Watching the Watchers: Practical Video Identification Attack in LTE Networks

Sangwook Bae, Mincheol Son, Dongkwan Kim, CheolJun Park, Jiho Lee, Sooel Son, and Yongdae Kim
Korea Advanced Institute of Science and Technology (KAIST)
{hoops, mcson, dkay, fermioncj, jiholee, sl.son, yongdaek}@kaist.ac.kr

Abstract

A video identification attack is a tangible privacy threat that can reveal videos that victims are watching. In this paper, we present the first study of a video identification attack in Long Term Evolution (LTE) networks. We discovered that, by leveraging broadcast radio signals, an unprivileged adversary equipped with a software-defined radio can 1) identify mobile users who are watching target videos of the adversary's interest and then 2) infer the video title that each of these users is watching. Using 46,810 LTE traces of three video streaming services from three cellular operators, we demonstrate that our attack achieves an accuracy of up to 0.985. We emphasize that this high level of accuracy stems from overcoming the unique challenges related to the operational logic of LTE networks and video streaming systems. Finally, we present an end-to-end attack scenario leveraging the presented video identification attack and propose countermeasures that are readily applicable to current LTE networks.

1 Introduction

An increasing number of users leverage cellular networks to watch videos on their mobile devices [18, 42]. Following this trend, an individual's video viewing history becomes private information that reveals their political, financial, and personal interests. Thus, an attacker's ability to identify videos that an individual watches poses a serious privacy threat.

Unfortunately, recent studies [20, 21, 34, 38, 43, 44] have shown that an adversary can identify videos by analyzing encrypted traffic, referred to as a *video identification attack*. The attack exploits the fact that an encrypted video stream has its own identifiable fingerprint owing to the operating logic of HTTP adaptive streaming (HAS), the leading video streaming protocol. In HAS, a video is segmented into smaller chunks, whose sizes vary according to content. Thus, a series of such chunks forms a unique pattern.

Most studies on video identification attacks have targeted wired networks [20, 21, 38, 43, 44]. They assumed a strong attack model; the adversary is required to have either 1) direct access to a victim's network infrastructure (*e.g.*, wired/wireless routers) or 2) an ability to run malicious apps or websites on a victim's device. These attacks thus require a *strong privilege* to monitor network traffic, which reduces the likelihood of the attack's success in practice.

By contrast, radio signals in wireless networks are transmitted over the air, which an adversary equipped with a software-defined radio (SDR) is able to monitor. If this adversary is able to extract effective features from these signals that contribute to identifying streamed videos, they become a strong adversary who can carry out a video identification attack without *any privilege*. Nevertheless, only a few studies have investigated the feasibility of conducting an attack in wireless networks [34].

In this paper, we present the first study of a video identification attack in LTE networks. We investigate whether, for a given cell, an adversary can 1) pinpoint users who are watching target videos of the adversary's interest and then 2) identify the exact title of the video that each user is watching. Here, we assume an *unprivileged* adversary equipped with an SDR, who has no access to victims' devices or cell towers but can sniff broadcast signals transmitted from a particular cell tower. This scenario can be exploited, for example, to track down users watching illegal or sensitive videos.

Conducting a successful video identification attack requires addressing three technical challenges that arise from the distinct characteristics of LTE networks (§3). (1) The adversary has a limited monitoring capability, which makes it challenging to determine the observed traffic type (*e.g.*, whether the video traffic is from YouTube or Netflix) and build precise video fingerprints. (2) It is non-trivial to collect a specific victim's traffic over time because the identifier, which binds traffic to the victim, often changes during video playback due to the interworking between HAS and the operational logic of LTE networks. (3) Lastly, the adversary needs to aggregate video traffic from multiple channels due to carrier aggregation (CA) logic [2, 4], which most LTE network operators employ to maximize network bandwidth.

To tackle these challenges, we propose a video identification attack that takes advantage of the following contributions (§4). (1) We identify distinctive characteristics of video streaming traffic and video service providers. These characteristics enable us to identify a video title and its service provider using convolutional neural network (CNN) and decision tree classifiers, respectively. (2) We propose to harness unencrypted information transmitted solely to a victim over the air to link the victim's changing identifiers, thus capturing the full traffic volume to each victim. (3) Instead of monitoring all channels for CA, we propose a heuristic method that estimates the volume of video traffic by monitoring only a single channel.

To evaluate the feasibility of the attack, we collected 46,810 LTE traffic traces (§5) for three major video streaming services (*i.e.*, YouTube, Netflix, and Amazon) from three operational mobile network operators (MNOs); this accounts for 2,035 hours of video involving 1.79 TB of video traffic. Using this dataset, we conducted a series of experiments considering multiple factors: service providers and video quality, MNOs and device types, changing identifiers, CA, network environments, class sizes, and unseen traffic. The experimental results demonstrate a high accuracy of 0.87, 0.95, and 0.98 for the Netflix, Amazon, and YouTube datasets, respectively.

In addition, we demonstrate an end-to-end attack scenario that selectively locates individuals who watch one of the attacker's target videos (§6). Assuming the adversary who is physically adjacent to victims, the attack is able to enforce the victims' devices to sound a loud alarm, possibly revealing the presence of victims who are watching one of the target videos. The attack targets only the victims' devices without affecting other user devices connected to a legitimate base station. To achieve this, we extended a signal injection attack [55] and chained it with our video identification attack.

We conclude by proposing countermeasures that are readily applicable to current LTE networks (§7). In particular, we propose a method that breaks the link between user identifiers and traffic information without requiring any changes to devices or the 3GPP specification.

In summary, our contributions are as follows:

- We present the first study of a video identification attack in LTE networks and concretize the challenges to realize it.
- We develop (a) a new technique of inferring CA-enabled downstream traffic volumes and (b) formulate new features to classify video streaming service providers. We also present a way to combine these techniques with existing ones to launch a successful video identification attack.
- We demonstrate that the proposed attack is a practical threat by showing a high accuracy of 0.985 using 46,810 LTE traces for three video streaming services from three MNOs.
- We suggest practical mitigations for the presented video identification attack.
- To encourage further research, we release our dataset and scripts for data collection, identification, and defense [1].

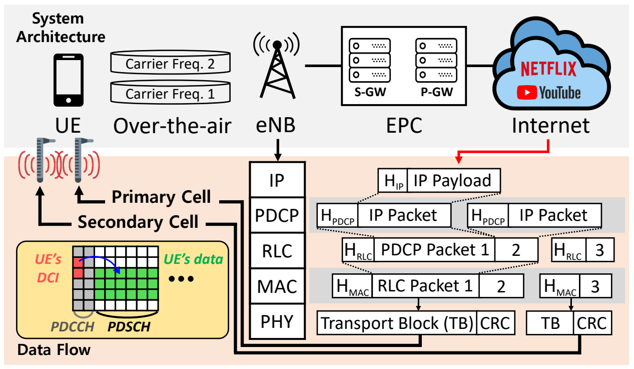


Figure 1: LTE network architecture and protocol stack

2 Background

2.1 Data Transmission in LTE

An LTE network consists of three main components: user equipments (UEs), evolved Node B (eNB), and evolved packet core (EPC). A *UE* is a device (*e.g.*, a smartphone) that provides cellular services to end-users. An *eNB* is a base station which provides a radio connection to UEs. An *EPC* refers to a core network that provides data services while managing user registration and mobility (Fig. 1).

Delivering video data from a video service provider to a UE entails the following steps. The gateways in an EPC first receive video packets from a streaming service provider. Then, they flood the video packets to the eNB, to which the UE is connected. The eNB encapsulates the received packets and transmits them to the UE over radio channels.

Identities. For proper data delivery to the designated UE, multiple identifiers are assigned to the UE. The eNB, to which the UE is connected, assigns the UE a *Radio Network Temporary Identifier (RNTI)*, and the EPC assigns the UE a *Temporary Mobile Subscriber Identity (TMSI)*. Using these identifiers enables the EPC and eNB to deliver data to the correct recipient as well as to manage per-user cryptographic keys for secure data transmission. An RNTI is a temporary yet unique identifier that the eNB assigns to the connected UE; the eNB thus differentiates the user from other connected UEs. It is only valid when the UE is connected to the serving eNB. Likewise, a TMSI is a unique identifier managed by the EPC; it is newly assigned when the UE is registered in the LTE network.

Radio Access Network (RAN). A RAN refers to a network between the UE and the connected eNB across which control messages and user data traffic are transmitted. They are delivered via an LTE protocol stack (Fig. 1). Control messages and user data traffic are encapsulated at an eNB through multiple layers. They are encrypted at the Packet Data Convergence Protocol (PDCP) layer, segmented at the Radio Link Control (RLC) layer, scheduled at the Medium Access Control (MAC) layer, and then delivered through the physical layer (PHY).

The Radio Resource Control (RRC) protocol defines how to generate control messages that manage all radio connections.

For example, when the UE requests a new connection to an eNB, the eNB responds with an `RRC Connection Setup` message to deliver wireless configuration information. When there is no traffic between the eNB and UE for a specified period, the eNB sends an `RRC Connection Release` message to remove the connection and save network resources as well as the UE's battery. When they need to communicate thereafter, the connection is established again.

Data acquisition. A fundamental feature of radio communication is that signals are broadcast to devices over a range of radio frequencies, called a channel. This channel becomes an information source from which each UE selectively retrieves its own data from broadcast signals. The physical layer instructs the UEs how to retrieve control information from a Physical Downlink Control Channel (PDCCH) and user data from a Physical Downlink Shared Channel (PDSCH).

Note that the PDCCH delivers multiple pieces of *Downlink Control Information (DCI; plural DCIs)* [5]. A DCI includes 1) assigned resource blocks in radio frames, which indicate the frequency and timing with which the UE should retrieve data, 2) a modulation coding scheme for decoding the assigned blocks, and 3) CRC bits masked with an RNTI. DCIs are transmitted on every LTE subframe (*i.e.*, 1 ms). A PDCCH is an unencrypted channel in which the DCIs of all LTE users on the same carrier frequency are exposed to the public. Thus, a single UE is able to decode every DCI on the same PDCCH.

Security and privacy in data transmission. Given that any UEs that share the same eNB can retrieve other users' data from the PDSCH, it is essential to provide secrecy and integrity protection that enables only legitimate users to access their own data. LTE networks achieve this by encrypting and checking the integrity of data at the PDCP layer. However, the headers of all protocol layers underneath the IP layer, including the PDCP layer, are not encrypted. Hence, anyone can still see header contents, which we exploited in approximating the volume of video streaming traffic.

Previous studies have addressed privacy threats in cellular networks that allow the adversary to track user identities and locations [22, 26, 45]. In this regard, an RNTI plays a role in mitigating the privacy threats, as it is a temporary identifier arbitrarily assigned to each radio connection. Because an eNB assigns multiple RNTIs to the same user over time, it becomes difficult for the adversary to track a particular user only based on a given RNTI without supplementary information.

Carrier aggregation (CA). In LTE Release 10, 3GPP introduced the CA technology [2], which simultaneously leverages multiple carrier frequencies for data transmission. It is designed to provide more bandwidth per user by assigning multiple frequency bands to the same user. An eNB often consists of multiple cells, each of which covers transmitting radio signals within a specified frequency band. The primary cell (PCell) is the first cell to which a UE connects to establish a connection and exchange control messages. In the eNB that the UE uses, all the remaining cells become secondary cells (SCells). The basis of CA technology is to use the PCell first for data transmission and later to leverage the SCells when a large amount of data needs to be delivered. CA execution is managed at the MAC layer; packets from the upper layers are multiplexed and delivered over frames of the PCell and SCells.

2.2 Video Identification Attack

The goal of an attacker is to precisely infer the video playing on a victim's device. In general, the attack targets encrypted network traffic sent to a victim's device [20, 21, 34, 38, 43, 44]. To identify a video from the encrypted traffic, the adversary mainly exploits information leakage stemming from the operational logic of HAS, which is the dominant streaming protocol used by popular streaming services. HAS segments a video into smaller chunks that share an identical playback time. Each chunk has a different size according to its content owing to variable bitrate (VBR) encoding. Thus, a series of such chunks can represent a unique pattern. Furthermore, the HAS client fetches multiple video chunks periodically with pausing, which produces so-called ON and OFF periods. By grouping a series of chunks using these periods, the adversary can build a concise video fingerprint for each target video.

In general, the attack consists of the following four steps:

(S1) Recording: The adversary chooses target videos for identification, generates network traffic for those videos by repeatedly playing the videos, and then extracts distinctive features from the observed traffic. Prior studies [20, 21, 38, 44] have leveraged the number, volume, or arrival interval of downlink and uplink packets as possible identifiable features.

(S2) Building a classifier: The adversary builds a classifier that identifies a video based on a given set of features. To implement and train this classifier, prior research has employed CNN [44], support vector machine (SVM) [20], and k-nearest neighbors using dynamic time warping (DTW) [21].

(S3) Monitoring: The adversary collects network traffic sent to a victim. To monitor the traffic, previous studies on wired networks assumed that an adversary can access the victim's device (to run their own app/website) [44] or network infrastructures that the victim uses [20, 21, 38, 43, 44]. In contrast, an adversary on wireless networks is free from these assumptions as the adversary can utilize information broadcast over the air [34].

(S4) Identification: Finally, the adversary queries the trained classifier to identify a video title given the victim's network traffic that the adversary monitored.

3 Threat Model and Challenges

Attacker's goal. We assume an adversary who selects a target cell and set of target videos. The adversary's goal is to recognize the identifiers (*i.e.*, TMSI and RNTIs) of UEs in this cell that have watched any of the target videos. Furthermore, for each identified UE, the adversary attempts to identify a video title that the UE has watched.

Table 1: Summary of the technical challenges that our study addressed compared to previous studies. (○: addressed)

Challenge	Video on wired [20, 21, 38, 44]	Video on Wi-Fi [34, 43]	App/Website on LTE [13, 29, 41, 49]	Video on LTE (Ours)	Our Approach
C1 - Monitoring traffic without privileged access	×	△†	○	○	Utilize broadcast information (§4.2) [13, 29, 41]
C1 - Handling noisy observed traffic	×	×	○	○	Leverage CNN (§4.4) [44]
C1 - Distinguishing streaming service provider	×	×	×‡	○	Formulate new features with a decision tree (§4.4)
C2 - Frequent changes of user identifiers	×	×	×*	○	Exploit the identity mapping attack (§4.2) [41]
C3 - Multi-channel data transmission	×	×	×	○	Estimate traffic volumes in SCell (§4.3)

† The authors partially addressed this challenge; they monitored victim's traffic in a wireless router. We thus mark this △.
‡ The authors distinguished app types, such as web surfing or teleconferencing; this is different from distinguishing streaming service providers.
* The authors partially introduced this challenge; their attacks do not require to address it owing to their short period of attack time. We thus mark this ×.

Attacker's capabilities. An adversary in our attack model does not require direct access to victims' UEs or the eNB. Furthermore, the adversary does not have any information about victims, including their usage (*e.g.*, traffic type, or service provider of a streaming video) and cryptographic keys for secure data transmission. Therefore, the adversary cannot decrypt any user data embodied in PDCP packets.

Meanwhile, the adversary can choose any target eNB and conduct the attack on any users attached to it. Within the radio coverage of the target eNB,[1] the adversary can eavesdrop on radio signals transmitted over the air and sniff downlink messages by leveraging an SDR [50] and publicly available software [12, 15]. By parsing the downlink messages, the adversary can obtain MAC packets, DCIs, and unencrypted PDCP header information for the attack.

Implication. The adversary determining whether specific videos are streamed solely by observing encrypted video traces has been considered a privacy threat [20, 21, 38, 44].

In particular, we argue that a video identification attack in cellular networks poses a critical privacy threat; the adversary with no privilege could affect the whole coverage area of a cell tower (1–5 km) [39], compared to the attacks in Wi-Fi networks [34] that target small areas ($\leq$ 100 m).

The attack can be exploited to track down users who watch illegal or sensitive videos in a target cell. Consider a scenario in which a law enforcement agency in an oppressive regime executes the attack. The agency (*i.e.*, adversary) builds a blacklist of videos featuring anti-government content. By conducting the attack, the adversary obtains the identifiers (*i.e.*, TMSI and RNTIs) of users who have watched prohibited videos. These identifiers are then linked to a personal identity with the help of an ISP or by further conducting identifier linking attacks [22, 25, 45]. Accordingly, the adversary can pinpoint users who have watched videos on the blacklist. Even worse, when the agency is able to reveal the locations of these users, the problem becomes critical, which we describe in §6.

Despite its severity, to the best of our knowledge, the video identification attack in cellular networks has not been studied yet. We believe that this stems from the difficulties in addressing the distinct challenges described below.

[1] A location that exhibits a high signal-to-noise ratio (over 23dB) for the eNB guarantees successful signal decoding.

3.1 Practical Challenges

Video identification attacks on LTE networks need to address the following three technical challenges: (C1) limited monitoring capability, (C2) frequent changes of user identifiers, and (C3) multi-channel data transmission. Note that C1 and C2 have been considered partially in different attack scenarios [13, 29, 41, 49]. However, none have investigated these challenges for the video identification attack, which demand additional considerations. Furthermore, the video identification attack in LTE networks introduces another unique challenge of C3 due to CA. Tab. 1 summarizes the challenges addressed in previous studies and our study.

C1: Limited monitoring capability. The adversary cannot decrypt encrypted user traffic delivered over the PDSCH. Therefore, the adversary needs to utilize only public information. By mimicking the data acquisition procedure of the victim's UE (§2.1), which decodes DCIs embodied in broadcast signals, the adversary can obtain a time series of packet volumes [13, 29, 41, 49]. They then use this information as a pattern for each video.

However, the size information extracted from DCIs is not consistent even when the same video is streamed, which impedes accurate video identification. The computed packet volumes often account for traffic other than video content, such as (1) retransmitted packets at the RAN layers including RLC and PDCP, depending on the network condition; (2) LTE control plane messages; and (3) packets from other apps running on the victim's UE. Moreover, occasional decoding failures of the downlink sniffer (*e.g.*, AirScope [12] or OWL [15]) contributes to obscuring the actual size of streamed chunks.

In our dataset (*i.e.*, `YouTube100` in §5.1), we measured the discrepancy between the estimated and actual volumes of streamed video chunks. We compared the estimated video size by decoding DCIs with the actual size of the streamed chunks at the IP layer. Depending on videos and LTE traces, the difference between the estimated and actual volumes ranged from -1.1 to 9.6% with an average of 7%. Therefore, the adversary has to build a robust classifier to handle inconsistent volumes for accurate video identification.

Furthermore, for accurate identification, the adversary should consider an underlying video service provider, which is a dominant factor in varying streaming traffic patterns. We

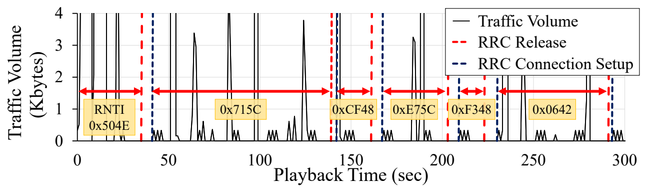


Figure 2: UE's changing RNTI during a single video stream

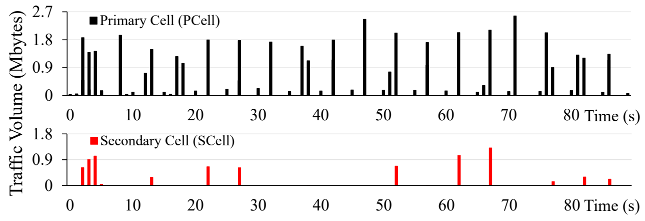


Figure 3: UE's streamed video volume using 2-band CA

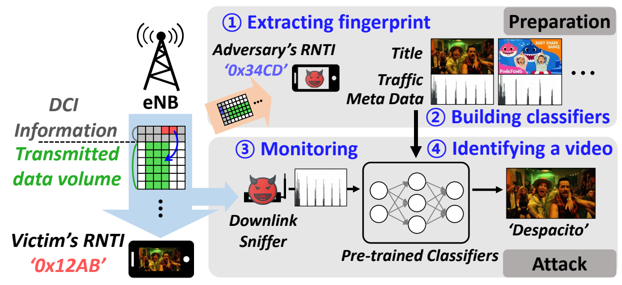


Figure 4: Overview of our video identification attack

observed that service providers affect the estimation of a traffic volume in a CA-enabled environment (§4.3). As the TCP/IP layer is encrypted, the adversary cannot utilize known IP addresses to identify the service providers.

C2: Frequent changes of user identifiers. DCIs are public information, which we use to build a fingerprint for each video. As users retrieve their own DCIs with their own RNTIs (§2.1), an adversary can follow the same step to collect DCIs for each user. The challenge here lies in the operating logic of HAS video streaming, which triggers frequent changes of these RNTIs. HAS is designed to enter a paused period, called the OFF period, after a UE fetches each video chunk(s) (§2.2). At the same time, the UE is supposed to release its connection to an eNB when there is no traffic for a certain period to save resources. Consequently, when watching a HAS video, the UE releases its connection during the OFF period. When the UE fetches the next video chunks, it reestablishes the connection to the eNB and is then assigned a new RNTI. As watching a HAS video produces multiple ON-OFF periods, the adversary must track various RNTIs for each user to collect a non-fragmented series of user's DCIs.

RNTIs may change differently in practice depending on the configuration of streaming service providers and MNOs. We investigated three major streaming services (*i.e.*, YouTube, Netflix, and Amazon) and three major MNOs. We observed that the OFF period of Netflix's HAS service is configured to be about 40 s while the OFF periods of YouTube and Amazon are about 5 s and 4 s, respectively. Meanwhile, in all MNOs, we found that their eNBs disconnect UEs when they have no traffic for 10 s. Thus, when a UE watches a Netflix video, its RNTI changes at least once every 40 s in all MNOs.

Fig. 2 shows the time series of received traffic volume and the changing RNTIs when a client plays a Netflix video. We played *Sherlock (Season 1, EP. 1)* on a Galaxy S6 Edge smartphone. During the 300 s playback, the client repeatedly released its radio connection during the OFF period and reestablished the connection six times. For each reconnection, a new RNTI was assigned to the device. To conduct video identification for any streaming services with an OFF period of over 10 s, the adversary should address this challenge.

C3: Multi-channel data transmission. An eNB activates CA to deliver a large volume of data over multiple channels, thus boosting the transmission bandwidth (§2.1). Given that a typical video stream comprises of several chunks having a large volume, the eNB to which a victim's UE connects is highly likely to activate CA when transmitting video content in practice. This means that the adversary needs to take into account the sniffing of CA-activated traffic over multiple channels simultaneously.

To examine the impact of CA on data transmission, we captured cellular traffic while playing the song *Despacito* on YouTube on a Galaxy S6 Edge using 2-band CA. Fig. 3 shows traffic traces that represent the size of transmitted data over time. In the figure, the black and red bars represent data traffic flooded through the PCell and SCell, respectively. The amount of data from the SCell comprises approximately 14% of the total received data. For the specific time window of between 4 and 4.5 s, the amount of SCell data accounts for 42% of all data. Therefore, for a precise video fingerprint, the adversary should consider both the PCell and SCell.

4 Video Identification Attack

4.1 Attack Overview

We present a video identification attack in LTE networks addressing the practical challenges (§3.1). Fig. 4 illustrates the overall attack procedure: 1) the attacker selects target videos and computes their fingerprints; 2) the attacker prepares the attack by building a classifier that identifies the target videos; 3) the attacker then performs the attack on a target cell, computing a tuple (a TMSI, the chain of RNTIs assigned to the TMSI, and a video title) for each UE in the target cell.

The adversary starts by playing target videos repeatedly on their own UE and obtains a time series of transmitted data volumes by decoding DCIs. In this step, they retrieve their own RNTIs from diagnostic monitoring tools [23, 54] connected to their UE and disable CA by configuring their UE to use only one band; thus, they are free from C2 and C3. Next, the adversary builds their classifier that takes the collected traces as input. As the traces from the same video differ, the adversary should build a robust classifier against inconsistent input traces (C1). To address this, we adopt a CNN classifier, which is known to be robust against unsteady input patterns with noise [30,44] (§4.4). For accurate identification, we set up a decision tree classifier that infers a video service provider of which HAS operational logic differs by vendors. We use this classifier to choose one among the CNN classifiers, each of which is trained for a video service provider.

Subsequently, the adversary executes the attack on a target eNB by monitoring the radio traffic of all UEs connected to the eNB using a downlink sniffer [12]. To obtain complete traces for each UE, the adversary tracks their frequently changing RNTIs (C2) and estimates the volume of missing packets due to CA (C3). To address C2, we fully utilize broadcast information delivered over the PDCCH and PDSCH (§4.2). For C3, we propose two approaches: one that exploits unencrypted header information and another that exploits a vulnerable RAN design (§4.3). The adversary then feeds the collected traces to the pre-trained classifiers and infers the video titles.

4.2 Utilizing Broadcast Information

The benefits of leveraging broadcast radio signals are twofold; the adversary can (1) extract traffic patterns from DCIs and (2) track the frequently changing RNTIs.

Extracting traffic patterns. The adversary listens to the PDCCH of the target eNB to which victim UEs are connected. They retrieve DCIs within an LTE subframe (*i.e.*, 1ms) and compute the volume of transmitted data for each DCI by mimicking the data acquisition procedure of the victim UEs. Specifically, the adversary decodes DCIs using the coding rate of the specified modulation coding scheme and the number of assigned resource blocks. By undertaking this over time, they obtain a time series of packet volumes over multiple DCIs. Finally, for each non-overlapping time window of 0.2 s, the adversary aggregates the computed volumes and generates a vector that encodes the downlink traffic pattern for each UE.

Although the use of DCIs has been proposed in previous studies [13,29,41,49], none of them have explored its application to the video identification domain. It should be noted that leveraging DCIs alone is insufficient for accurate video identification, which necessitates tracking the frequently changing RNTIs of UEs and addressing CA.

Identifier tracking. We propose linking the frequently changing RNTIs of each victim UE to its unified identifier, TMSI, using an identity mapping attack [41]. The identity mapping attack exploits the radio connection procedure of the LTE. According to the LTE specification [7], to establish a new radio connection to an eNB, a UE first sends a random access request to the eNB and receives a response that contains a temporal RNTI (T-CRNTI). Using this T-CRNTI, the UE then decodes the `RRCConnectionSetup` message broadcast over the PDSCH in plaintext; this message embodies the UE's TMSI.

The adversary mimics this procedure to track the RNTIs of each victim UE with its TMSI. They monitor all the responses from the eNB and collects all T-CRNTIs. With each collected T-CRNTI, the adversary decodes the sniffed `RRCConnectionSetup` message and obtains the embodied TMSI. They then match the TMSI to the RNTI used to decode the message. After this procedure, the adversary obtains multiple pairs of a TMSI and an RNTI. As TMSIs rarely change,[2] the adversary can map multiple RNTIs to their corresponding TMSIs, consequently obtaining a tuple (a TMSI, the chain of RNTIs for the TMSI). Using this information, they keep track of each UE's traffic.

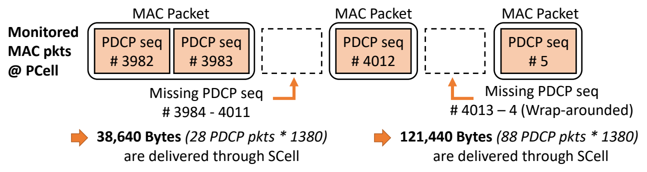

MAC Packet
MAC Packet
MAC Packet
Monitored MAC pkts @ PCell
PDCP seq # 3982
PDCP seq # 3983
PDCP seq # 4012
PDCP seq # 5
Missing PDCP seq # 3984 - 4011
Missing PDCP seq # 4013 – 4 (Wrap-arounded)
38,640 Bytes (28 PDCP pkts * 1380) are delivered through SCell
121,440 Bytes (88 PDCP pkts * 1380) are delivered through SCell

Figure 5: Example of estimating lost traffic volume in CA

4.3 Addressing Carrier Aggregation

CA is often activated when an eNB transmits a large number of video chunks to a UE. This tendency necessitates that the adversary captures traffic delivered over multiple channels. For this, we propose two approaches: 1) adopting multiple downlink sniffers and 2) estimating the volume of missing traffic with one downlink sniffer.

Adopting multiple sniffers. One straightforward solution is to deploy multiple downlink sniffers at each channel of the PCell and SCells. To measure traffic volume over the SCells, the adversary must know (1) the RNTI used for communication between a victim's UE and each target SCell and (2) the activated SCells for video streaming.

We propose leveraging the limitations of the standard CA design and the current status of SCell deployment by MNOs. (1) By design, the operational logic of CA uses the same RNTI for both the PCell and SCells. Therefore, the adversary can identify a victim's traffic in the SCells by leveraging the same RNTI in the PCell. (2) MNOs do not select random SCells for CA activation. Usually, an MNO is assigned with a limited number of available frequency bands. The information regarding active frequency bands and CA configurations for major MNOs is pre-defined [19] and open to the public [16].

Although this simple solution is promising, it has a limitation. The proposed solution demands multiple SDRs, one SDR for each activated SCell, thus increasing its deployment cost. Considering that the 3GPP standard supports 31 SCells and 5G uses more SCells, this simple solution is not scalable for conducting the proposed attack in practice.

Estimating traffic volume. We propose a novel method that enables the adversary to estimate the traffic volume delivered by the PCell and SCells simultaneously with only one SDR device. The method mainly exploits (1) sequence numbers in PDCP packet headers, which are not encrypted during data

[2] A TMSI rarely changes, even when a radio connection is reestablished after a UE reboots [45].

transmission, and (2) the fixed size of video streaming packets.

Fig. 5 illustrates an example of estimating a missing volume of traffic. The adversary starts by monitoring a series of PDCP sequence numbers from the PCell. By decoding transmitted data over the PDSCH, the adversary can obtain packets at the MAC layer and extract PDCP packets along with their headers. When there is a missing sequence number obtained from the PCell, the adversary knows that those packets are delivered over one of the SCells. In general, we count each gap between two discontinuous PDCP sequences and sum those gaps to compute the total number of missing packets.

Now, the adversary estimates the missing traffic volume delivered over SCells under the following two assumptions, which we empirically confirmed via manual traffic analysis.

(1) Each missing PDCP packet is fully packed with video data, and its packet size is fixed according to the maximum transmission unit (MTU) configured by a streaming service provider or an MNO. For Netflix and Amazon, we observed that the size of PDCP packets was fixed at an MTU varying by MNO, which is 1,440, 1,450, and 1,430 bytes for MNOs A, B, and C, respectively. On the other hand, for YouTube, the PDCP packet size was fixed at 1,380 bytes across all three MNOs. This is mainly due to QUIC [32], which is a transport layer protocol employed by YouTube for streaming. QUIC has an MTU of 1,350 bytes, which is smaller than the MTU configured by the MNOs. Thus, the size of PDCP packets in YouTube streaming is fixed at 1,380 bytes, including the 30-byte header.

(2) A PDCP sequence number is a 12-bit number that all three major MNOs use. As the MNOs adopt a fixed size and wrap-around logic for PDCP sequence numbers, the adversary can count the missing PDCP packets in the case that the PDCP sequence number has a lower value than that of the previous PDCP packet. Finally, the adversary calculates the lost traffic volume by multiplying the number of missing PDCP packets by the MTU size of the QUIC protocol. Note that in the case of a video service that does not use QUIC, the adversary calculates the size of missing packets by using the PDCP MTU of a target MNO.

One may argue that the proposed estimation method is unnecessary when an MNO employs a *cross-carrier scheduling* policy. In this policy, DCIs from the PCell contain the resource allocation information for each SCell, thus rendering obsolete the need for estimating the traffic volume over the SCells. However, according to our investigation into three major MNOs, none currently deploys this policy, making the estimation step essential for accurate identification.

This estimation process can be affected by two configurations: 1) the length of the PDCP sequence number, and 2) the adoption of robust header compression (ROHC) [8, 40]. For the former, the length of the PDCP sequence number varies over MNOs and data radio bearers. However, the adversary can easily obtain this length with their own UE before launching the attack by monitoring `RRC Connection Reconfiguration` messages. In practice, we observed that all MNOs use 12 bits for this length. For the latter, ROHC might affect the packet size due to its header compression on the IP/UDP layer. However, we observed that three MNOs use ROHC only on voice calls, not on video streaming.

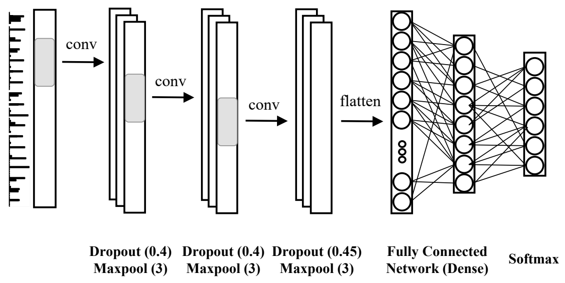

conv
conv
conv
flatten
Dropout (0.4) Maxpool (3)
Dropout (0.4) Maxpool (3)
Dropout (0.45) Maxpool (3)
Fully Connected Network (Dense)
Softmax

Figure 6: The structure of the used CNN model

4.4 Building a Classifier

The proposed attack involves two classifiers: one for identifying the video service provider, and the other for identifying the video title. The adversary prepares a video title classifier for each service provider, and the service provider classifier determines a video title classifier for a given set of traffic features.

Video service provider classifier. The first classifier is designed to identify the video service provider to which given traffic features belong. This classifier plays an important role in handing CA to derive accurate traffic volumes (§4.3). We leverage a decision tree with nine features extracted empirically from the traffic analysis: the number of chunks, the duration of the first ON/OFF period, and the average/maximum/minimum duration of ON/OFF periods. These nine features are directly related to the HAS configuration of each video service provider.

We observed that this HAS configuration highly affected the shape of video traffic (See App. C): 1) an initial playback buffer size affects the duration of the first ON/OFF period; 2) the duration of each chunk affects the OFF period duration; and 3) the logic of fetching video chunks affects the ON period duration. These features are different from the features of previous approaches on fingerprinting application types [13, 49] (*e.g.*, web surfing or teleconferencing). Our features are agnostic to traffic volumes and designed to capture the differences in the shape of video traffic caused by HAS configuration.

Video title classifier. We leverage a 1D-CNN 1) to capture traffic-level commonalities in video streaming for a given video title, even in the presence of traffic variations, and 2) to use raw traffic metadata for the input of the classifier. CNNs are effective for learning the commonalities of a given time series data via a convolution filter, which ensures that the generated model is robust to noise [30, 44].

Fig. 6 shows the structure of our CNN model. The model consists of three convolution layers, each of which accompanies a maxpool and dropout layer, and two dense layers.

Table 2: Dataset summary (a total of 46,810 traces for 2,035 h)

Dataset	# of Videos	# of Traces	Trace Length (s)	Description	Usage
`YouTube100`	100	29,715	120	YouTube Top 100	All
`Netflix`	22	1,001	800	Netflix Top 50	§5.2, §5.5
`NetflixDT`	31	298	800	Netflix (Random)	§5.2, §5.5
`Amazon`	32	1,210	120	Prime Video (Random)	§5.2
`AmazonDT`	36	310	120	Prime Video (Random)	§5.2, §5.5
`YouTubeCA`	100	7,383	120	YouTube Top 100 (CA)	§5.3
`YouTube200`	200	6,424	120	YouTube Top 101-300	§5.5
`Web`	-	268	120	Alexa Top 50	§5.5
`Teleconf`	-	201	120	Google Meet	§5.5

As an input to the model, we feed in a vector representation of the time series of estimated traffic volumes, obtained during the traffic monitoring (§4.2). Each of the vector elements represents an aggregated traffic volume for 0.2 s (*i.e.*, 200 LTE subframes). We empirically set each convolutional layer to use 150 filters, a ReLU activation function, and a kernel covering 20 vector elements (*i.e.*, 4 s of aggregated traffic volumes), considering the size-variation tendency of video chunks. We set the dropout rate and pool size to 0.4 and 3, respectively (Fig. 6). The two dense layers successively output 500 units and the number of video titles. For the other hyper-parameters, we used the default values of Keras [17].

5 Evaluation

We demonstrate our video identification attack in LTE networks from three major MNOs. We describe our dataset and experimental setup (§5.1) and present the attack performance in identifying a victim's video in a known set (§5.2), called a *closed-world* setup. We then evaluate the impact of handling CA for accurate identification (§5.3). We investigate the impact of network congestion by comparing the efficacy of our model with that of other classifiers (§5.4). We further evaluate our methodology with unseen videos as well as various mobile apps in an *open-world* setup (§5.5). In summary, we evaluate the degree to which our attack is affected by the following factors: (1) service provider and video quality, (2) MNO and device, (3) changing RNTIs, (4) CA, (5) network environment, (6) class sizes, and (7) unseen traffic.

5.1 Experimental Setup

Dataset. We constructed the dataset by sniffing LTE signals from three major MNOs with commercial UEs, as listed in Tab. 2. We collected a total of 46,810 data traces from YouTube, Amazon, and Netflix including different experimental settings. We also collected 469 additional traces for websites and teleconferences to show the robustness of the attack. The total dataset accounts for 2,035 hours of streaming time and 1.79 TB of video traffic. We describe the details of each dataset in their usage along with our experimental results.

Data collection procedure. For collecting data, we conducted the following three steps. (1) We set the UEs to connect to a commercial LTE network operated by one of the three MNOs. (2) We then launched a downlink sniffer, which listens

Table 3: Evaluation results on the `YouTube100` dataset.

MNO	Video Quality	# of Traces	Acc.	F_1	AUC	MNO	Video Quality	# of Traces	Acc.	F_1	AUC
A	480p	3,184	0.970	0.967	0.997	C	480p	3,262	0.981	0.961	0.996
A	720p	3,196	0.968	0.965	0.998	C	720p	3,218	0.967	0.965	0.997
A	1080p	3,645	0.985	0.986	0.998	C	1080p	3,293	0.957	0.954	0.997
B	480p	3,376	0.928	0.922	0.995	A	mixed	10,025	0.977	0.976	0.998
B	720p	3,318	0.973	0.970	0.997	B	mixed	9,917	0.958	0.958	0.997
B	1080p	3,223	0.940	0.937	0.996	C	mixed	9,773	0.980	0.979	0.998

to the same cell to which the current UE is connected. For this, we used two types of SDR: USRP B210 [50] and X310 [51], equipped with the downlink sniffer software, AirScope [12]. AirScope enabled us to get all MAC layer packets, transmitted by the cell, into a PCAP formatted file. (3) Finally, we recorded the downlink traffic as well as the changing RNTIs of each UE, enabling us to link fragmented traffic with the UE's TMSI (§4.2).

Target videos. We selected the most viewed music videos on YouTube [1] with lengths under 6 min each. As YouTube provides multiple streaming quality options, we set our UEs to play 120 s of each video at three different resolutions: 480p, 720p, and 1080p. We used a premium account for all YouTube datasets; thus, the video traces did not contain any ads. Note that this setup is the same as that in previous studies [20, 34, 38, 43, 44]. We further discuss this issue in §8.

We also selected 22 Netflix movies from the best 50 movies list and 32 random Amazon movies [1]. We then collected the first 800 s and 120 s of data traces for each video from Netflix and Amazon, respectively. Since Netflix and Amazon provide no option for users to select the video quality, we collected traces with the default option for each video. To examine the effect of MNO and video quality, we mainly used YouTube traffic, which is the largest dataset. To collect multiple traces for each video, we played each video over 27 times. Note that the number of traces differs because it takes a considerable amount of time to stream videos in practice. We collected the datasets at our best using our automated collection tool [1].

Operating networks. Each MNO may have a different operational policy for managing downlink resources and network operations. To investigate the impact of MNOs on successful video identification, we chose three major MNOs to construct the `YouTube100` dataset (Tab. 3). We collected traces when the eNB cells of the MNOs had a 5.9–99% downlink utilization and 1–113 active UEs.

UE. We used two UEs, Galaxy Note5 and Galaxy S6 Edge, which support 3-band CA.[3] To measure the degree to which CA impedes classifying target videos, we collected two versions of data traces for each video: one with CA-deactivated (`YouTube100`) and the other with CA-activated (`YouTubeCA`). When collecting the other datasets, we deactivated CA for precise evaluation in a controlled environment.

Classifier setup. We implemented two classifiers (§4.4). For

[3]The latest AirScope (19.09) does not support the 256QAM modulation scheme (3GPP release 12) [36]. Thus, we choose the above two UEs that support under 256QAM (*e.g.*, QPSK, 64QAM) for the following experiments.

Table 4: Impact of class size on the accuracy, F_1, and AUC.

Class Size	50	100	150	200	250	300
Acc.	0.994	0.986	0.985	0.980	0.978	0.978
F_1	0.993	0.985	0.985	0.979	0.980	0.977
AUC.	0.998	0.998	0.998	0.998	0.999	0.999

the CNN classifier, we used Keras [17] with a TensorFlow backend [11]. In each experiment, unless specified, we trained this classifier for 50 epochs with the Adam optimizer [28] and a batch size of 32. For a decision tree classifier, we leveraged Python scikit-learn [37]. We set the depth of the tree to nine, which is the number of input features. The time to train these classifiers took at most 129 s for the largest dataset (*i.e.*, `YouTube100`) using a machine equipped with an Intel Xeon E5-2630 CPU and a GTX 1080 GPU.

5.2 Closed-world Classification

We designed the experiments to answer the following four research questions: (1) Is our classifier indeed able to identify target videos using broadcast information alone? (2) How much do MNOs and devices affect the classifier's performance? (3) What is the degree to which class sizes affect the classifier's performance? (4) How much does the RNTI changing affect the classifier's performance? We answer these questions by evaluating our CNN classifier. For evaluation metrics, we conducted five-fold cross-validation and averaged the measured accuracy, area under the curve (AUC), and F1 score for each trial.

Baseline results. For each combination of the three MNOs (A, B, and C) and three streaming resolutions (480p, 720p, and 1080p) in the `YouTube100` dataset, we conducted a separate experiment. We built a separate classifier for each combination. Tab. 3 lists the classifier's performance for each experiment.

The accuracy of our classifier varied between 0.928 and 0.985 across the experiments, demonstrating that *our attack is effective in identifying encrypted video streaming transmitted over LTE networks*. These results are consistent with the prior experimental results in wired networks, which yielded an accuracy of 0.99 on 18 YouTube videos [44]. Thus, we have confirmed that a time series of the victim's encrypted traffic volumes is effective when identifying videos, even in cellular networks.

The three rows at the bottom right of Tab. 3 show the classifier's performance on the mixed dataset in which each video class has all three video resolutions. For all three MNOs, the classifier achieved an accuracy between 0.958 and 0.980. This means that the adversary does not need to accurately infer the video resolution that a target victim uses for watching YouTube videos. The adversary can instead use a mixed dataset to classify the victim's videos without losing accuracy.

Impact of MNOs and devices. To investigate the impact of MNOs, we evaluated the identification accuracy by using train and test sets, whose traces were recorded for different MNOs. The performance of all six train/test combinations showed an accuracy of 0.88–0.95, meaning that the performance of the attack is independent of the MNOs. To confirm the impact of devices and app versions, we compared the captured traces of the same video title streamed by two devices and three app versions in dataset construction (§5.1). We did not observe any significant changes due to device/software differences.

Table 5: Identification accuracy for various sniffing times

Sniffing Time (s)	Epoch	15	20	40	60	90	120
YouTube (MNO A)	50	0.78	0.87	0.95	0.96	0.97	0.98
Amazon	500	0.28	0.51	0.87	0.92	0.93	0.95
Sniffing Time (s)	**Epoch**	**40**	**60**	**120**	**200**	**400**	**800**
Netflix	50	0.24	0.37	0.46	0.57	0.75	0.87

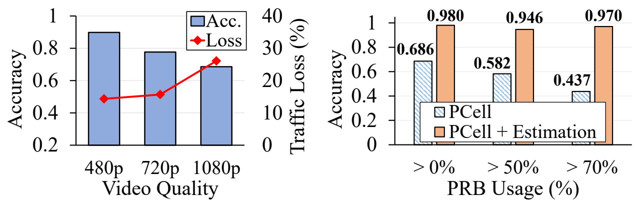
Impact of class sizes. As the proposed attack leverages machine learning techniques for video identification, it could lead to a trade-off between the number of video titles in the target list and the identification accuracy. To investigate the impact of a class size, we expanded the dataset to include 300 video titles. Specifically, we collected 10,069 traces of the top 300 music videos on YouTube in 1080p using MNO A; this consists of 3,645 traces from the `YouTube100` dataset, and 6,424 traces from the `YouTube200` dataset.

Tab. 4 shows the identification accuracy across the datasets of class sizes ranging from 50 to 300. Considering that the videos contain similar content (*i.e.*, music videos), the performance loss in accuracy is relatively small (1.6 %), demonstrating that the attack is scalable.

Impact of changing RNTIs. We investigated the degree to which changing RNTIs contributes to reducing the video identification accuracy. For this, we measured the accuracy of our classifier while varying the sniffing duration of LTE traffic. Unless the adversary handles changing RNTIs, they only have a short time window to sniff the victim's radio signals before the victim's RNTI is changed. By contrast, if the adversary tracks the RNTIs, they can compute a non-fragmented series of traffic volumes for a longer time window.

For evaluation, we trained and tested the classifiers with the first t time series of traffic volumes, where t represents the sniffing time. We varied t from 15 to 120 s for the `YouTube100` and `Amazon` datasets, and from 40 to 800 s for the `Netflix` dataset, considering the different OFF periods of each service.

Tab. 5 shows the identification accuracy depending on the sniffing time. In the `YouTube100` dataset, we observed that the accuracy degraded between 3% and 8% when the sniffing time spanned less than 40 s. For example, the accuracy of the classifier for MNO A dropped from 0.946 to 0.782 when the sniffing time was reduced from 40 s to 15 s. Interestingly, sniffing for a single minute was sufficient to achieve a robust accuracy of over 0.958. As the RNTIs of our UEs did not change within the first 2 min of all YouTube traces, the adversary was able to achieve high accuracy even with a short period of observed traffic before the victim's RNTI was changed. We made a similar observation regarding the `Amazon` dataset. In all Amazon traces, the UE's RNTI did not change because the OFF period did not last over 4 s, which is



(a) w/o considering CA (b) 1080p results

Figure 7: Identification accuracy on the `YouTubeCA` dataset.

less than the duration (10 s) of the MNOs for releasing radio connections due to no traffic.

In contrast, in the `Netflix` dataset, the classifier required a monitoring window of at least 800 s to achieve an accuracy of 0.866. During this 800 s window, we observed that RNTIs were changed 6–37 times, which makes it essential to track the RNTIs. In particular, 41% of traces in the dataset show that the UE's RNTI changed within the first 45 s. On average, the first RNTI change occurred in 69.2 s. This implies that an adversary without considering changing RNTIs can achieve an accuracy of below 0.251 when using trace data in which the RNTI is changed within the first 45 s. These results mean that *tracking RNTIs is crucial for the successful identification of videos served with a long-lasting OFF time in LTE networks*.

We further investigated the root causes of the performance difference for a short time window between YouTube and Netflix. We observed that a Netflix client fetched video chunks every 40 s and produced only two ON-OFF periods within the first minute. Meanwhile, in the YouTube dataset, we observed 10–11 ON-OFF periods within the first 20 s. Note that our classifier learns the series of traffic volume changes that these ON-OFF periods generate. Therefore, the more ON-OFF periods there are, the more distinctive the fingerprint becomes. Inevitably, the attack for Netflix videos demands a longer sniffing time to have more ON-OFF periods.

Comparison to prior work. We achieved comparable performance to prior work [44] on the YouTube dataset in a wired network while we used target video titles having over five times of the previous work. However, the performance on the Netflix dataset was relatively low. To determine the cause, we analyzed the working logic of Netflix clients in wired and LTE networks. From the analysis, we observed that the client in wired networks (*i.e.*, Chrome) periodically fetches video chunks every 10 s. This fetching period is much shorter than that of the mobile application (40 s). This means that the number of ON-OFF periods monitored in LTE networks is much lower than that in wired networks during the same monitoring time. Consequently, the classifier is fed only a few occurrences of ON-OFF periods, leading to a lower chance of extracting distinctive streaming patterns.

5.3 Handling Carrier Aggregation

We evaluate how much handling CA contributes to improving the efficacy of the proposed attack.

Impact of CA. To demonstrate the impact of CA on the identification accuracy, we ran experiments on the `YouTubeCA` dataset collected using a CA-enabled UE. This dataset consists of 1,859, 1,777, and 3,747 traces in MNO A for the three video qualities (480p, 720p, and 1080p), respectively. For each streaming resolution, we merged the traces in the `YouTube100` dataset and trained our CNN classifier. Then, we evaluated the classifier on the `YouTubeCA` dataset.

Fig. 7a shows the identification accuracy for the three video qualities when the adversary monitors the PCell without considering CA. When the UE received videos of 480p and 720p resolutions, the accuracy was 0.898 and 0.777, respectively. For both resolutions, we observed that the differences between the actual volume of streamed video chunks and the estimated volume calculated based on the DCIs are less than 14.4% and 15.7%, respectively. Particularly, when the UE receives high-quality videos (1080p) through CA, the adversary who only listens to the PCell loses 26.1% of the data volume on average. This results in a lower accuracy of 0.686. These results demonstrate that handling CA becomes essential to boost the accuracy in classifying higher resolution videos.

Improvement through our approach. We compared the identification accuracy of two attacks: 1) monitoring only the PCell without considering CA and 2) monitoring the PCell with CA estimation (§4.3). As shown in Fig. 7b, the proposed approach of estimating the lost traffic volume with one SDR improved the accuracy by 29.4% on the 1080p dataset (0.980).

In practice, the decision to use CA for data transmission is affected by network conditions as well as by the amount of transmitted data. To demonstrate the effectiveness of our solutions for handling CA, we additionally constructed two datasets from the `YouTubeCA` dataset, reflecting a congested network environment. Each dataset consists of the traces monitored when the usage of allocated downlink Physical Resource Block (PRB) is above 50% and 70% in the 1080p resolution dataset, wherein a higher value represents a higher load of the cell. The underlying assumption here is that CA is likely to be activated when the network is already congested.

Fig. 7b shows the identification accuracy of an adversary considering CA. The results for the dataset having a PRB usage of over 70% show an accuracy of 0.970. In contrast, the accuracy without considering CA for the same dataset drops to 0.437. These results demonstrate that *handling CA with packet volume estimation becomes even more crucial for successful video identification attacks as the underlying LTE network becomes congested*. Note that the estimated volume using the proposed approach differs by only 1% from the actual volume measured at the application layer, showing that our method successfully estimated a missing traffic volume.

5.4 Impact of Network Environment

We investigate the impact of network congestion on identification performance. For this, we evaluate our CNN classifier with the intuitive classifier, dynamic time warping (DTW), on the `YouTube100` dataset. This is particularly because the DTW

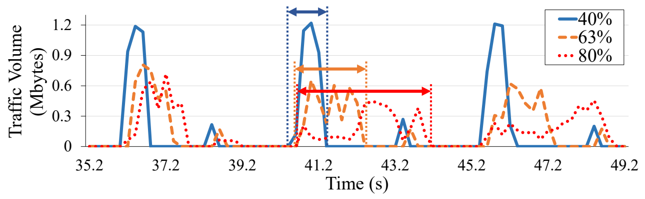


Figure 8: Impact of PRB usage to the traffic shape

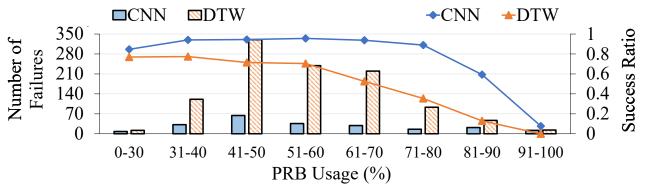


Figure 9: Identification accuracy of two classifiers depending on PRB usage (1080p video dataset in MNO B)

classifier directly reflects and observes streaming traffic. After a comparative analysis of the failure cases from the DTW and CNN classifiers, we observed two patterns: the CNN classifier effectively handles a lagged traffic shape of the network environment, and it shows robustness in a congested network environment compared to that of the DTW classifier.

First, we observed that the network environment significantly affects the shape of monitored traffic and eventually reduces the performance of the classifier. Fig. 8 shows three traces of the same YouTube video (*Despacito*) captured in various network environments with different PRB ratios (40%, 63%, and 80%). When the network becomes congested, the monitored traffic shape lagged. For example, the traffic transmission time for a chunk at an 80% PRB ratio required 3.4 times the transmission time for that at 40% (Fig. 8). Such different traffic shapes for one video affect the performance of classifiers. By design, DTW is not robust to accommodate different shapes of time-series traffic from one source because it computes the traffic similarity by direct comparison. That is, a DTW classifier trained on a 40% PRB trace only excels at classifying data traces at 40% PRB. On the other hand, the CNN classifier has a convolution filter, which inherently accounts for traffic chunks lagged over a long time window.

Second, to evaluate the robustness of the classifiers across network environments, we counted the number of failures in the identification of the two classifiers at various PRB ratios (Fig. 9) using the 1080p traces collected for MNO B in the `YouTube100` dataset. These traces can effectively show the congested network environment as MNO B has the largest number of users among the three MNOs. As expected, the DTW classifier was highly affected by the network environment, showing an accuracy lower than 0.6 when the PRB ratio was over 60%. On the other hand, our CNN classifier has 7.6 and 5.8 times fewer failure cases for traces with 61–70% and 71–80% PRB ratios, respectively. Thus, we confirmed that *the CNN classifier is robust to the cellular network environment, where congestion affecting a traffic shape frequently occurs.*

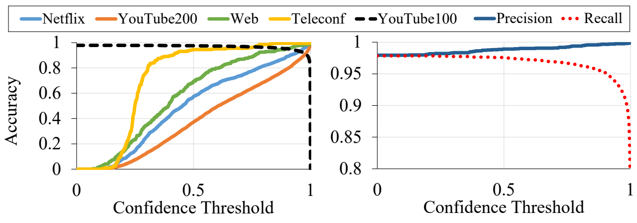


Figure 10: Identification accuracy for datasets in Tab. 2 (left) and precision/recall for YouTube known traces (right)

5.5 Open-world Classification

In practice, the downlink traffic of a victim UE includes a mixture of multiple videos from different service providers as well as non-video data. Moreover, the victim may not watch a video in the pre-selected list. Considering these practical scenarios, we examine whether the proposed attack is able to detect videos from unseen traffic of various data types.

Identifying the video service type. We demonstrate the performance of the video service type classifier (*i.e.*, decision tree classifier) on the three major services. For a given trace, the adversary has to know the video service type for handling CA and properly feeding it to the pre-built video title classifier. We trained a classifier on each of the three datasets (*i.e.*, `AmazonDT`, `NetflixDT`, and `YouTube200`) and evaluated the classifier on the datasets that do not have overlapping video titles; `YouTube100`, `Netflix`, and `Amazon`. This implies that all test traces are *unseen* to the classifier. The performance of each test set shows an accuracy of 0.980–0.991, meaning that each service provider employs a distinguishable HAS logic.

We further analyzed the misclassified cases. For YouTube, we observed that most incorrect cases were misclassified as Amazon (0.013) rather than Netflix (0.004). This is because the traces of YouTube and those of Amazon have more features with similar values (*e.g.*, number of chunks or average ON/OFF period) than those of Netflix.

Identifying the video title. We next evaluated the video title classifier (*i.e.*, the CNN classifier) with *unseen* traffic. We prepared three types of traces recorded by 1) streaming video titles that were not in the target list but were provided by the same content provider (`YouTube200`), 2) streaming videos from other content providers (`Netflix`), and 3) web-browsing (`Web`) and teleconferencing (`Teleconf`) to reflect non-video type traffic (Tab. 2). For the web browsing traces (`Web`), we randomly visited one of the Alexa top 50 websites [29] every 3 s for 120 s, to generate a similar traffic shape (*i.e.*, the ON-OFF pattern) as video traffic. Finally, we constructed the `Teleconf` dataset by using Google Meet and letting conferences last 2 min.

We considered the 100 most viewed YouTube videos as the known video list and trained a classifier using the 1080p video traces of the `YouTube100` dataset used in §5.3. After building the classifier, we defined an additional *"unseen"* class, to which unseen traffic was classified. We established classification criteria as follows: if the maximum value in the softmax (*i.e.*, the classifier's output) is below a specified threshold, a given trace in query belongs to the *"unseen"* class, regardless of the video title. Here, we followed the same assumption regarding the open-world setting used in previous studies [44, 46].

For practical deployment, the adversary may need to set a confidence threshold. In general, the classifier successfully filtered out the unseen traces as the confidence threshold value increased; however, the known traces were easily misidentified as unseen at a high confidence threshold.

Fig. 10 shows the identification accuracy across various confidence thresholds for four different types of traces along with the known video traces (dotted line). In particular, we observed that the classifier performed differently according to the type of unseen traces. The identification accuracy of each dataset reached 0.9 at the following confidence threshold; 0.91 (`Netflix`), 0.97 (`YouTube200`), 0.4 (`Teleconf`), and 0.76 (`Web`). Meanwhile, the classifier achieved an accuracy of over 0.9 at any confidence thresholds for the known YouTube traces.

Among the traffic types, the classifier outperformed for the non-video type (`Teleconf` & `Web`) and video streaming traffic from other content providers (`Netflix`). To explain this outperformance, we investigated the recorded traffic for each type and confirmed that each unseen traffic type had a distinctive shape. Since video traffic has a distinctive ON-OFF pattern distinguishing it from that of other service types, we observed that the classifier successfully identified the non-video type traffic (`Teleconf` & `Web`). For the `Netflix` dataset, the duration of ON-OFF periods differs from that of the YouTube traces, resulting in a low confidence threshold value to achieve a high identification accuracy. For the *unseen* traffic from the same content provider (`YouTube200`), although the classifier requires a high confidence value to achieve a high accuracy, the adversary benefits from a high precision.

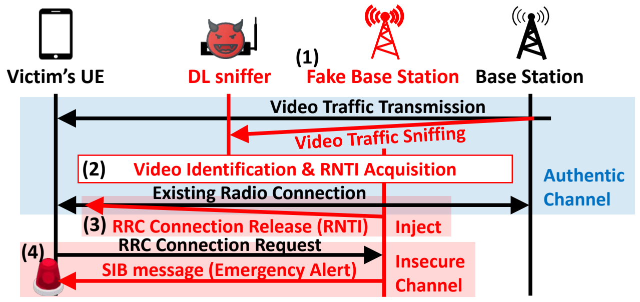
6 End-to-End Attack Scenario

We demonstrate an end-to-end attack that chains the proposed video identification attack with an emergency alerting attack to disclose the physical locations of victims. In particular, we introduce a scenario in which an adversary (*e.g.*, a law enforcement agency in an oppressive regime) forces the UEs of victims (*e.g.*, citizens) that have streamed a particular (*e.g.*, anti-government) video to make a loud tone, enabling the adversary who resides near the UEs physically to possibly locate them.

The key idea is that the adversary *selectively* forces only a target UE to move to their fake base station (FBS) without disrupting other UEs connected to a legitimate eNB. Then, the FBS sends *presidential alerts* to the target UE.

6.1 Attack Overview

Fig. 11 shows the overall procedure, which we detail below:
(1) Setting up an FBS in unused radio frequency: The adversary first sets up an FBS in a radio frequency not in use by MNOs, preventing every UE connected to a legitimate base station from moving to the FBS. The adversary simply avoids all the frequencies dedicated to MNOs, which are publicly available. All UEs, by design [6], connect to i) the radio frequency to which it is connected in prior, or ii) one in the list of commercially used frequencies. As the FBS operates in an unused radio frequency, all UEs ignore this FBS and do not connect to it. Therefore, to force the target UEs to connect to their FBS, the adversary is required to identify them beforehand.



Victim's UE
DL sniffer
(1)
Fake Base Station
Base Station
Video Traffic Transmission
Video Traffic Sniffing
(2) Video Identification & RNTI Acquisition
Existing Radio Connection
Authentic Channel
(3) RRC Connection Release (RNTI)
Inject
RRC Connection Request
(4)
SIB message (Emergency Alert)
Insecure Channel

Figure 11: End-to-End attack procedure

(2) Identifying victims: The adversary approaches the legitimate base station and tracks down UEs via the video identification attack (§4). Note that before conducting the video identification attack, the adversary is not assumed to have any prior knowledge of the target UEs. Given traffic collected from a cell and a target video list, the adversary conducts the attack and consequently obtains a tuple (a TMSI, the chain of RNTIs assigned to the TMSI, and a video title) for each UE. With this information, the adversary can pinpoint a set of target UEs.
(3) Redirecting victims to FBS: The adversary now forces each of the identified UEs to connect to their FBS by sending a crafted `RRCConnectionRelease` message. Specifically, the adversary **injects** the crafted message into the communication between the target UE and the legitimate base station to which the target UE is currently connected. Therefore, the target UE is not yet connected to the FBS when it receives the crafted message. Recall that a UE receives data only if its own RNTI is marked in DCIs (§2.1). The adversary can selectively inject the message to the target UE using their RNTIs obtained in the prior step without affecting the connections of other UEs to the legitimate base station.

Upon receiving the message, the target UE immediately releases the existing connection to the legitimate base station. Furthermore, by forging the `redirectedCarrierInfo` and `idleModeMobilityControlInfo` fields in the crafted message, which specify the next frequency to connect, the adversary can redirect the target UE to their FBS. To successfully inject the crafted message, the adversary is required to address the two challenges that we describe in §6.2.
(4) Sending fake presidential alerts: Finally, the adversary sends fake presidential alerts to the target UEs redirected to their FBS, as presented in previous studies [24, 33]. By design, a UE, in the majority of the cases, sounds a loud alarm for

each presidential alert, thereby revealing its physical presence at a venue. For example, in a square or large hall, by arranging the co-workers around the space, the adversary can possibly identify the victims with the alarm.

6.2 Details of Redirecting Target UEs

When conducting the end-to-end attack, all UEs in a target cell are connected to a legitimate base station. To redirect only the target UEs without disturbing other UEs (step (3) in §6.1), the adversary has to address two challenges. The adversary has to 1) inject a crafted message for each target UE through the radio connection to the legitimate base station and 2) enforce the target UE to accept the crafted message.

Injecting a crafted message. We leveraged the signal overshadowing attack [55], which is designed to inject a broadcast message. Particularly, we extended it to support injecting a unicast message, such as an `RRCConnectionRelease` message. Although this seems straightforward, it requires additional considerations that involve a) locating the message on the resource grid properly, and b) encoding the message with the proper eNB's configurations (*e.g.*, transmission mode and generating DCI). By addressing these issues, we succeeded in injecting a crafted `RRCConnectionRelease` message over the air to a specific UE.

Enforcing the target UE to accept the crafted message. We propose to inject an `RRCConnectionRelease` message during the reconnection procedure of a target UE. When a UE connects to an eNB, its security context for secure data transmission is also established. Having the security context, a UE, by design [9], is enforced to discard plain `RRCConnectionRelease` messages. Here lies the problem that the adversary can inject only plain messages due to the absence of the victim's cryptographic keys (§3). For this, we propose to exploit the reconnection procedure that accompanies the reestablishment of the security context. By tracking the victim's connection status, the adversary can detect the moment when the security context is released. Thus, the adversary injects a crafted `RRCConnectionRelease` message before a new security context is established.

Meanwhile, if the target UE has an implementation flaw that it accepts plain `RRCConnectionRelease` messages although having a security context, the attack becomes more efficient and effective. Note that the 3GPP specification [9] explicitly states that *the UE shall discard plain* `RRCConnectionRelease` *messages if it has a security context.* By analyzing several types of UEs, we indeed found such flaws in two types of Samsung Galaxy S4/S5 devices equipped with Qualcomm baseband chipsets. We reported the vulnerability to Qualcomm and Samsung and received confirmation from both vendors. This means that UEs equipped with such vulnerable chipsets accept malicious messages. Thus, the adversary can immediately conduct the attack.

Differences to previous studies. Other studies [24, 33] have also presented attacks that send presidential alerts. The attacks involving presidential alerts generally consist of two phases: 1) making UEs connect to an FBS and 2) sending presidential alerts to all UEs connected to the FBS. Our attack differs in the first phase compared to the previous ones [24, 33]. It selectively moves only target UEs to an FBS whereas previous studies enforce all UEs in the target cell to move to an FBS by increasing the FBS's signal strength.

Verifying the feasibility of the attack. We implemented an FBS by using an open-source LTE protocol stack [47]. It consists of two components: a) one that broadcasts fake presidential alerts, and b) the other that injects a crafted message. The components ran on USRP B210 [50] and X310 [51], respectively. Using the vulnerability described above, we succeeded in conducting the attack [1]. We further discuss the practical requirements of the attack in App. E.

7 Mitigations

In this section, we first explore lightweight mitigation, which requires no changes to UEs and the 3GPP specification. We then discuss viable alternatives and their requirements.

Light-weight mitigation to MNOs. One mitigation is to make the adversary unable to link observed RNTIs to the TMSI (§4.2), thus rendering them incapable of capturing the complete traffic volume. For this, we propose a small modification to the operational sequence to reassign RNTIs as confidential. When establishing a new radio connection including its security context, a UE would use the first assigned RNTI, which could be tracked by the adversary. At this point, the proposed mitigation is to make an eNB issue a new RNTI via an encrypted `RRCConnectionReconfiguration` message after establishing a security context. The proposed approach induces a UE to use a new RNTI by handover to the same cell. This approach requires additional transmission of `RRCConnectionReconfiguration/ReconfigurationComplete` messages and the radio connection reestablishment procedure.

We implemented this mitigation on an eNB that runs on USRP B210 by modifying an open-source LTE network [47]. We only added fewer than 70 LoC in the srsENB code, which is mainly related to the logic of re-assigning RNTIs. We used five UEs from three different baseband manufacturers and confirmed that the implemented mitigation seamlessly worked without any changes to the UEs.

Limitations. The proposed mitigation is easily applicable to commercial networks but has limitations. It imposes performance overhead. We measured the time gap between the receipt of an `RRCConnectionReconfiguration` message and the establishment of a radio connection with the new RNTI.[4]

Table 6: Overhead of the proposed mitigation

Baseband	Qualcomm		Exynos		MediaTek
UE	Galaxy S8	Galaxy S20	Galaxy S8	Galaxy S10	LG X6
Avg. (ms)	35.70	30.20	8.19	7.38	20.2

[4]`RRCConnectionReconfiguration` message containing *newUE-Identity* was sent after an `AttachComplete` message. We measured time on

This gap represents the additional processing time introduced by the mitigation. Tab. 6 shows that the average overhead of 30 trials is 7.38–35.7 ms across five different UEs. These overheads are approximately 4–21% of the average latency of LTE service establishment (168.7 ms) for major US operators [35].

Another limitation is that the mitigation only prevents the adversary from tracking RNTIs. This partially mitigates the threats of video identification because it does not eliminate distinguishable traffic patterns of streaming videos.

Viable yet impractical mitigation. There have been various mitigations to remove the root cause of the video identification attack: 1) eliminating distinguishable traffic patterns for each video; and 2) encrypting DCI. However, those are hard to be adopted in cellular networks because of their limited practicality. They require significant changes in terms of both the implementation and design of video services and cellular networks.

To eliminate distinguishable traffic patterns, prior works have introduced several approaches: 1) making the bitrate constant with a rate control [44]; 2) adding a noise or padding to each streaming chunk [20, 60]; and 3) varying the size of each chunk randomly [44]. However, increasing the size of a video chunk may undermine the video quality as it would require a larger number of chunks to play the same video. Furthermore, users tend to be sensitive to the usage of mobile data, which affects their payments. For instance, Zhang *et al*. [60] proposed adding noises with a volume more than twice that of an original video; this policy would deplete the users' monthly data allowance three times faster.

Another mitigation is to encrypt DCIs. Considering that the proposed attack exploits a time series of traffic volumes by decoding DCIs, encrypting DCIs eliminates the source of exploitation. However, this approach would cause significant overhead to both a cell tower and a UE, as DCIs are sent via radio signals every 1 ms, already entailing a large number of decoding and encoding processes. In addition, introducing encryption to a non-secure protocol layer requires additional considerations on the design choice of key encryption and management (including the overhead), thus involving disruptive changes for the LTE protocol design.

Discussion on mitigations. The fundamental cause enabling this privacy-threatening attack is unencrypted information broadcast over the air. However, encrypting such information entails inevitable performance overhead, thus provoking mundane discussions of the trade-off between privacy and performance. We thus encourage ground-breaking research that eliminates information-leaking public channels with negligible overhead for the next generation of cellular networks.

8 Discussion

Handling the moving target. One possible limitation of this work is to link the identity of the victim who moves to different eNBs. Roger *et al*. revealed that the identifier of the victim (*i.e.*, RNTI) could be tracked when the UE moves to another eNB [26] because the `RRCConnectionReconfiguration` message is not encrypted and RNTI is not random. However, according to our measurement, the RNTI values are changed to random values through encrypted messages whenever the UE moves to another cell. Nevertheless, due to the unique working logic of HAS, there is still a chance to track the user. As we pointed out in §3.1, TMSI-RNTI mapping is possible when the radio connection is reestablished after the OFF periods. Thus, the adversary is able to sniff the victim's traffic continuously. Otherwise, the adversary needs to handle the partially monitored traffic. For this, it may involve 1) augmenting the training data with LTE traces from different parts of the video, and 2) making the classifier shift-invariant.

Identifying the traffic containing ads. Our work provides a way to measure the complete LTE traffic volume, but it still has room for improvement in identifying the traffic with ads. Note that the observed video traffic becomes distorted when the ads are inserted. For example, due to ad traffic, fetching the remaining chunks of a target video is delayed. Thus, the collected traffic becomes different from the shape of its original traffic without video ads. One plausible solution is to train the classifier along with traces in which video ads are redacted. Therefore, a query for this classifier should become an ad-redacted video as well. However, the starting time and duration of ads are not deterministic, and these ads vary by users [56, 57]. Thus, for each classifier query, the adversary should identify time windows of playing ads and redact traffic volumes in these windows for accurate video identification. One alternative is to design a shift-invariant classifier. However, designing such a classifier is known to be challenging [59], and we believe that designing a shift-invariant classifier is orthogonal to our attack. We leave these additional challenges as future work.

Targeted attacks. One may extend our attack to target a specific user who is streaming a video of the adversary's interest. This *targeted* attack assumes an adversary who has the victim's soft-identifier, such as a phone number or social media accounts. This adversary requires addressing the following challenges for successful video identification: they should 1) localize the victim's physical cell location and approach the cell coverage, 2) determine the victim's MNO, and 3) obtain the victim's TMSI. The adversary may have several options to achieve these conditions. Unfortunately, the first challenge is known to be challenging [22, 31, 45]; thus, this can be a promising future work. For the second challenge, the adversary can set up multiple downlink sniffers for all MNOs in their coverage; in most countries, there are 2-4 MNOs. Besides, scaling up the attack does not require a high cost considering an SDR cost. For the third challenge, the adversary can leverage several techniques that associate the victim's soft-identifier with the TMSI [22, 25, 31, 45]. One representative attack is a silent paging that exploits the relatively

the UE side from the reception of the `RRCConnectionReconfiguration` message to the reception of the `RandomAccessResponse` message.

unchanging nature of TMSI. The adversary makes multiple calls to the victim and monitors a repeatedly appearing TMSI in paging messages [22, 31, 45]. ToRPEDO [25] also proposed a way to retrieve the victim's TMSI by exploiting the deterministic paging location.

9 Related Work

Video identification attack. There have been several studies with the same goal of exploiting the VBR-encoded content in other types of networks. Most studies on wired networks [20, 21, 38, 44, 52] or 802.11 [43] assumed a strong attack model that the adversary has the ability to access the victim's network infrastructure or device. In contrast, the adversary in our threat model only harnesses broadcast channels over the air without any direct access, as presented in wireless networks [34]. We concretized practical challenges and presented a new approach to a video identification attack in cellular networks. The prior study on 802.11 [43] also assumed a strong adversary; the authors collected data packets in a pre-connected desktop in the same subnet of the victim, without considering actual radio signals broadcast over the air.

Traffic analysis in cellular networks. In cellular networks, although a few previous studies [29, 41] have presented website identification attacks, no one has presented a video identification attack. Compared to the website identification attacks, video identification attacks in cellular networks differs in two aspects: (1) video streaming often carries a large volume of video chunks, which triggers CA; (2) the streaming policy of a HAS can cause a long OFF period, which triggers the victim's RNTI to be changed. These two characteristics eventually raise practical challenges, as discussed in §3.1.

10 Conclusion

This work is the first empirical study on a video identification attack in commercial LTE networks. We show that an adversary can pinpoint victims watching target videos of their interest without any access to their UEs or the LTE infrastructure. By leveraging the proposed methods that exploit the information embedded in broadcast radio signals, our extensive evaluation of commercial LTE traffic emphasizes that the adversary with a single SDR can achieve high accuracy in video identification. We conclude by proposing a lightweight countermeasure readily deployable without hardware or specification changes to MNOs and LTE networks.

Acknowledgment

We sincerely appreciate our reviewers and shepherd, Omar Haider Chowdhury, for their insightful comments and suggestions. This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2020-0-00209, Investigation of new security and privacy threats in Web5G platforms, and No.2019-0-01343, regional strategic industry convergence security core talent training business)

References

[1] Watching the Watchers. https://sites.google.com/view/sec21-wtw/watching-the-watchers.

[2] 3GPP. TR 36.808. Carrier Aggregation; Base Station (BS) radio transmission and reception.

[3] 3GPP. TS 33.501. Security architecture and procedures for 5G System.

[4] 3GPP. TS 36.101. User Equipment (UE) radio transmission and reception.

[5] 3GPP. TS 36.212. Multiplexing and channel coding.

[6] 3GPP. TS 36.304. User Equipment (UE) procedures in idle mode.

[7] 3GPP. TS 36.321. Medium Access Control (MAC) protocol specification.

[8] 3GPP. TS 36.323. Packet Data Convergence Protocol (PDCP) specification.

[9] 3GPP. TS 36.331. LTE RRC Protocol specification.

[10] 3GPP. TS 38.331. 5G NR RRC Protocol specification.

[11] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *USENIX Symposium on Operating Systems Design and Implementation*, pages 265–283, 2016.

[12] AirScope. http://www.softwareradiosystems.com/tag/airscope.

[13] Arjun Balasingam, Manu Bansal, Rakesh Misra, Rahul Tandra, Aaron Schulman, and Sachin Katti. Poster: Broadcast LTE Data Reveals Application Type. In *International Conference on Mobile Computing and Networking*, pages 531–533, 2017.

[14] Donald J Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop*, pages 359–370. Seattle, WA, 1994.

[15] Nicola Bui and Joerg Widmer. Owl: a reliable online watcher for LTE control channel measurements. In *ACM Workshop on All Things Cellular: Operations, Applications and Challenges*, pages 25–30, 2016.

[16] List of LTE networks. https://en.wikipedia.org/wiki/List_of_LTE_networks.

[17] Francois Chollet. *Deep Learning with Python and Keras: The practical guide from the developer of the Keras library*. MITP-Verlags GmbH & Co. KG, 2018.

[18] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update 2017-2022 White Paper.

[19] Haotian Deng, Kai Ling, Junpeng Guo, and Chunyi Peng. Unveiling the Missed 4.5 G Performance In the Wild. In *International Workshop on Mobile Computing Systems and Applications*, pages 86–91, 2020.

[20] Ran Dubin, Amit Dvir, Ofir Pele, and Ofer Hadar. I Know What You Saw Last Minute-Encrypted HTTP Adaptive Video Streaming Title Classification. *IEEE Transactions on Information Forensics and Security*, 12(12):3039–3049, 2017.

[21] Jiaxi Gu, Jiliang Wang, Zhiwen Yu, and Kele Shen. Walls have ears: Traffic-based side-channel attack in video streaming. In *IEEE Conference on Computer Communications*, pages 1538–1546, 2018.

[22] Byeongdo Hong, Sangwook Bae, and Yongdae Kim. GUTI Reallocation Demystified: Cellular location tracking with changing temporary identifier. In *Network and Distributed System Security Symposium*, 2018.

[23] Byeongdo Hong, Shinjo Park, Hongil Kim, Dongkwan Kim, Hyunwook Hong, Hyunwoo Choi, Jean-Pierre Seifert, Sung-Ju Lee, and Yongdae Kim. Peeking over the Cellular Walled Gardens-A Method for Closed Network Diagnosis. *IEEE Transactions on Mobile Computing*, 17(10):2366–2380, 2018.

[24] Syed Rafiul Hussain, Omar Chowdhury, Shagufta Mehnaz, and Elisa Bertino. LTEInspector: A systematic approach for adversarial testing of 4G LTE. In *Network and Distributed System Security Symposium*, 2018.

[25] Syed Rafiul Hussain, Mitziu Echeverria, Omar Chowdhury, Ninghui Li, and Elisa Bertino. Privacy attacks to the 4G and 5G cellular paging protocols using side channel information. In *Network and Distributed System Security Symposium*, 2019.

[26] Roger Piqueras Jover. LTE security, protocol exploits and location tracking experimentation with low-cost software radio. *arXiv preprint arXiv:1607.05171*, 2016.

[27] Ahmed Khattab, Joseph Camp, Chris Hunter, Patrick Murphy, Ashutosh Sabharwal, and Edward W Knightly. WARP: a flexible platform for clean-slate wireless medium access protocol design. *ACM SIGMOBILE Mobile Computing and Communications Review*, 12(1):56–58, 2008.

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

[29] Katharina Kohls, David Rupprecht, Thorsten Holz, and Christina Pöpper. Lost traffic encryption: fingerprinting LTE/4G traffic on layer two. In *Conference on Security and Privacy in Wireless and Mobile Networks*, pages 249–260, 2019.

[30] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[31] Denis Foo Kune, John Koelndorfer, Nicholas Hopper, and Yongdae Kim. Location leaks on the GSM Air Interface. In *Network and Distributed System Security Symposium*, 2012.

[32] Adam Langley, Alistair Riddoch, Alyssa Wilk, Antonio Vicente, Charles Krasic, Dan Zhang, Fan Yang, Fedor Kouranov, Ian Swett, Janardhan Iyengar, et al. The QUIC transport protocol: Design and internet-scale deployment. In *ACM SIGCOMM*, pages 183–196, 2017.

[33] Gyuhong Lee, Jihoon Lee, Jinsung Lee, Youngbin Im, Max Hollingsworth, Eric Wustrow, Dirk Grunwald, and Sangtae Ha. This is your president speaking: Spoofing alerts in 4G LTE networks. In *International Conference on Mobile Computing and Networking*, pages 404–416, 2019.

[34] Ying Li, Yi Huang, Richard Xu, Suranga Seneviratne, Kanchana Thilakarathna, Adriel Cheng, Darren Webb, and Guillaume Jourjon. Deep Content: Unveiling video streaming content from encrypted WiFi traffic. In *International Symposium on Network Computing and Applications (NCA)*, pages 1–8. IEEE, 2018.

[35] Yuanjie Li, Zengwen Yuan, and Chunyi Peng. A control-plane perspective on reducing data access latency in LTE networks. In *International Conference on Mobile Computing and Networking*, pages 56–69, 2017.

[36] Andre Puschmann (Developer of AirScope). Personal communication, July. 27 2020.

[37] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

[38] Andrew Reed and Michael Kranch. Identifying https-protected netflix videos in real-time. In *ACM Conference on Data and Application Security and Privacy*, pages 361–368, 2017.

[39] Jean-Gabriel Remy and Charlotte Letamendia. *LTE standards*. Wiley Online Library, 2014.

[40] RFC 3095. RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed.

[41] David Rupprecht, Katharina Kohls, Thorsten Holz, and Christina Pöpper. Breaking LTE on Layer Two. In *IEEE Symposium on Security and Privacy*, pages 1121–1136, 2019.

[42] Sandvine 2019 Mobile Internet Phenomena Report.

[43] T Scott Saponas, Jonathan Lester, Carl Hartung, Sameer Agarwal, Tadayoshi Kohno, et al. Devices that tell on you: Privacy trends in consumer ubiquitous computing. In *USENIX Security Symposium*, pages 55–70, 2007.

[44] Roei Schuster, Vitaly Shmatikov, and Eran Tromer. Beauty and the burst: Remote identification of encrypted video streams. In *USENIX Security Symposium*, pages 1357–1374, 2017.

[45] Altaf Shaik, Ravishankar Borgaonkar, N Asokan, Valtteri Niemi, and Jean-Pierre Seifert. Practical Attacks Against Privacy and Availability in 4G/LTE Mobile Communication Systems. In *Network and Distributed System Security Symposium*, 2016.

[46] Sandra Siby, Marc Juarez, Claudia Diaz, Narseo Vallina-Rodriguez, and Carmela Troncoso. Encrypted DNS–> Privacy? A traffic analysis perspective. In *Network and Distributed System Security Symposium*, 2020.

[47] srsLTE. https://github.com/srsLTE/srsLTE.

[48] Johan AK Suykens and Joos Vandewalle. Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300, 1999.

[49] Hoang Duy Trinh, Angel Fernandez Gambin, Lorenza Giupponi, and Paolo Dini. Mobile traffic classification through physical channel fingerprinting: a deep learning approach. *arXiv preprint arXiv:1910.11617*, 2019.

[50] USRP B210. https://www.ettus.com/UB210-KIT.

[51] USRP X310. https://www.ettus.com/x310-kit/.

[52] Andrew M White, Austin R Matthews, Kevin Z Snow, and Fabian Monrose. Phonotactic reconstruction of encrypted voip conversations: Hookt on fon-iks. In *IEEE Symposium on Security and Privacy*, pages 3–18, 2011.

[53] Haoyang Wu, Tao Wang, Zengwen Yuan, Chunyi Peng, Zhiwei Li, Zhaowei Tan, Boyan Ding, Xiaoguang Li, Yuanjie Li, Jun Liu, et al. The tick programmable low-latency sdr system. In *International Conference on Mobile Computing and Networking*, pages 101–113, 2017.

[54] Accuver XCAL. http://www.accuver.com/.

[55] Hojoon Yang, Sangwook Bae, Mincheol Son, Hongil Kim, Song Min Kim, and Yongdae Kim. Hiding in plain signal: physical signal overshadowing attack on LTE. In *USENIX Security Symposium*, pages 55–72, 2019.

[56] Manage mid-roll ad breaks in long videos. https://support.google.com/youtube/answer/6175006?hl=en.

[57] How advertisers target. https://creatoracademy.youtube.com/page/lesson/how-ads-work_advertising-landscape_video?cid=how-ads-work&hl=en#strategies-zippy-link-1.

[58] Jialiang Zhang, Xinyu Zhang, Pushkar Kulkarni, and Parameswaran Ramanathan. OpenMili: a 60 GHz software radio platform with a reconfigurable phased-array antenna. In *International Conference on Mobile Computing and Networking*, pages 162–175, 2016.

[59] Richard Zhang. Making convolutional networks shift-invariant again. In *International Conference on Machine Learning*, pages 7324–7334. PMLR, 2019.

[60] Xiaokuan Zhang, Jihun Hamm, Michael K Reiter, and Yinqian Zhang. Statistical privacy for streaming traffic. In *Network and Distributed System Security Symposium*, 2019.

Appendix

A Experimental Setup and Ethics

Network environment. To collect the video traces for each MNO, we connect a downlink sniffer and UE, which reside in the same building, to the same cell. The distance between the sniffer and the cell for MNO A, B, and C is 380m, 170m, and 300m, respectively. For all MNOs, we set the UE to use the PCell covering a 10 MHz bandwidth. When CA is activated, the UE connects to the two SCells that cover the 10 MHz and 20 MHz bandwidth, respectively. When collecting these CA-activated traces, we use MNO A that exhibits a high signal to noise ratio (over 23 dB) among the three MNOs at our site.[1]

Configuration of the UE. We used a legitimate application from each video streaming service provider as follows: Netflix (7.47.0 and 7.48.0), YouTube (15.05.54, 15.07.52, and 14.47.50), Amazon Prime (3.0.281), and Google Meet (43.5.3213). Particularly for YouTube traces (*i.e.*, `YouTube100` & `YouTube200`), we used a premium account; thus, the traces do not contain any ads. During the data collection procedure, we did not execute other applications, meaning that one video application was displayed on top of the test UE.

Ethics. We collected data traces from the operating LTE networks. We honored the privacy of other users in the same eNB in which our sniffer locates. We analyzed information only

Table 7: DTW and SVM classifier result on YouTube dataset

	DTW			SVM		
MNO	480p	720p	1080p	480p	720p	1080p
A	0.860	0.791	0.798	0.877	0.865	0.951
B	0.815	0.897	0.666	0.827	0.858	0.890
C	0.950	0.899	0.767	0.905	0.898	0.894

related to our testing UEs and computed the classifier only with traffic from these UEs. We did not keep the remaining information in the output from the downlink sniffer.

B Comparative Analysis of Other Classifier

We implemented two additional classifiers, DTW [14] and SVM [48], which were used in previous studies [20, 21] to evaluate their attacks. We performed five-fold cross-validation on the `YouTube100` dataset in Tab. 2, with the same experimental setup. That is, the classifiers received the same input as our CNN classifier. Both classifiers exhibited relatively low performance compared to ours (Tab. 7). The DTW classifier yielded an accuracy of 0.666–0.950 across all MNOs and video resolutions. The SVM classifier achieved a comparable accuracy of 0.827–0.951 and showed a higher accuracy than that of the DTW classifier on the 1080p dataset. As we discussed in §5.4, the classifier using DTW is highly affected by network congestion and the quality of streamed video chunks. This is mainly because the traffic shape is likely to lag when the client receives a high-resolution video chunk with a larger volume than that of the low-resolution chunk.

C Characteristics of Video Service Providers

The HAS working logic of the tested video streaming service providers differs in three key factors: the size of an initial playback buffer, the duration of a chunk, and the algorithm to fetch a chunk. Based on these factors, the video traffic also shows a distinctive pattern. As Fig. 12 illustrates, Amazon shows a long "First ON period", which implies that it uses a large initial buffer. In contrast, YouTube has a short "First ON period". In addition, they show the same duration of the OFF period, which implies that they have the same chunk duration.

D Extension to 5G

We demonstrated that our video identification attack is feasible in LTE networks. Meanwhile, extending the attack to upcoming 5G networks requires the following challenges to be addressed: multiple 5G deployment options, sniffing 5G downlink messages, large computation resources, and SDRs that support 5G radio signals.

Our attack may not work in 5G-NSA networks. In 5G-NSA, to use legacy LTE core networks, each LTE radio connection and its security context must be established first. Then, an eNB assigns a new 5G-RNTI and establishes a new 5G connection [10]. As the 5G-RNTI is sent in an encrypted message, an adversary cannot map the victim's TMSI to the RNTIs.

Conversely, the root cause of the attack remains in 5G-SA networks. Although the 5G specification introduces new security features, TMSIs are not affected [3]. Furthermore, a 5G-RNTI is exposed in plaintext; therefore, an adversary is still able to link the victim's RNTI to the TMSI and distinguish the victim's traffic. As DCIs broadcast over the air remain unencrypted, the proposed attack is still feasible.

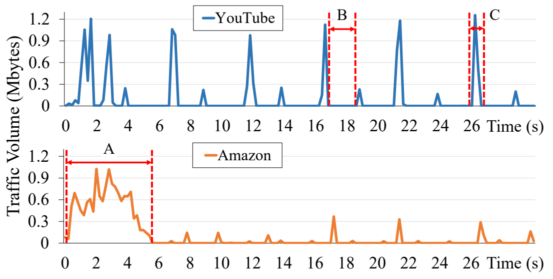


Figure 12: Video traffic volumes for YouTube and Amazon

Nevertheless, there still exist several practical challenges. First, there is currently no downlink sniffing tool for 5G networks to decode data traffic on either 5G-NSA or 5G-SA networks. Second, there is no existing commercial SDR that is capable of sniffing 5G radio signals; existing commercial SDRs only support up to sub-6 GHz. Furthermore, the wider bandwidth of 5G networks increases the volume of traffic to monitor on the PDCCH and PDSCH channels; thus, the adversary has to leverage a higher computational power to capture traffic. We believe that these challenges will be resolved soon as the advancement of SDR technology [27, 53, 58] accelerates, and downlink sniffing software is being developed.

E Requirements of End-to-End Attack

Our end-to-end attack shares two requirements with the original signal injection attack [55]: 1) precise time/frequency synchronization, and 2) 3 dB higher signal strength. First, the adversary has to synchronize their FBS to a legitimate eNB, to which a target UE is connected currently, in the time and frequency domain. This is because the adversary has to overwrite a malicious signal over the target UE's dedicated region in the resource grid precisely. For a stable attack, the adversary can use a high-precision clock, such as a GPS disciplined oscillator. Second, the adversary has to transmit a malicious signal with a 3 dB higher strength compared to the one from the legitimate eNB. The required signal strength can be calculated as follows: $P_a = 10^{(\frac{3+20*\log_{10}(\frac{D_a}{D_b})}{10})} \cdot 100\,W$, where D_a denotes the distance between the victim and FBS, D_b indicates the distance between the victim and legitimated eNB, and 100 W is the eNB's signal strength; we assume a free-space path loss.