACL, we utilize UPPAAL [36], a model-checking tool. With UPPAAL, we generate a model with two major stages: a decision-making stage and a validation stage (Fig. 4). In the first

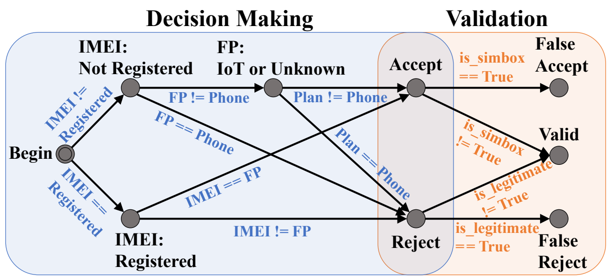

Fig. 4: UPPAAL model used for security analysis of ACL.

stage, the model receives input composed of five parameters: is_legitmiate, is_simbox, reported IMEI, reported fingerprint, and plan. The first two parameters, is_legitmiate and is_simbox, are ground truths used to check false rejects and false accepts, respectively. The remaining three parameters represent the properties of a device connected to the network. For each property, we define 1) three reported IMEIs (phone, registered IoT, and unregistered IoT), 2) three reported fingerprints (phone, IoT, and unknown), and 3) two plans (phone and IoT). For all 18 possible cases, our model determines whether to accept each case according to the ACL.

The model then verifies which cases lead to the false decision by comparing acceptance results from the above stage with the type of input devices, *i.e.* SIM box or legitimate user. As a result of verification, two wrong decisions can occur: *false accept* and *false reject*. False accept implies that an unauthorized SIM box can be accepted by the network. On the other hand, false reject means that the legitimate user is restricted to utilize the cellular network services.

Results. We confirm that the results of model checking are in accordance with what we discussed. Among all possible cases, none of the cases show a false reject. Meanwhile, we find that the following cases of false accept: Cases 6, 10, and 12 (Tab. VIII). However, fraud could not occur if the device uses IoT plans, as they are prohibited for voice calls under our policy. Thus, we conclude that the system has no false rejects, but false accepts could occur in: Case 6 with a phone plan. In Appendix §B, we discuss each case in more detail.

False accept. We observe that the ACL could experience false accepts once fraudsters exploit Case 6 with phone plans. For instance, the ACL accepts SIM boxes reporting the IMEI of a registered IoT device having the same fingerprint. This does not, however, reduce the effectiveness of our SIM box fraud prevention system at a meaningful scale because to exploit this niche false accept case, an adversary must know the exact IMEI of the registered IoT device that happens to share the fingerprint with their SIM box. We show in §VIII that it is non-trivial and costly to manipulate control-plane messages. Additionally, we believe that one can easily distinguish typical IoT devices using CDR or usage pattern analysis and easily detect fraudsters by adopting IMEI duplication checking logic based on the registration/subscription information.

VII. Deployment in Operational Networks

Our ultimate goal is to consider the deployment of the system for commercial networks in addition to presenting a high-level concept of the SIM box fraud prevention system. More specifically, we provide practical deployment considerations for the system. First, we provide an analysis of the system overhead in various scenarios: bootstrapping (§VII-A), specification updates (§VII-B), and the release of new devices and firmware versions (§VII-C). Second, we consider the changes required to the operational network to deploy the system (§VII-D).

A. *Overhead Analysis in Bootstrapping the System*

In the system bootstrapping state, the main overhead would be to build the fingerprint database. Recall that the construction requires four steps (Fig. 2): (S1) feature vector collection and conversion, (S2) specification analysis, (S3) intra-model analysis, and (S4) feature vector refinement. Among them, only S1 and S2 require manual efforts, while S3 and S4 are automatic processes. Due to this fact, we mainly consider the system overheads from S1 and S2 in the bootstrap stage.

Specification analysis (S2). To check the role of each feature and determine whether it is suitable for constructing the fingerprint, manual inspection of the 3GPP specification is inevitable. Intuitively, this process may seem overwhelming, even with the help of the two rules we set for the analysis (§IV-D). However, we benefited from the structure of the specifications: 1) the table structure of the feature definition, and 2) the nested structure. The features in the messages are given in a table, which makes it easier to examine them. Also, definitions are given in a nested format, where tables lead to a section or document with the definitions we look for. As a result of our empirical measurement, we could finish this analysis in *5h*. Note that this process is required at the system bootstrap stage only once.

Feature vector collection and conversion (S1). In our setup, it took less than 20 min for one smartphone model to process this step considering its configuration modification. We used the modified srsLTE implementation and manually changed the smartphone configuration. Notably, we can automate this collection procedure by utilizing a phone mirroring and automation tool such as *scrcpy* [27], which enables computer-based phone control. This approach enables us to automate the process to adjust the configurations and collect messages automatically. Therefore, with the predefined control flow, we can automatically change the configurations described.

B. *Overhead in Specification Updates*

After the bootstrap, manual inspection in specification analysis (S2) is often required. As the filter used to prune the features (§IV) is built based on specifications, it should be updated whenever the specifications are updated, which may require a huge maintenance cost. However, there are several reasons supporting that the proposed system is deployable.

First, the specifications are not written from scratch, which implies that the effort necessary to analyze the added features does not require considerable manual effort afterward. For instance, only 10 new pages of specification analysis are required to upgrade the system from release 16 to 17. To quantify the overhead, we count the number of newly added IEs and fields in updates. We examine the two specifications [5], [6] and compare the document with every release. Tab. IX reveals that on average, 21 IEs and 73.3 fields are newly

TABLE IX: # of new features in each specification version.

Release	9	10	11	12	13	14	15	16	17	Average
# of UE Cap. Fields	22	30	27	47	103	105	181	122	23	73.3
# of Attach Req. IEs	12	14	12	9	17	5	85	26	9	21

introduced per release. For instance, a large number of new features are included in release 15 as the new radio technology (5G) is introduced. Second, the specifications take a few months to be updated, so investigation is only required every few months. Also, the specifications always appear before the corresponding devices, making it possible to generate a filter in advance.

C. Overhead in Handling New Devices/Firmware

Considering the operational networks in which many new devices are introduced every year, the system needs to conduct the process of building a fingerprint and updating an existing fingerprint. In this context, there are two potential concerns: (1) overheads in fingerprint construction and (2) difficulties in tracking all of the new devices and firmware.

First, the required steps for handling new devices and firmware do not necessarily involve manual effort. Except for the manual analysis of the specifications, only the remaining steps in Fig. 2 must be conducted. This is because any new device models should adhere to the latest standards, which were already analyzed and used to generate a filter.

Second, MNOs can collect new fingerprints and add them to the database before the new products are even released with the help of relationships between the manufacturers. In general, before the deployment of new phones and firmware versions, MNOs test them on their testbed to check network compatibility. Thus, MNOs can collect the control-plane messages of the new devices and devices with new firmware from the tests. To confirm this, we asked a tier-1 MNO and a global tier-1 phone manufacturer and received the following two responses. First, Apple and Samsung, two major manufacturers, provide new phones and firmwares to MNO for every new release to ensure that a new or updated device will not harm the network. Second, new firmware and devices can be released only after they are authorized by the MNOs through the tests. This policy holds for all Android phones.

D. Required Changes on the Operational Network

There exists one immediate concern that could arise when deploying the proposed system in large operational LTE networks. Our system utilizes the control-plane messages in two different control-plane protocols (*i.e.*, NAS and RRC) and each protocol is managed by different entities (*i.e.*, MME and eNB), respectively. Thus, each entity with different roles has limited access to different sets of information on the control-plane messages; *e.g.*, eNBs cannot monitor NAS messages and MMEs cannot see RRC messages.

Luckily, there already exist channels between eNBs and MMEs, and more importantly, they are used to exchange the control-plane messages that we utilize. According to the specifications, the eNB delivers the device capabilities to the MME whenever it receives 1) `UECapabilityInformation` from the device or 2) a request from the MME. For this, an `UE Capability Info Indication` message in the S1AP [7], the control-plane protocol between eNB and MME, is used. Therefore, our detection system does not require additional channels for the operational networks or new message formats for the new standards. Only by communicating with MMEs, the core network can collect the control-plane messages used to generate fingerprints. Therefore, the core network can receive both messages with only incremental changes on MMEs.

VIII. Circumventing Our Prevention System

Advanced fraudsters will wish to circumvent our SIM box fraud prevention system once the system is deployed. They will attempt to manipulate the control-plane messages and IMEI to the model that SIM box impersonates. However, we argue that such mimicry of the control-plane messages is highly impractical and costly in all attack scenarios. As we do not consider supply-chain attackers, the fraudsters cannot manipulate the baseband chipset itself. Thus, there are three options for the fraudsters to manipulate the control-plane messages: 1) change the SIM box configuration (§VIII-A), 2) modify the control-plane messages with MitM device (§VIII-B), and 3) implement fully controllable software SIM box (§VIII-C).

A. Changing the SIM box Configuration

We first investigate whether fraudsters can generate the same fingerprints as smartphones using their own SIM boxes. For this, we take the approach of being fraudsters and building our own SIM boxes to have a greater ability on modifying the control-plane messages. We embed two chipsets, EC25-E [19] with two versions, widely used by SIM boxes on the market, onto the dev-board [20].

The operations that could affect the control-plane messages are performed in two ways: (1) by sending AT commands and (2) by modifying the configuration files in the firmware of the chipset (called modem configuration binary, MBN). Inspecting the AT command manuals [44] reveals 16 cellular-communication-related AT commands and 12 MBN files that could be adopted to the baseband chipset (Tab. XII).

We first investigate how such modifications affect control-plane messages. By sending AT commands and modifying MBN files, we could modify the SIM boxes and collect 75 feature vectors in total (see Tab. XII for the list of modifications). Comparative analysis of the feature vectors reveals that 383 features are affected by the operations. This observation suggests that fraudsters may have the freedom to change those features.

Base on these observations, we conduct an evaluation to determine if SIM boxes could have the same feature vectors as smartphones. To assume that the 383 features could be modified freely by fraudsters, we exclude these 383 features from the entire set of features and evaluate the remaining features. Comparing the feature vectors in our entire phone dataset (Tab. III) to the 75 generated feature vectors reveals that no two vectors are identical. As a result, we confirm that a SIM box cannot mimic a phone. In contrast, the comparison with I reveals three overlapping SIM boxes, and one of them has a feature vector identical to that generated by the modification. These results also support observations **[O2]** and **[O3]**. In conclusion, *without manufacturer's support or changing the* *baseband chipset itself*, mimicking the phone's control-plane message is infeasible for fraudsters.

B. Control-plane Message Manipulation with MitM Device

We next consider fraudsters who attempt to report manipulated control-plane messages with additional devices. In particular, fraudsters can apply a Man-in-the-Middle (MitM) scheme [46], [58], where a MitM device sends a manipulated control-plane message instead of the SIM box's one. The MitM often utilizes the SDR and software LTE stack. For example, in a capability hijacking attack described in [51], fraudsters try to circumvent our system by relaying different capability information to the network.

While this approach seems a promising means of bypassing our system, the MNO can easily block such attacks with the help of slight changes to the network access procedure of the device. The only change is to request and receive `UECapabilityInformation` message, which our system uses to generate the fingerprint, only after access stratum security activation. As the `UECapabilityInformation` message is now encrypted and integrity-protected, the fraudsters using MitM device are not able to manipulate the message. Note that both UE and network do not require any changes in their implementation as it only requires changing the order of message transmission. Notably, this change in the procedure is an exact recommendation in the standard [6]. According to our investigation, we observe that tier 1 MNO requests the message (`UECapabilityInformation`) after the access stratum security activation. In summary, unless the collusion with device manufacturers and baseband vendors together, it is hard to bypass our fraud detection system by reporting manipulated control-plane messages to the network.

C. Implementing Software SIM box

Fraudsters may build their own software SIM boxes to circumvent our system. However, we argue that this possibility cannot motivate fraudsters, as they need to put a significant engineering effort to make it as a COTS SIM box (*e.g.*, connect to an IMS server to use VoLTE, make a phone call, and support most functions of a SIM box). It is worth mentioning that it is infeasible to manipulate the control-plane messages of a COTS SIM box, as discussed in §VIII-A and §VIII-B. Therefore, rather than purchasing a COTS SIM box, fraudsters need to implement SIM box software from scratch to alter the control-plane messages. One way to implement the SIM box software is to leverage a SDR-based UE. However, all open-source UE implementations (including the state-of-the-art open-source LTE, srsUE [28]) do not support quite a few common functionalities of commercial UEs (*e.g.*, voice calls, SMS, 3G, 2G, circuit switch fallback, or other advanced functions).

To show the empirical evidence, we inspected four functions for srsUE, voice call supported by VoLTE, SMS over IMS, SMS over NAS, and 3G redirection. For this, we built the cellular network testbed utilizing commercial UEs, srsUE, srseNB, and Amari callbox [12]. As a result, we verified that all the functions do not work properly on srsUE. We confirmed this with the main maintainer of the srsLTE project, through personal communication [43]. Once we request such behaviors to the srsUE, we could observe abnormal behaviors on the network side. For example, we could see control-plane error messages when we attempted to make a call to srsUE, as it is not connected to the IMS. Therefore, any features associated with these basic functionalities, of course, cannot work because they don't provide these functionalities. Note that we cannot test for the possibility of counterfeiting those features unless we implement the corresponding function stacks.

IX. Limitations

Easing policy in the real world. Our ACL works with no false rejects and niche false accepts. Unfortunately, the statement holds only if the network contains all fingerprints of the smartphone models attaching to the network. If not, legitimate users with phones having unknown fingerprints will get false rejects once they attach to the network. To handle this problem, MNOs can lower the decision level from "reject" to "flagging" to prevent such false rejects of legitimate users. For the "flagged" devices, MNO can employ additional technologies [10], [11], [13], [23], [32], [33], [38], [39], [45], [50], which are orthogonal to our ACL, to detect SIM box calls on them. Once the system is deployed with well-gathered smartphone fingerprints (not fully collected), the ACL flags only a few devices so that the overhead of deploying such technologies is reduced on a great scale.

Dataset size. Even though we utilized a dataset of 102 distinct device models, the dataset was relatively small compared to the number of device models on the market. However, comprehensive fingerprinting experiments with various scenarios and the dataset revealed that 1) most phones, except for reasonable cohorts, have distinct fingerprints, and 2) SIM boxes have distinct fingerprints from phones. We also discovered some reasons for such observations using feature analysis. In summary, even though the dataset cannot represent all devices in the market, we anticipate that the two observations will hold for the majority of devices in the market. This also implies that the proposed prevention system is highly likely to be feasible. We also plan to collaborate with MNOs to obtain a better understanding of our results and the effectiveness of the proposed system with larger datasets.

X. Related Work

Device identification using control-plane messages. An approach for identifying devices using the contents of `ATTACH Request` and `UECapabiltyInformation` messages is presented in [34], [51]. These studies only examined the feasibility of baseband manufacturer identification, baseband modem identification, and device type identification with small-scale data. In contrast, using large-scale data, we demonstrate that device model identification is feasible and analyze how each factor (*i.e.*, baseband vendor, phone vendor, and device model) affects the fingerprints. Park *et al.* [41] proposed a baseband vendor identification based on active testing of control-plane messages. In an actual network, however, this approach is challenging because an abnormal message may reduce the quality of service or cause the user to experience denial of service. Unlike all previous researchers, we performed feature analysis to eliminate features that should not be part of the fingerprint. Also, we considered end-user options in constructing the fingerprint, which may result in changes to

TABLE X: Comparison to previous works.

	Fingerprint Target	# of Devices	Testing Method	# of Used Features	Feature Analysis	End-User Options
Shaik.et.el [51]	Baseband-Vendor, OS, Device Type	36	Passive	Unknown	X	X
LTrack [34]	Baseband-Modem	22	Passive	Unknown	X	X
DoLTEst [41]	Baseband-Vendor	5	Active	5 (msgs used)	X	X
Ours	Device-Model	102	Passive	922	O	O

the control-plane messages utilized for fingerprints. Tab. X compares between our work and the previous works.

Other approaches for detecting SIM boxes. Several studies have been conducted on detecting SIM boxes. The majority of past SIM box identification techniques relied on the call detail record (CDR) created during phone calls [10], [11], [23], [32], [33], [38], [39], [50]. Another study [24] evaluated the voice and guessed the number of speakers for the same SIM card. MNOs can detect SIM boxes with both kinds of methods only after fraud has occurred. Some other approaches for detecting SIM boxes used voice call quality as a criterion (Pindr0p [13] and Boxed out [45]). They characterized the SIM boxed calls whose audio had a distinct pattern due to packet drops in the VoIP connection. Instead, our research is focused on the characteristics of the SIM box unit itself in control-plane of cellular networks. In addition, our SIM box unit detection scheme is orthogonal to existing SIM boxed call detection schemes, and they can be utilized in conjunction to complement one another. Similar to our work, LATRO asserts that it can detect SIM boxes using signaling message signatures [55]; however, the specific mechanism for detection is not disclosed.

XI. Conclusion

By utilizing a large-scale dataset, we show that device model fingerprinting is possible through concrete feature analysis and consideration of user-configurable options. By utilizing this fingerprinting, we propose a SIM box fraud prevention system that prevents SIM boxes from using cellular networks. Practical deployment issues such as fingerprint collection are discussed in detail so that our system can easily be adopted to the commercial network. We are currently in discussion with a tier 1 carrier to apply the proposed mechanism inside its infrastructure.

Acknowledgement

We sincerely appreciate our anonymous reviewers for valuable feedback and suggestions. We would also like to thank Cheoljun Park and Jiho Lee for discussions and helping us to develop our paper. This research was supported and funded by the Korean National Police Agency [Project Name: Tracking and identifying devices and call traffic in voice phishing ecosystem / Project Number: PR10-03-020-22]. This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2020-0-00428, automated security diagnosis framework for cellular network protocol using formal and comparative analysis, and No.2022-0-01202, regional strategic industry convergence security core talent training business).

References

[1] "Preventing SIM Box Fraud Using Device Fingerprinting," https://sites.google.com/view/devicefingerprinting.

[2] 3GPP. CR S3-212362, "CR to TS 43.020 - R11 - Removal of GEA1 and GEA2 due to security concerns," 2021.

[3] 3GPP. TS 24.007, v17.4.0, "Mobile radio interface signalling layer 3; General aspects ," 2022.

[4] 3GPP. TS 24.008, v17.7.0, "Mobile radio interface Layer 3 specification; Core network protocols; Stage 3," 2022.

[5] 3GPP. TS 24.301, v17.7.0, "Non-Access-Stratum (NAS) protocol for Evolved Packet System (EPS); Stage 3," 2022.

[6] 3GPP. TS 36.331, v16.9.0, "LTE RRC Protocol specification," 2022.

[7] 3GPP. TS 36.413, v16.8.0, "S1 Application Protocol (S1AP)," 2021.

[8] 3GPP. TS 43.020, v11.3.0, "Digital cellular telecommunications system (Phase 2+) (GSM); Security related network functions," 2021.

[9] AB Handshake Corporation, "Interconnect Bypass - Which Regions of the World are Most Vulnerable?" 2021, https://www.totaltele.com/511656/Interconnect-Bypass-Which-Regions-of-the-World-are-Most-Vulnerable.

[10] M. R. AlBougha, "Comparing data mining classification algorithms in detection of simbox fraud," 2016.

[11] I. S. Alsadi and N. Abuhamoud, "Study to use neo4j to analysis and detection sim-box fraud," *Journal of Pure & Applied Sciences*, vol. 17, no. 4, 2018.

[12] "Amarisoft. [n.d.]. Amarisoft Callbox Classic," "https://www.amarisoft.com".

[13] V. A. Balasubramaniyan, A. Poonawalla, M. Ahamad, M. T. Hunter, and P. Traynor, "Pindr0p: Using single-ended audio features to determine call provenance," in *ACM CCS*, 2010, pp. 109–120.

[14] C. Beierle, P. Derbez, G. Leander, G. Leurent, H. Raddum, Y. Rotella, D. Rupprecht, and L. Stennes, "Cryptanalysis of the GPRS Encryption Algorithms GEA-1 and GEA-2," Cryptology ePrint Archive, Paper 2021/819, 2021, https://eprint.iacr.org/2021/819. [Online]. Available: https://eprint.iacr.org/2021/819

[15] T. Brewster, "Fraudsters Cloned Company Director's Voice In $35 Million Bank Heist, Police Find," 2021, https://www.forbes.com/sites/thomasbrewster/2021/10/14/huge-bank-fraud-uses-deep-fake-voice-tech-to-steal-millions/?sh=56c41fd57559.

[16] "Bring your phone to T-Mobile," https://www.t-mobile.com/resources/bring-your-own-phone.

[17] "In 2020, China cracked 322,000 telecom and network fraud cases, saving more than 187 billion yuan," 2021, https://www.chinanews.com.cn/gn/2021/04-09/9451434.shtml.

[18] Communications Fraud Control Association, "Fraud Loss Survey," 2021.

[19] "Quectel LTE EC25-E," https://www.quectel.com/product/lte-ec25-e.

[20] "Quectel Mini PCIe EVB Kit," https://www.quectel.com/product/lte-open-evb-kit.

[21] "EFS professional," https://goo.gl/wQKJ59.

[22] "Ejoin Corporate Communication Solution," http://en.ejointech.com/news/13.html.

[23] A. H. Elmi, S. Ibrahim, and R. Sallehuddin, "Detecting sim box fraud using neural network," in *IT Convergence and Security 2012*. Springer, 2013, pp. 575–582.

[24] O. M. Elrajubi, A. M. Elshawesh, and M. A. Abuzaraida, "Detection of bypass fraud based on speaker recognition," in *2017 8th International Conference on Information Technology (ICIT)*. IEEE, 2017, pp. 50–54.

[25] L. Ferreira, L. Silva, D. Pinho, F. Morais, C. M. Martins, P. M. Pires, P. Fidalgo, H. Rodrigues, P. Cortez, and A. Pilastri, "A federated machine learning approach to detect international revenue share fraud on the 5G edge," in *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*, 2022, pp. 1432–1439.

[26] S. Fischer, "Observed time difference of arrival (OTDOA) positioning in 3GPP LTE," *Qualcomm White Pap*, vol. 1, no. 1, pp. 1–62, 2014.

[27] Genymobile, "scrcpy v1.24." [Online]. Available: https://github.com/Genymobile/scrcpy

[28] I. Gomez-Miguelez, A. Garcia-Saavedra, P. D. Sutton, P. Serrano, C. Cano, and D. J. Leith, "srsLTE: An Open-Source Platform for LTE Evolution and Experimentation," in *ACM WiNTECH*, 2016, https://github.com/srsran/srsRAN.

[29] GSMA. TS.06, v16.0, "IMEI Allocation and Approval Process," 2019.

[30] E. Houweling, "Chinese 'anti-fraud centre' becomes most downloaded app," 2021, https://www.verdict.co.uk/chinese-anti-fraud-centre-becomes-most-downloaded-app/.

[31] N. Jiang, Y. Jin, A. Skudlark, W.-L. Hsu, G. Jacobson, S. Prakasam, and Z.-L. Zhang, "Isolating and analyzing fraud activities in a large cellular network via voice call graph analysis," in *Proceedings of the 10th international conference on Mobile systems, applications, and services*, 2012, pp. 253–266.

[32] H. Kahsu, "Sim-box fraud detection using data mining techniques: the case of ethio telecom," Ph.D. dissertation, Ph. D. thesis AAU, 2018.

[33] M. Kashir and S. Bashir, "Machine learning techniques for sim box fraud detection," in *ComTech*. IEEE, 2019, pp. 4–8.

[34] M. Kotuliak, S. Erni, P. Leu, M. Roeschlin, and S. Capkun, "LTrack: Stealthy Tracking of Mobile Phones in LTE," in *USENIX Security*, 2022.

[35] A. J. Kouam, A. C. Viana, and A. Tchana, "SIMBox Bypass Frauds in Cellular Networks: Strategies, Evolution, Detection, and Future Directions," *IEEE COMST*, vol. 23, no. 4, pp. 2295–2323, 2021.

[36] K. G. Larsen, P. Pettersson, and W. Yi, "UPPAAL in a nutshell," *International journal on software tools for technology transfer*, vol. 1, no. 1, pp. 134–152, 1997.

[37] B. Lee, "Number of cases of voice phishing ↓, damage 3 times ↑... Loan scam type," 2022, http://www.nspna.com/news/?mode=view&newsid=552544.

[38] H. M. Marah, O. M. Elrajubi, and A. A. Abouda, "Fraud detection in international calls using fuzzy logic," in *International Conference on Computer Vision and Image Analysis Applications*, 2015, pp. 1–6.

[39] I. Murynets, M. Zabarankin, R. P. Jover, and A. Panagia, "Analysis and detection of SIMbox fraud in mobility networks," in *IEEE INFOCOM*. IEEE, 2014, pp. 1519–1526.

[40] B. Pareglio, "How do I get A TAC/IMEI for my mobile IOT device?" Jan 2019. [Online]. Available: https://www.gsma.com/iot/resources/how-do-i-get-a-tac-imei-for-my-mobile-iot-device/

[41] C. Park, S. Bae, B. Oh, J. Lee, E. Lee, I. Yun, and Y. Kim, "DoLTEst: In-depth Downlink Negative Testing Framework for LTE Devices," in *USENIX Security*, 2022.

[42] H. Park, "Are there voice phishing devices around me? Arrest of a crew member who operated a switchgear," 2021, http://www.kookje.co.kr/news2011/asp/newsbody.asp?code=0300&key=20211104.99099001305.

[43] A. Puschmann, private communication, 2022.

[44] "Quectel EC25 EC21 AT Commands Manual V1.3," https://www.quectel.com/download/quectel_ec25ec21_at_commands_manual_v1-3.

[45] B. Reaves, E. Shernan, A. Bates, H. Carter, and P. Traynor, "Boxed out: Blocking cellular interconnect bypass fraud at the network edge," in *USENIX Security*, 2015, pp. 833–848.

[46] D. Rupprecht, K. Kohls, T. Holz, and C. Pöpper, "Breaking LTE on Layer Two," in *2019 IEEE Symposium on Security and Privacy (SP)*, 2019, pp. 1121–1136.

[47] M. Sahin and A. Francillon, "Over-the-top bypass: Study of a recent telephony fraud," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 1106–1117.

[48] ——, "Understanding and Detecting International Revenue Share Fraud." in *NDSS*, 2021.

[49] M. Sahin, A. Francillon, P. Gupta, and M. Ahamad, "Sok: Fraud in telephony networks," in *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2017, pp. 235–250.

[50] R. Sallehuddin, S. Ibrahim, A. M. Zain, and A. H. Elmi, "Detecting sim box fraud by using support vector machine and artificial neural network," *Jurnal Teknologi*, vol. 74, no. 1, 2015.

[51] A. Shaik, R. Borgaonkar, S. Park, and J.-P. Seifert, "New vulnerabilities in 4G and 5G cellular access network protocols: exposing device capabilities," in *Conference on Security and Privacy in Wireless and Mobile Networks*, 2019.

[52] ""Manipulation of '070 → 010', falsification of caller ID, voice phishing arrest per day"," 2021, https://newsis.com/view/?id=NISX20211104_0001639023.

[53] S. Song, "144 'phone number changers' installed in motels nationwide... 14 arrests per day for voice phishing," 2021, https://www.asiae.co.kr/article/2021111909364820433.

[54] "Sysmocom SIM/USIM/ISIM cards," https://www.sysmocom.de/products/lab/sysmousim/index.html.

[55] Technology Research Institute and LATRO Services, "Network Protocol Analysis: A New Tool for Blocking International Bypass Fraud Before Revenue is Lost," 2015.

[56] "USRP B210," https://www.ettus.com/UB210-KIT.

[57] T. Xie, G.-H. Tu, C.-Y. Li, and C. Peng, "How can IoT services pose new security threats in operational cellular networks?" *IEEE TMC*, 2020.

[58] H. Yang, S. Bae, M. Son, H. Kim, S. M. Kim, and Y. Kim, "Hiding in Plain Signal: Physical Signal Overshadowing Attack on LTE," in *USENIX Security Symposium*, 2019.

APPENDIX

A. ACRONYMS

3GPP	Third Generation Partnership Project
5G-NSA	5G Non-standalone
5G-SA	5G Standalone
ACL	Access Control List
ASN.1	Abstract Syntax Notation One
CA	Carrier Aggregation
CDR	Call Data Record
C-RNTI	Cell RNTI
eNB	Evolved Node B
EEA	EPS Encryption Algorithm
EIA	EPS Integrity Algorithm
EPC	Evolved Packet Core
EPS	Evolved Packet System
GUTI	Globally Unique Temporary Identity
IE	Information Element
IMEI	International Mobile Equipment Identity
IMSI	International Mobile Subscriber Identity
IoT	Internet of Things
MME	Mobility Management Entity
MNO	Mobile Network Operator
NAS	Non Access Stratum
PLMN	Public Land Mobile Network
RNTI	Radio Network Temporary Identifier
RRC	Radio Resource Control
SDR	Software Defined Radio
SIM	Subscriber Identity Module
TAC	Type Allocation Code
TMSI	Temporary Mobile Subscriber Identity
UE	User Equipment
USIM	Universal SIM
VoIP	Voice over IP
VoLTE	Voice over LTE

B. Security Analysis on Proposed Access Control List

In order to verify whether the proposed ACL (Tab. VIII) can prevent the SIM box fraud effectively, we conduct security analysis on the ACL. In total, there are 6 possible SIM box fraud attack scenarios, according to the (1) IMEI that SIM box reports to the network and (2) its subscription plan. We name each scenario to represent three conditions as followings: First, SIM box can possibly report its IMEI to the network using (1) its own IMEI (not spoofed, SIM box IMEI), (2) IMEI of a phone (Phone IMEI), or (3) IMEI of an IoT (IoT IMEI); Second, the reported IMEI could be registered to the network or not; and Third, the SIM box can use two subscription plans (Phone plan or IoT plan). For each scenario, we match the corresponding cases (policies) of the ACL and verify the decision.

A1: SIM box IMEI, Unregistered, Phone Plan. This is the standard scenario when a fraudster uses a SIM box without configuration change or IMEI spoofing. According to its fingerprint, we can separate the scenario into two cases. First, if the SIM box shares the same fingerprint as other IoT devices, it is handled in case 9. Second, if the SIM box has an unknown fingerprint, it is dealt with in case 11. In both instances, unregistered IoT makes use of a forbidden phone plan. Accordingly, we reject them per our policy and conclude that fraudsters cannot exploit A1.

A2: Phone IMEI, Registered, Phone Plan. To avoid being blocked in the scenario A1, a savvy fraudster might counterfeit the reported IMEI into a phone's IMEI. However, as described in §V, fingerprints of the SIM box and the phone are distinct, and a fraudster cannot imitate the phone's fingerprint using the SIM box. Consequently, we know that the IMEI is a forgery regardless of the type of fingerprint (IoT or unknown). This scenario involves cases 3 and 4, which we reject due to IMEI spoofing. As a result, we conclude that fraudsters cannot exploit A2.

A3: IoT IMEI, Un/Registered, Phone Plan. Even if a fraudster uses the IoT's IMEI, the SIM box is completely rejected with one exception (case 6). If the reported IMEI is registered in the network, this scenario is handled in case 6 or 7. If not, this scenario is handled in case 9 or 11. In case 7, if the system does not know the fingerprint from control plane messages, then it does not make sense. This is because the reported IMEI indicates that the system knows the device. Therefore, the SIM box's access would be rejected. However, in case 6, if the SIM box spoofs its IMEI with the device having the same fingerprint with it, the network allows the attach. This is the only case of false accept in our ACL, as explained in §VI-B. Still, the fraudsters must either know the precise IMEI of the registered IoT device that shares their SIM box's fingerprint or set up the SIM box's control-plane features to imitate the target IoT device's features. This means that the possible fraud scenario is narrow so that the proposed access control system can cover most of the SIM box fraud. In addition, when the IMEI of the registered IoT device (or SIM box) is exposed, one can easily distinguish typical IoT devices using CDR or usage pattern analysis, and easily detect the fraudster by adopting IMEI duplication-checking logic based on the registration/subscription information. Lastly, in cases 9 and 11, a phone plan is not allowed for non-registered devices. Therefore, the SIM box's access is rejected.

A4: SIM box IMEI, Unregistered, IoT Plan. This scenario is identical to A1, except for an IoT plan subscription. Cases 10 and 12 are included in this scenario, which involves unregistered IoT devices with IoT plans and SIM boxes with IoT plans. To prevent false rejects (blocking access of legitimate unregistered IoT devices), we have to accept both cases even considering that a SIM box might access the network. Even allowing both cases, the fraudsters cannot accomplish their objective as voice calls are prohibited in IoT plans under our policy. Thus, we conclude that fraudsters cannot exploit A4.

A5: Phone IMEI, Registered, IoT Plan. In this scenario, a SIM box is attempting to masquerade as a legitimate phone while subscribing to an IoT plan. However, this scenario is not included in the ACL because it is simple to detect and reject the SIM box for two reasons. First, legitimate phones do not subscribe to an IoT plan, and second, the SIM box reports IMEI which does not match with its fingerprint. Particularly, the reported IMEI belongs to a phone, but its fingerprint is "unknown". Thus, we conclude that fraudsters cannot exploit A5.

A6: IoT IMEI, Un/Registered, IoT Plan. In the last scenario, the SIM box uses a faked IMEI of an IoT device and subscribes to an IoT plan. When the reported IMEI is registered, cases 6 and 7 are applicable to this scenario. If the fingerprint does not match the registered IoT device, our policy rejects the access being considered as IMEI spoofing, according to case 7. Note that, to prevent false reject, we might accept the SIM box if the fingerprint matches the registered IoT (Case 6). However, similar to A4, this is not a significant issue because making a voice call with an IoT plan is prohibited. When it comes to the scenario where the reported IMEI is unregistered, cases 10 and 12 handles such a scenario. Since it is possible to subscribe to an IoT plan without registering, cases 10 and 12 must be accepted to avoid rejecting a legitimate user. Again, similar to the A4, it might accept SIM boxes. However, it is still acceptable as the IoT plan policy does not allow voice service. In conclusion, fraudsters cannot exploit A6.

C. *Additional Factors that Affect Fingerprints.*

We discuss two additional factors that can affect fingerprints: eNB manufacturers and MNOs of the inserted SIM cards (*i.e.*, PLMN).

First, to investigate the impact of eNB manufacturers, We compare the two fingerprints of the same device, which are generated by connecting the device to the two eNBs from different manufacturers. As a result, we confirm that the fingerprints remain the same even though the device communicates with eNB having different configurations. (*e.g.*, manufacturer, bandwidth, and radio frequency). This is because the two control-plane messages we utilize are used to inform the network of the device's capabilities; thus, they are not affected by the condition of current radio connection. Furthermore, the devices transmit the same capabilities even when accessing a base station of a different network generation (*i.e.*, 5G base station).

Second, unlike the eNB manufacturer, the device's capabilities are affected by the SIM card's PLMN. To investigate the impact of the inserted SIM card, we obtain capabilities from 11 phones while varying SIM cards from three major MNOs. As a result, we observe that a) the devices have pre-defined information related to the network operation settings of each MNO and b) the fingerprints differ based on the SIM card accordingly.

For example, one of the MNOs does not operate the 3G network, and the devices equipped with that MNO's SIM card do not include 3G-related features (*e.g.*, 3G encryption algorithms, radio configuration for 3G) in their fingerprints. Furthermore, with the help of static analysis of the baseband firmware, we find an implementation differentiating some of the configurations (*e.g.*, `SRVCC`) depending on the MNO of equipped SIM card.

D. Additional Tables

TABLE XI: Devices used for open-world evaluation (Tab. VII).

#	Device	Phone Vendor	Baseband Vendor	Chipset
1	T Galaxy Note 4	Samsung	Samsung	Exynos 5433
2	T Galaxy Z Flip 4	Samsung	Qualcomm	SM8475 Snapdragon 8+ Gen 1
3	T S21 5G	Samsung	Samsung	Exynos 2100
4	T Galaxy Note 5	Samsung	Samsung	Exynos 7420 Octa
5	T Galaxy Note 8	Samsung	Samsung	Exynos 8895
6	T Galaxy Note 9	Samsung	Samsung	Exynos 9810
7	T Galaxy Note 10	Samsung	Samsung	Exynos 9825
8	T Galaxy S10 (A)	Samsung	Samsung	Exynos 9820
9	T Galaxy S10 (B)	Samsung	Samsung	Exynos 9820
10	T Galaxy S22+	Samsung	Qualcomm	SM8450 Snapdragon 8 Gen 1
11	T Galaxy S5 (A)	Samsung	Qualcomm	MSM8974AC Snapdragon 801
12	T Galaxy S5 (B)	Samsung	Qualcomm	APQ8084 Snapdragon 805
13	T Galaxy S5 (C)	Samsung	Qualcomm	APQ8084 Snapdragon 805
14	T Galaxy S7	Samsung	Qualcomm	MSM8996 Snapdragon 820
15	T Huawei Mate 9	Huawei	HiSilicon	Kirin 960
16	T Huawei P30 Pro	Huawei	HiSilicon	Kirin 980
17	T iPhone 13 Max Pro	Apple	Qualcomm	Snapdragon X60
18	T iPhone 7	Apple	Qualcomm	Snapdragon X12
19	T iPhone 7+	Apple	Qualcomm	Snapdragon X12
20	T iPhone 8	Apple	Intel	XMM 7480
21	T iPhone 8+	Apple	Intel	XMM 7480
22	T iPhone SE2	Apple	Intel	XMM 7660
23	T iPhone12 mini	Apple	Qualcomm	Snapdragon X55
24	T Nexus 5 (A)	LG	Qualcomm	MSM8974 Snapdragon 800
25	T Nexus 5 (B)	LG	Qualcomm	MSM8974 Snapdragon 800
26	T Nexus 5 (C)	LG	Qualcomm	MSM8974 Snapdragon 800
27	T Nexus 5X	LG	Qualcomm	MSM8992 Snapdragon 808
28	T Nexus 6 (A)	Motorola	Qualcomm	APQ8084 Snapdragon 805
29	T Nexus 6 (B)	Motorola	Qualcomm	APQ8084 Snapdragon 805
30	T Oppo A73 5G	Oppo	MediaTek	MT6853V Dimensity 720

TABLE XII: MBNs & AT commands affecting control-plane messages.

Idx	MBN name	# of feature vectors obtained
1	Reliance_India_VoLTE	1
2	TW_Mobile_China_VoLTE	2
3	Bouygues_France_VoLTE	2
4	VF_Germany_VoLTE	2
5	ROW_Generic_3GPP	2
6	Swis_switzerland_VoLTE_VoWiFi	1
7	Telstra-Commercial_VoLTE	1
8	Commercial-Smartfren	1
9	TF_Spain_VoLTE	1
10	Commercial-DT-VOLTE	1
11	STC_Saudi_VoLTE	1
12	Smartfren_Indonesia_VoLTE	1

Idx	Category	Configuration	Option	Description	# of feature vectors obtained
1	AT+QCFG	nwscanmode	Network search mode	Network Search Mode Configuration	2
2	AT+QCFG	nwscanseq	Network search sequence	Network Searching Sequence Configuration	2
3	AT+QCFG	roamservice	The mode of roam service	Roam Service Configuration	3
4	AT+QCFG	servicedomain	Service domain of UE	Service Domain Configuration	2
5	AT+QCFG	band	Bandval / Ltebandval / Tdsbandval	Band Configuration	8
6	AT+QCFG	hsdpacat	HSDPA category	HSDPA Category Configuration	5
7	AT+QCFG	hsupacat	HSUPA category	HSUPA Category Configuration	2
8	AT+QCFG	rrc	RRC release version	RRC Release Version Configuration	3
9	AT+QCFG	sgsn	SGSN release version	UE SGSN Release Version Configuration	3
10	AT+QCFG	msc	MSC release version	UE MSC Release Version Configuration	3
11	AT+QCFG	pdp/duplicatechk	Enable or disable establishment	Establish Multi PDNs with the Same APN	2
12	AT+QCFG	ims	Enable or disable IMS	Enable/Disable IMS compulsorily or follows MBN setting	5
13	AT+QGPSCFG	gnssconfig	GNSS On and off (GLONASS / BeiDou and QZSS / Galileo)	Configure Enabled GNSS Constellations	3
14	AT+QGPSCFG	plane	Plane mode used by MO AGPS session	Configure Plane Mode Used by MO AGPS Session	3
15	AT+QGPSCFG	autogps	Auto run of GNSS	Enable/Disable GNSS to run automatically	3
16	AT+QGPSCFG	agnssprotocol	AGPS and AGLONASS positioning protocol	Configure AGNSs Positioning Mode	10

TABLE XIII: A full list of tested devices.

#	Device	Manufacturer	Baseband Vendor	Chipset
1	iPad Pro (1st g.)	Apple	Qualcomm	MDM 9645M
2	iPad Pro (3rd g.)	Apple	Intel	PMB 9955
3	iPhone 12 Pro	Apple	Qualcomm	Snapdragon X55
4	iPhone12 mini	Apple	Qualcomm	Snapdragon X55
5	iPhone 13	Apple	Qualcomm	Snapdragon X60
6	iPhone 6	Apple	Qualcomm	MDM 9625
7	iPhone 7	Apple	Qualcomm	MDM 9645M
8	iPhone 8	Apple	Intel	PMB 9655
9	iPhone XS	Apple	Intel	PMB 9955
10	Pixel 5a	Google	Qualcomm	Snapdragon 765G
11	Be Y	Huawei	HiSilicon	Kirin 658
12	Mate 10	Huawei	HiSilicon	Kirin 970
13	Mate 10 Pro	Huawei	HiSilicon	Kirin 970
14	Nova 3	Huawei	HiSilicon	Kirin 970
15	P10	Huawei	HiSilicon	Kirin 960
16	P20	Huawei	HiSilicon	Kirin 970
17	P20 Pro	Huawei	HiSilicon	Kirin 970
18	P30 Pro	Huawei	HiSilicon	Kirin 980
19	Y7 Prime	Huawei	Qualcomm	Snapdragon 430
20	Y9	Huawei	HiSilicon	Kirin 659
21	G3	LG	Qualcomm	Snapdragon 805
22	G6 ThinQ	LG	Qualcomm	Snapdragon 821
23	G7 ThinQ	LG	Qualcomm	Snapdragon 845
24	G7+ ThinQ	LG	Qualcomm	Snapdragon 845
25	G8 ThinQ	LG	Qualcomm	Snapdragon 855
26	K50	LG	Mediatek	Helio P22 MT6762
27	Nexus 5 (A)	LG	Qualcomm	Snapdragon 800
28	Nexus 5 (B)	LG	Qualcomm	Snapdragon 800
29	V30 ThinQ	LG	Qualcomm	Snapdragon 835
30	V35 ThinQ	LG	Qualcomm	Snapdragon 845
31	V40 ThinQ	LG	Qualcomm	Snapdragon 845
32	V50 ThinQ	LG	Qualcomm	Snapdragon 855
33	X4	LG	Mediatek	MT6750
34	X6	LG	Mediatek	Helio P22 MT6762
35	Oppo Find X	Oppo	Qualcomm	Snapdragon 845
36	A31	Samsung	Mediatek	Helio P65 MT6768
37	A9 Pro	Samsung	Qualcomm	Snapdragon 710
38	Galaxy A30	Samsung	Samsung	Exynos Octa 7904
39	Galaxy Fold 5G	Samsung	Qualcomm	Snapdragon 855
40	Galaxy Gear S3	Samsung	Samsung	Exynos 7 Dual 7270
41	Galaxy Note10 5G	Samsung	Samsung	Exynos 9825
42	Galaxy Note20 Ultra	Samsung	Qualcomm	Snapdragon 865
43	Galaxy Note 5	Samsung	Samsung	Exynos 7420
44	Galaxy Note 8 (A)	Samsung	Samsung	Exynos 8895
45	Galaxy Note 8 (B)	Samsung	Samsung	Exynos 8895
46	Galaxy Note 9	Samsung	Qualcomm	Snapdragon 845
47	Galaxy Note FE	Samsung	Samsung	Exynos 8890
48	Galaxy S10 5G	Samsung	Samsung	Exynos 9820
49	Galaxy S10 (A)	Samsung	Samsung	Exynos 9820
50	Galaxy S10 (B)	Samsung	Qualcomm	Snapdragon 855
51	Galaxy S10e	Samsung	Samsung	Exynos 9820
52	Galaxy S20	Samsung	Qualcomm	Snapdragon 865
53	Galaxy S21 5G	Samsung	Samsung	Exynos 2100
54	Galaxy S22+	Samsung	Qualcomm	Snapdragon SM8450
55	Galaxy S5 (A)	Samsung	Qualcomm	Snapdragon 801
56	Galaxy S5 (B)	Samsung	Qualcomm	Snapdragon 805
57	Galaxy S6	Samsung	Samsung	Exynos 7420
58	Galaxy S6 Edge	Samsung	Samsung	Exynos 7420
59	Galaxy S7 (A)	Samsung	Samsung	Exynos 8890
60	Galaxy S7 (B)	Samsung	Qualcomm	Snapdragon 820
61	Galaxy S7 Edge (A)	Samsung	Samsung	Exynos 8890
62	Galaxy S7 Edge (B)	Samsung	Qualcomm	Snapdragon 820
63	Galaxy S8+	Samsung	Qualcomm	Snapdragon 835
64	Galaxy S8	Samsung	Qualcomm	Snapdragon 835
65	Galaxy S9+ (A)	Samsung	Samsung	Exynos 9810
66	Galaxy S9+ (B)	Samsung	Qualcomm	Snapdragon 845
67	Galaxy S9 (A)	Samsung	Samsung	Exynos 9810
68	Galaxy S9 (B)	Samsung	Qualcomm	Snapdragon 845
69	Galaxy Z Flip3	Samsung	Qualcomm	Snapdragon 888
70	Black Shark	Xiaomi	Qualcomm	Snapdragon 845
71	K40 Gaming	Xiaomi	Mediatek	MT6893 Dimensity 1200
72	MI 5S	Xiaomi	Qualcomm	Snapdragon 821
73	MI 5S+	Xiaomi	Qualcomm	Snapdragon 821
74	MI 6	Xiaomi	Qualcomm	Snapdragon 835
75	MI 8	Xiaomi	Qualcomm	Snapdragon 845
76	MI A1	Xiaomi	Qualcomm	Snapdragon 625
77	MI MAX 3	Xiaomi	Qualcomm	Snapdragon 636
78	MI MIX 2	Xiaomi	Qualcomm	Snapdragon 835
79	MI MIX 2S	Xiaomi	Qualcomm	Snapdragon 845
80	Pocophone F1	Xiaomi	Qualcomm	Snapdragon 845
81	Redmi 10X	Xiaomi	Mediatek	MT6875 Dimensity 820
82	Redmi 5	Xiaomi	Qualcomm	Snapdragon 450
83	Redmi Note 4	Xiaomi	Qualcomm	Snapdragon 625
84	Redmi Note 5	Xiaomi	Qualcomm	Snapdragon 636
85	Redmi Note 9T 5G	Xiaomi	Mediatek	Dimensity 800U
86	Redmi S2	Xiaomi	Qualcomm	Snapdragon 625
87	Axon 7	ZTE	Qualcomm	Snapdragon 820
88	Blade V8 Pro	ZTE	Qualcomm	Snapdragon 625
89	Mudi	GL.iNet	Qualcomm	EC-25E
90	E3372h-607	Huawei	HiSilicon	Huawei HiSilicon V7R2
91	E397Bu-502	Huawei	Qualcomm	MDM9200
92	UFI-1B	UFI	Qualcomm	Snapdragon 410
93	A701	IEASUN	Qualcomm	MDM9200
94	UC120	Dinstar	Qualcomm	EC-25-E
95	ACOM508L-8 (CHN)	Ejoin	Qualcomm	EC-20
96	ACOM508L-8 (KOR)	Ejoin	Qualcomm	EC-25-E
97	MV-374	Portech	Qualcomm	EC25-G
98	SC-111	Suncomm	Qualcomm	EC25-E
99	NeoGate TG200	Yeastar	Qualcomm	EC-21
100	T Pocket-Fi	Smobile	GCT	GDM 7243
101	TL-MR6500v	TP-Link	Qualcomm	MDM 9607
102	Galaxy Tab A	Samsung	Samsung	Exynos 7 Octa 7870