**TickTock**: Detecting Microphone Status in Laptops Leveraging Electromagnetic Leakage of Clock Signals

Soundarya Ramesh  
sramesh@comp.nus.edu.sg  
National University of Singapore  
Singapore

Ghozali Suhariyanto Hadi  
ghadi@comp.nus.edu.sg  
National University of Singapore  
Singapore

Sihun Yang  
ddyshsh@yonsei.ac.kr  
Yonsei University  
Republic of Korea

Mun Choon Chan  
chanmc@comp.nus.edu.sg  
National University of Singapore  
Singapore

Jun Han  
junhan@cyphy-lab.org  
Yonsei University  
Republic of Korea

**ABSTRACT**

We are witnessing a heightened surge in remote privacy attacks on laptop computers. These attacks often exploit malware to remotely gain access to webcams and microphones in order to spy on the victim users. While webcam attacks are somewhat defended with widely available commercial webcam privacy covers, unfortunately, there are no adequate solutions to thwart the attacks on mics despite recent industry efforts. As a first step towards defending against such attacks on laptop mics, we propose *TickTock*, a novel mic on/off status detection system. To achieve this, *TickTock* externally probes the electromagnetic (EM) emanations that stem from the connectors and cables of the laptop circuitry carrying mic clock signals. This is possible because the mic clock signals are only input during the mic recording state, causing resulting emanations. We design and implement a proof-of-concept system to demonstrate *TickTock*’s feasibility. Furthermore, we comprehensively evaluate *TickTock* on a total of 30 popular laptops executing a variety of applications to successfully detect mic status in 27 laptops. Of these, *TickTock* consistently identifies mic recording with high true positive and negative rates.

**CCS CONCEPTS**

- Hardware → Sensors and actuators; Security and privacy → Side-channel analysis and countermeasures.

**KEYWORDS**

Electromagnetic Side Channels, Audio Privacy, Speech Privacy, Side-channel based Defense, Sensor Usage Detection

**ACM Reference Format:**

Soundarya Ramesh, Ghozali Suhariyanto Hadi, Sihun Yang, Mun Choon Chan, and Jun Han. 2022. *TickTock*: Detecting Microphone Status in Laptops Leveraging Electromagnetic Leakage of Clock Signals. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security (CCS ’22)*, November 7–11, 2022, Los Angeles, CA, USA. ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3548606.3560698

---

**1 INTRODUCTION**

Remote privacy attacks on modern day laptops continue to cause significant social problems. For example, remote attackers inject malware to gain access to webcams to stealthily spy on victims by disabling the webcam’s indicator LED [3, 11, 44, 55]. To defend against such attacks, users often place commercially available webcam privacy covers to physically block the webcams [46, 55]. Exacerbating the problem, there are also reported attacks that *spy on laptop microphones* – including zero-day vulnerabilities and stalker-installed malware that stealthily eavesdrop from victims’ laptops [9, 35, 45, 59]. Moreover, bugs have been identified until recently in popular video calling apps, such as Zoom, which captured audio on Mac OS, even after the meeting had ended [52]. Unlike webcam covers, there are no immediately adequate solutions to defend against mic-based eavesdropping.

To defend against such attacks, companies such as Purism are pushing forward new laptop designs with hardware kill switches for mics, which can cut off power supply to the mics when not in use [48, 49]. Apple designed a hardware disconnect feature for Macbook 2019 and later models, which disables the mic whenever the lid is closed [56]. Dell has updated its drivers on newer devices to allow for disabling mics at the operating systems level [64]. Furthermore, several operating systems such as Windows 10 and Mac OS 12 are providing indicators on screen during mic usage for increased user awareness [16, 17].

While these efforts are promising first steps, they all suffer from significant shortcomings. First, these solutions require users to trust the implementation of the laptop manufacturers or the operating systems, both of which have been compromised by attackers several times in the past [3, 15, 66] or that the manufacturers themselves could be malicious. Second, these solutions are incorporated in only a small fraction of devices, hence most current day laptops do not have a way to detect/prevent eavesdropping.

The aforementioned shortcomings lead us to the following research question: *Can we design a novel mic-based eavesdropping attack detection system that – (1) is robust to powerful remote attackers, (2) is applicable to existing laptops without any modifications, and (3) places limited trust on device manufacturers?* To this end, we propose *TickTock* that utilizes the phenomenon that digital MEMS mics equipped in commodity laptops, when turned on (i.e., while recording), emanate electromagnetic (EM) signals. The emanation stems from the cables and connectors that carry the clock
signal to the mic hardware, ultimately to operate its analog-to-
digital converter (ADC) (see §4.1). *TickTock* captures this leakage to *identify the on/off status* of the laptop mic. Figure 1 depicts the process of utilizing *TickTock*. The user locates *TickTock* device — consisting of a small EM probe — on the external housing of the laptop near the leakage location. When the mic starts recording, *TickTock* detects the clock signal and alerts the user (e.g., LED lights up). We envision *TickTock* to have a form-factor, similar to a USB drive (Figure 1), that can be adhered to the external of the laptop for detecting mic on/off status. However, *TickTock*’s current fully-functional prototype has a table-top form-factor (Figure 2), but we see several opportunities to miniaturize this further (see §8).

Designing *TickTock* leads to three significant challenges. First, the frequency of the mic clock signal is unknown as its value varies across devices (typically ranging between 512 kHz to 4.8 MHz), particularly depending on the audio codec chip. Second, the location of maximum leakage of the EM signals due to the mic clock signals is also unknown, as it depends on the underlying location of the leaking cables and connectors. Third, as the EM signals captured typically include noise from neighbouring signal lines, we need to devise a robust mechanism for preventing false predictions.

To overcome the aforementioned challenges, *TickTock* uses a one-time bootstrapping process per device model to infer the mic clock frequency ($f_{mic}$), as well as the maximum leakage location ($l_{max}$). In order to solve the third challenge of robust detection of clock signals in the presence of noise, *TickTock* leverages both the fundamental clock frequency as well as the harmonics, which are multiples of the fundamental frequency, to improve detection accuracy.

*TickTock* has several advantages. First, adversaries with software capabilities cannot evade our detection as *TickTock*’s approach relies on EM leakage due to the mic hardware, hence making it robust against powerful remote attackers. Second, as *TickTock*’s detection system is completely external to the devices themselves, it places minimal trust on the device manufacturers and software vendors.

We evaluate *TickTock* on a total of 30 laptops, with EM signals collected for over ten hours to demonstrate that *TickTock* detects mic activities across most laptop brands we tested including Lenovo, Dell, HP and Asus. We comprehensively evaluate *TickTock*’s performance over different mic-based applications (e.g., Zoom, Audacity), non-mic based applications (e.g., Google News, YouTube), as well as different audio driver implementations (e.g., Ubuntu vs. Windows). In addition, we also evaluate its real-time performance, as well as its robustness to EM noise. From our analysis, *TickTock* successfully identifies the mic clock frequency in 27 out of 30 tested laptops. Of the 27 laptops, *TickTock* consistently predicts mic activities with high true positive and negative rates.

## 2 SYSTEM AND THREAT MODEL

We present the system and threat models of *TickTock*.

**System Model.** The goal of *TickTock* is to identify mic recording status (i.e., on/off) in victim-owned devices, such as his/her laptop. We define a mic to be recording (i.e., mic on), whenever it captures physical acoustic signals and converts them into digital signals. Hence, we do not distinguish between cases where the digital signals from the mic are saved to memory vs. when they are discarded, and consider both as recording. *TickTock* is constrained to only capture EM leakages from close contact on a device (e.g., from external housing of a laptop). Hence, we do not consider mic status detection in spying devices (e.g., audio bugs hidden in a room). Furthermore, *TickTock* is constrained to only detect mic status in devices with digital mics (i.e., mics that require clock signals for their operation).

**Threat Model.** In designing *TickTock*, we consider an attacker with the following goal and capabilities. The attacker’s goal is to stealthily capture audio from the mic of the victim’s laptop. The attacker’s capabilities include launching remote attacks with unconstrained software capabilities. Specifically, we consider powerful attackers that may control malicious or compromised applications, and are capable of exploiting kernel vulnerabilities to modify the audio drivers. However, we assume that the attacker does not have physical access to the laptop, and cannot modify the hardware (e.g., embed a standalone audio bug within the laptop).

## 3 TICKTOCK USAGE SCENARIOS

This section presents the potential usage scenarios of *TickTock*.

![Figure 1: Figure depicts a scenario where a user places *TickTock* device (that equips an electromagnetic (EM) probe) in close vicinity of his/her laptop in order to detect a possible mic-based eavesdropping attack, namely by determining if its mic is ON or OFF. *TickTock* is able to do so based on the presence/absence of EM emanation of clock signals that are input to the mic in the laptop circuitry.](image1.png)

![Figure 2: Figure depicts fully functioning prototype of *TickTock*, consisting of different components stacked to the side of the laptop. However, as depicted on the left, we envision *TickTock* with a form-factor similar to a small USB drive to be placed in contact with the laptop's exterior housing.](image2.png)
TickTock: Detecting Microphone Status in Laptops Leveraging Electromagnetic Leakage of Clock Signals

Figure 3: Bootstrapping scenarios – (a) depicts bootstrapping performed by laptop manufacturers, e.g., Lenovo, who subsequently ship laptops with stickers denoting leakage location, \(l_{\text{mic}}\), and an accompanying TickTock device set to detect mic clock frequency, \(f_{\text{mic}}\). (b) depicts a crowd-sourcing scenario where users upload \(f_{\text{mic}}\), along with an image of \(l_{\text{mic}}\) to TickTock’s public server. (c) depicts a scenario where a user locally performs bootstrapping. (d) depicts a deployment scenario, where a user deploys TickTock to detect mic on/off status by placing the TickTock device at the location of the sticker, and by setting the TickTock device to detect \(f_{\text{mic}}\).

**Bootstrapping Scenario.** TickTock requires a one-time bootstrapping phase, to infer mic clock frequency, \(f_{\text{mic}}\), and maximum EM leakage location, \(l_{\text{mic}}\), for each device model. We present three scenarios we envision for different entities performing bootstrapping. (1) **Bootstrapping by Manufacturers.** Laptop manufacturers, e.g., Lenovo, can perform the bootstrapping phase for their products. Following which, as depicted in Figure 3(a), they ship each of their laptops with – (a) an accompanying TickTock device, that is set to detect the \(f_{\text{mic}}\) (e.g., 2 MHz), and (b) a sticker placed on the laptops (e.g., in the top-right corner), in order to mark \(l_{\text{mic}}\). (2) **Crowd-sourced Bootstrapping.** A crowd-sourced approach (see Figure 3(b)) is where average users conduct bootstrapping on one/more devices, and upload detected \(f_{\text{mic}}\) and \(l_{\text{mic}}\) to TickTock’s server. This information can be utilized when users deploy TickTock. (3) **User-level Bootstrapping.** TickTock’s bootstrapping is conducted by the user (Figure 3(c)) intending to use TickTock on his/her laptop.

**Deployment Scenario.** To use TickTock (Figure 3(d)), the user leverages the bootstrapping information, and sets \(f_{\text{mic}}\) on the TickTock device. Subsequently, the user places the TickTock device on \(l_{\text{mic}}\) to enable TickTock to function as a mic on/off status indicator.

**4 BACKGROUND**

We provide background on the role of clock signals in determining mic status, why they leak, and how their leakage can be detected.

**4.1 Digital MEMS Mics**

Laptops typically contain Micro-Electro-Mechanical systems (MEMS) mics mainly due to their compact form-factor and better noise performance [4, 14]. Amongst them, digital MEMS mics, which are immune to electromagnetic interference (EMI), are a preferred alternative. This is because in laptops, the long cables or PCB traces carrying mic data lines may run close to electromagnetic disturbances such as the laptop’s liquid crystal display [30, 61]. Digital mics sample the analog signal to output data in the form of discrete, high amplitude signals, alternating between the two extreme voltage levels – representing 0 and 1 respectively. As depicted in Figure 4(a), digital mics contain an analog-to-digital converter (ADC) within the mic housing, and the ADC’s operation is driven by an input clock signal. Furthermore, these mics support a wide range of operating clock frequencies from about 512 kHz to 4.8 MHz [23, 25].

**Role of Clock Signals in Mics.** In digital MEMS mics, clock signals function as a control signal that can switch the mic between several power modes. As depicted in Figure 4(b), when the mic is provided with a clock signal in the frequency range around 1 – 4.8 MHz, it enters active mode where it consumes about 0.5 mA of current, and hence is capable of capturing audio [26–28]. On the other hand, when the mic is provided with clock signals whose frequencies are below 250 kHz, the mic enters sleep mode in order to reduce power consumption (≈ 40 µA) [19]. In this work, we identify this difference in clock frequency when the mic is in active mode (i.e., the mic is on), and when in sleep mode (i.e., mic is off), from the EM leakage signals, in order to infer mic’s on/off status.

**4.2 Clock Signals and their Detection**

Clock signals, expressed as voltage in the time domain (Figure 5(a)), are periodic square waves with a fixed time period (denoted by \(T\)), and has a fundamental frequency, \(f_{\text{clk}} = \frac{1}{T}\). When observed as current in the time domain (Figure 5(b)), clock signals are seen as a series of impulses, as the current flows only during a voltage change. Nevertheless, this signal has the same time period, \(T\), and fundamental frequency, \(f_{\text{clk}}\).

Due to their periodic nature, as well as the short rise-time for transition between voltage levels (sub-microseconds), clock signals concentrate their energy in the fundamental clock frequency, \(f_{\text{clk}}\), as well as its odd harmonics (i.e., \(3x, 5x, 7x, \ldots\)) (see Figures 5(c) and 5(d)). Furthermore, if the clock signals spend unequal time in the

---

1 Some mics also support a low-power mode for clock frequencies from 512 kHz to 1 MHz, suitable for wake-word detection in voice-enabled applications.
4.3 Factors of Clock Signal Leakage

As depicted in Figure 7, we identify three potential factors, namely — (a) connectors, (b) cables, and (c) common grounding, that lead to EM leakage of clock signals. For each factor, we explain their causes of leakage based on theory (part (i) in figure), and identify the exact leakage location in a laptop that leaks due to the factor discussed (part (ii)). Subsequently, we capture EM traces from the exterior of the laptop (part (iii)), to finally obtain the EM leakage spectrum containing the mic clock signals (part (iv)). In order to capture the leaked EM signal, we place the near-field probe at the leakage location, and utilize the setup described in Section 5.1. We now explain each leakage factor in detail below.

4.3.1 Leakage (a) – Connectors. Impedance mismatch in connectors is a major contributor to EM emanation. As depicted in Figure 7(a)-(i), when the impedance values of two adjacent elements, e.g., a connector ($Z_{conn}$) and a cable carrying clock signals ($Z_{cable}$) are mismatched, part of the transmitted signal can be reflected and emitted as EM signals. The amount of reflection, or reflection ratio, can be approximated as: $\frac{Z_1 - Z_2}{Z_1 + Z_2}$, where $Z_1$ and $Z_2$ refer to the impedance of the source element and the receiving element, respectively. This reflection ratio is directly proportional to the amount of EM emission. Such EM emission issues occur when circuit designers do not take into account the additional impedance that may be produced on cables while carrying high frequency signals.

In order to confirm the theory, we perform a teardown on Dell Latitude E5570 laptop where the connector is adjacent to the mic. As depicted in Figure 7(a)-(ii), we identify the connector’s location on the right side of the mic. Furthermore, by placing an E-field probe on the laptop’s exterior (Figure 7(a)-(iii)) at the same location, we obtain the EM spectrum with the clock frequency (2.048 MHz) and its harmonics as depicted in the figure (Figure 7(a)-(iv)), confirming that connectors indeed lead to EM leakage.

4.3.2 Leakage (b) – Cables. As depicted in Figure 7(b)-(i), sharp turns in cables and PCB traces change the impedance characteristic of the cable due to difference in the propagation delay resulting from unequal lengths between inner (i.e., $L_{inner}$) and outer sides (i.e., $L_{outer}$) of the PCB traces and cables. Consequently, these unaccounted impedance changes cause impedance mismatch between two sides of the cable (e.g., $Z_{left}$ and $Z_{down}$ as depicted in the figure), leading to EM emissions. We confirm this source of leakage by performing a teardown of a Fujitsu Lifebook in which the microphone cables bend along the top-left corner of the laptop (Figure 7(b)-(ii)). We identify the clock frequencies and their harmonics by placing the near-field probe on the laptop’s exterior at the same location (Figure 7(b)-(iii),(iv)).

Similar to bending of cables, usage of flexible PCBs (or flex cables) for connecting mic board to the audio codec, can result in EM signal leakage due to their flexible nature. While adding grounding copper layers can shield flex PCBs from leakage, such additional makes the PCB rigid, hence ruining their utility [67].

4.3.3 Leakage (c) – Common Grounding. As clock signals have high current slew rate (i.e., $\frac{di}{dt}$), current spikes in mic ground
When the Mic is NEAR

We place the near-field probe (specifically, E-field) at a location of (with the mic on)s

30

SDR) that captures and digitizes the signals in the frequency range of interest, and finally an RPi 4B, executing GNU Radio Companion software, that performs signal processing [1, 2, 10, 13, 37].

3.1 Feasibility Setup

Our setup (Figure 2) consists of – a test device (e.g., laptop), a near-field probe (E-field / H-field) that captures EM leakage signals, connected to a 27 dB wideband RF low-noise amplifier (with an input voltage of 10V DC) to amplify the weak EM signals, which is in-turn connected to an SDRPlay RSP-1A software defined radio (SDR) that captures and digitizes the signals in the frequency range of interest, and finally an RPi 4B, executing GNU Radio Companion software, that performs signal processing [1, 2, 10, 13, 37].

6.1 Design Overview

TickTock leverages mic clock signals from the leaked EM signals in order to serve as a mic on/off status indicator. Recall from §3 that TickTock’s design consist of two phases, namely bootstrapping and deployment phases. Bootstrapping is a one-time phase where we identify the mic clock frequency, $f_{mic}$, as well as the mic clock’s maximum leakage location, $l_{mic}$, of a certain device model. Subsequently in the deployment phase, a user with the same device model, utilizes the identified frequency, $f_{mic}$, and location, $l_{mic}$, in order to predict mic status of his/her device.

5.2 Clock Signals as Mic Status Indicators

To confirm if the clock leakage signals can serve as a mic status indicator, we perform experiments on Dell Latitude E5570 laptop. We place the near-field probe (specifically, E-field) at a location of maximum leakage, i.e., near the connector for this laptop (from Section 4.3.1). As depicted in Figure 8, the mic clock frequencies, i.e., 2.048 MHz and odd harmonics, are present only when the mic is on (Figure 8(a)), and are absent otherwise (Figure 8(b)). This preliminary experiment suggests that presence/absence of clock signals can serve as proxy for indicating mic’s on/off status respectively. In the following sections, we elaborate on how we identify the mic clock frequency and leakage location, as well as perform comprehensive experiments to test the robustness of TickTock.

5 FEASIBILITY STUDY

By means of preliminary experiments, we demonstrate the feasibility of mic clock leakage signals serving as a proxy for mic status.

5.1 Feasibility Setup

Consequently, this results in EM emissions of mic clock frequencies at locations distant from the mic clock lines. In particular, we observe this phenomenon at the location of the WLAN antenna in the top-right corner of the bezel of Dell Latitude E5570 laptop as depicted in Figure 7(c)-(ii).

Figure 7: Figure depicts the cause of leakage, leakage location (internal and external) as well as the captured EM leakage signals (with the mic on) due to three different leakage factors, namely (a) connectors, (b) cables, and (c) common grounding.

Figure 8: Figure depicts the EM leakage spectrum when – (a) the mic is on (i.e., recording on Audacity app), and (b) when the mic is off. The clock frequency (2.048 MHz) and harmonics are present only when the mic is on, hence indicating the feasibility of using EM signals to detect mic on/off status.

6 DESIGN AND IMPLEMENTATION

We now present the design and implementation of TickTock.
Figure 9: Figure depicts the system overview of TickTock. (a) depicts bootstrapping phase where (1) we identify the mic clock frequency, \( f_{\text{mic}} \), as the frequency with maximum number of occurrences (or count) amongst the clock frequencies that only occur when the mic is on; (2) we identify the maximum leakage location, \( l_{\text{mic}} \), as the location where the leakage score due to the mic clock signal is maximum. (b) depicts deployment phase, where a user places the probe at \( l_{\text{mic}} \) identified during bootstrapping, to detect mic on/off status of the device based on the presence/absence of its \( f_{\text{mic}} \).

During bootstrapping (Figure 9(a)), we identify the mic clock frequency, \( f_{\text{mic}} \), by performing two scans (Scans (a) and (b)), in the region near the mic (e.g., laptop’s top bezel) with a near-field probe – once when the mic is on, and the second time when the mic is off. A scan consists of probing multiple locations in a region (e.g., near the mic) and observing the EM signals at each location over multiple time periods. Following the scans, we identify \( f_{\text{mic}} \) as the frequency that occurs uniquely only in the EM signals when the mic is on, and has maximum occurrences compared to all other unique frequencies. Subsequently, in order to identify the maximum leakage location, \( l_{\text{mic}} \), we perform a third scan (Scan (c)), by determining a location with maximum leakage score, which we compute based on the detection of the identified \( f_{\text{mic}} \) and its harmonics.

In the deployment phase (Figure 9(b)), a user places the near-field probe at location, \( l_{\text{mic}} \), identified during bootstrapping, for mic status detection. TickTock predicts that the mic is on, only if the set of detected clock frequencies from the EM signals (i.e., the output of Clock Frequency Detection module in Figure 9), contains exactly one frequency which equals the mic clock frequency, \( f_{\text{mic}} \). Hence, TickTock predicts that the mic is off even when \( f_{\text{mic}} \) is detected along with other spurious frequencies, in order to minimize false predictions. However, TickTock tolerates some error margin (i.e., \( \theta_{\text{margin}} = 10 \text{ kHz} \)) around \( f_{\text{mic}} \), while predicting that the mic is on.

In the following subsections, we address the three main challenges of TickTock: **Challenge 1:** Clock Frequency Detection (§6.2), **Challenge 2:** Mic Clock Frequency Identification (§6.3), and **Challenge 3:** Maximum Leakage Location Identification (§6.4).

### 6.2 Challenge 1: Clock Frequency Detection

One of the main challenges in robustly detecting clock frequencies is the presence of EM noise from neighbouring components or signal lines in the captured EM signals, leading to detection of spurious frequencies. We overcome this issue by detecting their harmonics, in addition to the fundamental frequency. Both bootstrapping and deployment phases utilize this module to take as input the EM leakage signals and output the set of detected clock frequencies.

Figure 10 depicts the module overview. We capture EM signals, or traces, across several frequency spans from the Software Defined Radio (SDR), and compute their spectrum (§6.2.1). Subsequently, we perform mean offset removal (§6.2.2), and identify the peaks in their frequency spectrum based on an amplitude threshold (§6.2.3). We leverage the detected peaks to identify a set of candidate clock frequencies based on the number of harmonics detected as peaks (§6.2.4). Finally, we input the candidate clock frequencies to a pruning stage and eliminate frequencies that are harmonically related to other more likely candidate clock frequencies (§6.2.5).

#### 6.2.1 Signal Capture

In order to capture the EM trace from the SDR, we specify two parameters, namely – center frequency \( f_c \), and bandwidth \( B \). By doing so, we obtain the EM trace information within frequency span, \( [f_c - B/2, f_c + B/2] \). However, as the maximum bandwidth supported by many low-end SDRs may not be sufficient to detect the mic clock frequency and their first several harmonics, we sweep across several \( n_s \) adjacent frequency spans in order to obtain leakage signals with an overall larger bandwidth \( (= n_s B) \). Subsequently, we compute the magnitude spectrum of each span (Figure 10(a)), and stitch all the spans together (here \( n_s = 4 \)), to obtain the overall spectrum of the leaked EM signals.

#### 6.2.2 Mean Offset Removal

We observe that the noise floors of different spans may be different due to different gain values across spans. This is a result of implementation of automatic gain control feature in several SDRs. We perform span-wise mean-offset removal in order to equalize the noise floors across spans (see Figure 10(b)).

#### 6.2.3 Peak Detection

We first obtain the magnitude spectrum of the entire EM trace, we compute a set of peaks in the frequency domain, namely, \( F_p = \{ f_1, f_2, f_3, \ldots \} \) (see Figure 10(c)). The peaks satisfy a minimum amplitude cutoff, \( \theta_a \), and are separated in frequency at least by a distance, \( \theta_d \). The amplitude threshold, \( \theta_a \), varies across devices, depending on the level of leakage of the mic clock signals. However, the distance parameter, \( \theta_d \), is fixed across all devices for both phases \( (= 300 \text{ kHz}) \), which is less than the distance between any two harmonics for any mic clock frequency.

#### 6.2.4 Candidate Clock Frequency ID

Given the set of frequency peaks, \( F_p \), we predict a list of candidate clock frequencies, \( F_c \), and their corresponding set of harmonics, \( H_c \). Recall from §4.2 that clock signals consist of a fundamental frequency \( (1x) \), and harmonics \( (2x, 3x, \ldots) \).
3x, etc.), as peaks in the frequency domain. Hence, for their robust identification, we require detection of a minimum number (i.e., $\theta_0$) of their harmonics (inclusive of the fundamental frequency). By doing so, we prevent prediction of spurious clock signals.

This is straightforward if we assume that fundamental frequency is always detected as a peak. We can iteratively check the likelihood of each frequency peak, $f_i$, to be a candidate clock frequency. For example, consider the peaks identified in Figure 10(d-1), where we compute the likelihood of the first peak (denoted by $\star$), to be a clock frequency. We observe that it has a total of eight harmonics (1x, 3x, ..., 15x), hence, frequency, $f_1$, would be added to the set of candidate clock frequencies, $F_c$ (default value of threshold, $\theta_0 = 4$).

However, the aforementioned approach does not work if the fundamental frequency is missing (see Figure 10(d-2)). In fact, as we show later in §7.2, more than 60% of the EM traces have a missing fundamental frequencies. Hence, we check the likelihood of each peak to not only be the fundamental frequency, but also a harmonic of a potential clock frequency. For example, for the first peak (denoted by $\star$) in Figure 10(d-2), we check for its likelihood to be a third harmonic. By doing so, we indirectly check for the likelihood of the missing fundamental (= $\frac{2}{3}$) to be a candidate clock frequency. In general, we check for each peak’s likelihood to be one of the first $H$ harmonics ($H = 10$), thereby handling the case of not just the missing fundamental, but also its several harmonics. Finally, this module outputs the set of candidate clock frequencies, $F_c$, and their corresponding set of detected harmonics, $H_c$.

6.2.5 Clock Frequency Pruning. We prune the set of candidate clock frequencies, $F_c$, by leveraging their harmonics, $H_c$, to obtain the final set of clock frequencies, $F$, and their harmonics, $H$. We identify frequency pairs, $(f_1, f_2)$, both belonging to the candidate set, $F_c$, such that (1) the set of harmonics of one is a proper subset of the other (i.e., $H_c(f_1) \subset H_c(f_2)$); or (2) the set of harmonics is identical (i.e., $H_c(f_1) = H_c(f_2)$). In both these cases, we eliminate one of the two frequencies (i.e., $f_1$ or $f_2$). Figure 10(e) depicts an example for case (1), where the frequency pair consists of candidate frequencies, $f_1 = 2.048$ MHz, and $f_2 = 6.144$ MHz, where $f_2 = 3 \times f_1$. Clearly, the harmonics of $f_2$, are a subset of the harmonics of frequency, $f_1$, hence we prune the frequency, $f_2$, which is likely a spurious prediction. As an example for case (2), we consider frequencies, $f_1 = 2.048$ MHz, and $f_2 = 1.024$ MHz. All the harmonics of frequency, $f_1$, are also harmonics of the frequency, $f_2$ as, $f_2 = \frac{f_1}{3}$ (e.g., 3x of $f_1$ is 6x of $f_2$). We prune the smaller frequency, $f_2$, as had it been the underlying clock frequency, we would expect to detect intermediate harmonics (e.g., 3.072 MHz) of frequency, $f_2 = 1.024$ MHz, that are not harmonics of frequency, $f_1 = 2.048$ MHz. Finally, after pruning the spurious frequencies, we output the retained clock frequencies, denoted by $F$. Optionally, we also output the number of detected harmonics ($H$), as required in §6.4.

6.3 Challenge 2: Mic Clock Frequency ($f_{\text{mic}}$) ID

As part of the bootstrapping phase, we identify the mic clock frequency, $f_{\text{mic}}$. The main challenge, however, is that its value is device dependent, particularly on the clock frequencies supported by the device audio hardware, hence is not known a priori. To circumvent this problem, we identify $f_{\text{mic}}$ by taking as input the EM leakage signals captured from two scans – Scan (a) when the mic is on, and Scan (b) when the mic is off (see Figure 9(a)). Subsequently, we collect a total of $n_a$ and $n_b$ EM traces, respectively, across different locations around the mic (e.g., laptop’s top bezel). Although the scans are performed over the same region, the number of traces, $n_a$ and $n_b$, can be different. We input each of these traces to the Clock Frequency Detection module (§6.2) to obtain the set of clock frequencies per trace. Note that the number of clock frequencies output can be zero or more, depending on the precise location where the EM trace is captured.
6.3.1 Frequency Aggregation. We now combine the frequencies obtained from all the traces of a particular scan, and output a set of tuples, containing the unique frequencies present and their number of occurrences (i.e., count). In particular, for Scan (a), we obtain the set of tuples, \( T_{\text{on}} \), consisting of \( \{(f^1, c^1), (f^2, c^2), \ldots\} \), where frequency, \( f^i \), indicates a distinct frequency, and the count, \( c^i \), indicates the total number of occurrences of the frequency, \( f^i \). Likewise, for Scan (b), we obtain the set of tuples, \( T_{\text{off}} \). Figure 11(a) depicts this with a toy example with three EM traces. Furthermore, among the tuples of a single scan, we merge frequencies that are present in set, \( T_{\text{on}} \), with the maximum number of harmonics to within an error margin (\( \theta_{\text{margin}} \approx 10 \text{kHz} \)) of each other into a single frequency, by summing up their individual count values.

6.3.2 Mic Frequency with Maximum Occurrence Identification. This module takes as input, the set of tuples, \( T_{\text{on}} \), and \( T_{\text{off}} \), obtained from Frequency Aggregation module, in order to output the mic clock frequency, \( f_{\text{mic}} \). Figure 11(b) depicts how we first identify the frequencies that uniquely occur in set, \( T_{\text{on}} \) (and hence absent in the set, \( T_{\text{off}} \)). Subsequently, we choose \( f_{\text{mic}} \) as the one with the maximum count value among all the unique frequencies (in cases with more than one unique frequency).

We also identify the average leakage amplitude, corresponding to \( f_{\text{mic}} \), by computing the average amplitude of the mic clock signals (i.e., clock frequency and harmonics), in traces where \( f_{\text{mic}} \) is detected. The average leakage amplitude differs across devices, hence is leveraged as a threshold (\( \theta_{l} \)) for successful detection of \( f_{\text{mic}} \) in the Clock Frequency Detection module of all the subsequent stages (see §6.2.3).

At the end of this step, if we fail to identify any unique clock frequency across different scanning locations (e.g., top bezel, bottom bezel, and so on), we conclude that TickTock’s technique does not hold for such a device.

6.4 Challenge 3: Max Leakage Location (\( l_{\text{mic}} \)) ID

This module which is part of the bootstrapping phase, takes as input the EM signals along with their location information, in order to identify the maximum leakage location, \( l_{\text{mic}} \), corresponding to the mic clock frequency, \( f_{\text{mic}} \), and its harmonics.

The main challenge in identifying the EM leakage location is its dependence on the location of underlying leakage sources (e.g., connectors and cables), which in-turn depends on the device’s hardware layout. Additionally exacerbating the problem, the leakage region can be highly localized, i.e., to an area as small as a few cm².

In order to identify \( l_{\text{mic}} \), we perform a third scan (Scan (c), with the mic on, along the same scanning region as in Scans (a) and (b) (Figure 9(a)). We input each EM trace captured at each location, \( l_{\text{loc}} \) (loc to denote location ID step), to the Clock Frequency Detection module (§6.2), and obtain the set of clock frequencies, \( c_{\text{loc}} \), as well as the number of harmonics detected per clock frequency, \( \#H_{\text{loc}} \).

6.4.1 Leakage Score Computation. This module takes as input – detected clock frequencies, \( F_{\text{loc}} \), their corresponding number of harmonics, \( \#H_{\text{loc}} \), as well as the identified mic clock frequency, \( f_{\text{mic}} \), to output a leakage score, \( s_{\text{loc}}^{\text{leak}} \). We compute the leakage score as the number of detected harmonics of \( f_{\text{mic}} \) obtained from the list, \( \#H_{\text{loc}} \). Hence, a location with higher number of detected harmonics for frequency, \( f_{\text{mic}} \), has a higher leakage score. However, if \( f_{\text{mic}} \) is not detected, or if it is detected in addition to other spurious frequencies, we output a leakage score of zero, to indicate the unsuitability of the location for reliable detection of \( f_{\text{mic}} \).

6.4.2 Location with Maximum Score Identification. This module takes as input the set of location and leakage score tuples, i.e., \( \{(l_{\text{loc}}, s_{\text{loc}}^{\text{leak}}), (l_{\text{loc}}, s_{\text{loc}}^{\text{leak}}), \ldots\} \), to output \( l_{\text{mic}} \), with the maximum leakage score, as the best location for probe placement. In the current implementation of TickTock, this module is performed manually, where-in the person performing the bootstrapping process decides the best location, by probing several locations, and identifying the location with maximum score as provided by our system (see Figure 11(c)). However, we highlight that this is a one-time effort, as the bootstrapping phase is performed only once per device. We refer interested readers to - https://bit.ly/3w2QTDA for a video demo on how we user perform this scan.

In general, there could be more than one location with maximum leakage, in which case we choose any one of them as \( l_{\text{mic}} \). On the flip side, if we encounter a device with no suitable locations (e.g., with a score of zero everywhere), this implies that we identified a spurious frequency as \( f_{\text{mic}} \) in the previous step (§6.3), and hence conclude that TickTock’s approach is inapplicable to such a device.

7 EVALUATION

We evaluate TickTock comprehensively on several devices and for various differing conditions, to demonstrate its feasibility.
Figure 12: Figure (a) depicts the brands of the 30 laptops we evaluate, and (b) depicts the release years of the laptops.

Figure 13: Figure depicts the setup of TickTock’s experiment conditions, specifically when varying different conditions.

7.1 Experimental Setup

Apparatus. We utilize the same setup described in §5.1, consisting of the device to be tested (e.g., laptop), a near-field probe, RF amplifier, software defined radio (SDR) and an RPi 4B, for our experiments (see Figure 2). We test each laptop using both the near-field probes, namely the E-field and H-field probes. We also custom-design an amplifier with gain of 27 dB for its low power consumption [37]. We leverage RSP-1A SDR (US$140) that captures signals covering a large portion of radio spectrum, from 1 kHz to 2 GHz, with a maximum bandwidth of 10 MHz [10]. During the detection process, we sweep across four (overlapping) frequency bands to obtain a total bandwidth of 30 MHz (from 0.85 – 30.85 MHz) in order to detect mic clock frequencies and their harmonics.

Data Collection. We evaluate TickTock on a total of 30 laptops of popular brands including Lenovo, Dell, HP and Apple, all released in the last ten years (see Figures 12(a) and 12(b)). For consistency of experiments, we run Ubuntu 20.04 LTS with kernel version 5.11.0-27 on each of the laptops (except Macbooks that run Mac OS X). We record audio at 32-bit 48 kHz using the command-line tool, arecord, unless mentioned otherwise [65]. Furthermore, we ensure that the laptop is plugged into power source and that its screen is active throughout. Furthermore, as depicted in Figure 13, we evaluate TickTock’s performance across different mic (§7.3.1) and non-mic applications (§7.3.2), different audio driver implementations (§7.3.3), its robustness to internal and external EM noise (§7.3.4, §7.3.5), its real-time performance (§7.3.6), the influence due to speaker-access (§7.3.7), as well as the effect of varying sound levels (Appendix A.2 of our extended version [50]). For the evaluation, we perform TickTock’s detection in an offline manner, i.e., we identify the clock signals after capturing all traces (except in §7.3.6). Furthermore, we determine the number of harmonics to be identified (i.e., parameter, \( \theta_h \)) to be three, based on the results from Appendix A.1 of our extended version [50].

Performance Metrics. We define the following three metrics to evaluate TickTock’s overall results. Device Hit Rate refers to the fraction of total devices tested in which TickTock identifies the mic clock frequency, \( f_{\text{mic}} \). Furthermore, we leverage True Positive Rate (TPR) and True Negative Rate (TNR) to evaluate the performance of TickTock in predicting mic status (on vs off) in devices. We consider an EM trace to be a positive example, if TickTock detects \( f_{\text{mic}} \) as the only clock frequency from the EM trace (and negative example otherwise). Hence, we define TPR as the fraction of all traces that are identified to be positive examples, while the mic is on, and TNR as the fraction of all traces that are identified as negative examples, when the mic is off.

7.2 TickTock Performance

We present TickTock’s overall performance by first presenting the results of bootstrapping followed by their performance in determining mic’s on/off status.

7.2.1 Bootstrapping Summary. In Table 1, we present 30 laptops we test, along with their detected \( f_{\text{mic}} \), if any, from the bootstrapping phase. We achieve a device hit rate of 90%, as we successfully identify \( f_{\text{mic}} \) for 27 laptops, with \( f_{\text{mic}} \) ranging between 2.048 – 6.144 MHz. Figure 14 depicts prominent leakage locations, \( l_{\text{mic}} \), observed on laptops that have their mics located on either sides of the webcam on the top bezel. The leakage locations within regions annotated in yellow potentially correspond to locations of connectors (either top/ bottom), while those annotated in blue represent potential locations of cable bends or presence of other peripherals (see §4.3).

Although our approach works well on 90% of the tested laptops, including all tested models from popular vendors such as Lenovo, we present detailed results in Appendix B of our extended version [50] including leakage amplitude, performance due to each probe type and the harmonics detected.
Table 1: Evaluation of TickTock on a total of 30 laptops. TickTock can successfully detect mic clock frequency, $f_{\text{mic}}$, on 27 out of the 30 laptops, i.e., 90% of all tested laptops. ✓ depicts successful, ✗ depicts unsuccessful detection, and ⊗ depicts confounding cases. The A/D column indicates whether the microphone(s) in the laptops are digital (D), analog (A) or unknown (U). We present more detailed results in Appendix B of our extended version [50].

| Device Model       | $f_{\text{mic}}$ (kHz) | Unique Clk? | A/D | Device Model       | $f_{\text{mic}}$ (kHz) | Unique Clk? | A/D | Device Model       | $f_{\text{mic}}$ (kHz) | Unique Clk? | A/D |
|--------------------|------------------------|-------------|-----|--------------------|------------------------|-------------|-----|--------------------|------------------------|-------------|-----|
| ASUS Strix         | 2048                   | ✓           | D   | HP Probook 440 G1  | 2352                   | ✓           | D   | Lenovo X230        | 2048                   | ✓           | D   |
| Asus X450v         | 2048                   | ✓           | D   | HP Zbook Studio G5 | 3072                   | ✓           | D   | Lenovo X250        | 2048                   | ✓           | U   |
| Dell Inspiron 13   | 3072                   | ✓           | D   | Lenovo P14s gen 1  | 2400                   | ✓           | U   | Lenovo X260        | 2048                   | ✓           | D   |
| Dell Inspiron 4549 | 2048                   | ✓           | D   | Lenovo T430U       | 2048                   | ✓           | D   | Razer RZ09-0102    | 2048                   | ✓           | D   |
| Dell Inspiron 7572 | 2048                   | ✓           | D   | Lenovo T460s       | 2048                   | ✓           | U   | Samsung Chronos    | 2048                   | ✓           | U   |
| Dell Latitude E5570| 3072                   | ✓           | U   | Lenovo T470S       | 3072                   | ✓           | U   | Terras Force T5    | 2048                   | ✓           | U   |
| Dell Latitude E7450| 2048                   | ✓           | U   | Lenovo T590        | 2048                   | ✓           | U   | Toshiba Portege    | 6144                    | ✓           | U   |
| Dell XPS L321x     | 2048                   | ✓           | D   | Lenovo X1 Carbon G7| 2400                   | ✓           | U   | Mac Pro 2014 15"   | ✗                      |            | ✓   |
| Fujitsu Lifebook   | 2048                   | ✓           | D   | Lenovo X1 Extreme G3| 2400                   | ✓           | U   | Mac Pro 2017 13"   | 2823                    | ⊗           | U   |
| HP Envy 13         | 3072                   | ✓           | D   | Lenovo X13 Gen 2   | 2400                   | ✓           | U   | Mac Pro 2019 16"   | ✗                      |            | ✓   |

Figure 16: Figure depicts a histogram of the total number of occurrences of different harmonics (as a percentage) among the successfully detected trials across all 27 laptops.

Figure 15: Figure (a) depicts the confusion matrix representing TickTock’s overall detection efficacy, and (b) depicts the individual TPR, for the 27 successful laptops.

Dell, HP and Asus, TickTock fails to detect the mic clock signals in three laptops, all of which are Apple Macbooks. On each of the tested Macbooks, the mics are located either on the left or right side of the keyboard (along the speaker vent), and are connected to the motherboard via short flex cables. We believe that the aluminium enclosure of Macbooks, along with the usage of short flex cables, result in significantly attenuating the leakage signal [47, 68–70]. Of the three laptops, we encounter a confounding case in Macbook Pro 2017 (13’), where a mic clock frequency ($f_{\text{mic}} = 2.823$ MHz) with a low leakage amplitude is detected, although its detection fails to be consistent across different audio recording applications (i.e., clock frequency is absent for some audio recording apps but present for others). We tested ten additional Macbooks using a different setup consisting of a high gain amplifier and spectrum analyzer. However, TickTock is still unable to detect the clock signals consistently. The results are shown in the Appendix B.1 of our extended version [50].

7.3 Differing Experimental Conditions

We evaluate TickTock’s performance over several factors. For this purpose, we choose three representative laptops, Lenovo Thinkpad T430U ($L_{2012}$), Lenovo Thinkpad T470s ($L_{2017}$), and Lenovo X1 Extreme Gen 3 ($L_{2021}$), released in 2012, 2017 and 2021, respectively. We report our results by capturing 650 EM traces over three minutes per device for each differing condition.

7.3.1 Running Mic-based Applications. To evaluate TickTock’s performance in detecting mic on status while capturing audio, we...
report the true positive rate (TPR) obtained on running five applications namely – arecord and Audacity (for recording audio), Zoom (for performing video calls), Cheese (for recording video), and browser-based IBM Watson Speech to Text Service (for transcribing audio). TickTock obtains high TPR for all applications over the three laptops, with a minimum TPR of 98.8% (i.e., $\frac{5.4}{5.11}$) obtained for recording video on laptop, $L_{2021}$ (see Figure 17). These results represent the consistency of TickTock in identifying mic on status.

7.3.2 Running Non-mic Based Applications. To evaluate false triggers during mic off state, we evaluate TickTock by performing everyday tasks (that do not involve the mic) such as taking notes, browsing news, downloading data at high speed (100 Mbps) over Wi-Fi, playing audio and playing video, using five representative applications/tools, namely, Google Docs, Google News, iPerf3, aplay, and YouTube, respectively. From Figure 18(a), we observe that, for the first three tasks, TickTock obtains a TNR of 100% across all laptops. However, for the last two applications involving access to speaker, although the newest laptop, $L_{2021}$, continues to achieve 100% TNR, the older laptops, i.e., $L_{2012}$ and $L_{2017}$, obtain a TNR of less than 2%. We defer the explanation of this result to §7.3.7.

7.3.3 Effect of Different Audio Driver Implementations. We evaluate the effect of different audio driver implementations on TickTock’s performance. We consider drivers that are part of the OS – Ubuntu 16.04 (kernel v4.15), Ubuntu 18.04 (kernel v5.4), Ubuntu 20.04 (kernel v5.11 and v5.13), as well as the driver on Windows 10. We evaluate all laptop-OS combinations, with the exception of laptop, $L_{2021}$, with Ubuntu 16.04, due to lack of compatibility (depicted by a × in the figure). As depicted in Figure 19, for all OSes except Ubuntu 16.04, the TPR and TNR are consistently above 99% across all laptops. However, in the case of Ubuntu 16.04, for laptops, $L_{2012}$, and $L_{2017}$, although the TPR is above 99%, the TNR is close to 0%. On further analysis, we infer that in this driver implementation, the clock signal is always provided to the mic, irrespective of whether the mic is on or off, resulting in a low TNR. We believe that the future Linux driver implementations’ retract clock signals in order to enhance security (i.e., prevent accidental audio capture), while conserving power in laptops. Hence, newer driver versions will likely follow suit, thereby improving TickTock’s accuracy.

7.3.4 Effect of Internal Electromagnetic Noise. We evaluate the effect of electromagnetic noise arising from within the laptop, e.g., due to screen, camera and radio communication. As depicted in Figure 20, we evaluate TickTock’s performance when the mic is on in the background, along with the following six sources of EM interferences – (1) Video Playback: fluctuations in screen content due to high bit-rate video playback, (2) Camera Snaps: photo capture (once every five seconds) from a camera application, (3) Power Interruption: disruption in power (once every five seconds) due to plugging-in and plugging-out of the laptop charging cable, (4) Wi-Fi Download: data download over Wi-Fi at 100 Mbps using iPerf3, (5) Bluetooth Music: music playback via Bluetooth; and (6) USB Keyboard: serial communication via USB to capture keyboard input.
As depicted, we obtain a TPR above 98% for all the three laptops across all scenarios. This good performance can be attributed to be the spatial specificity of the near-field probes to capture EM leakage in a highly localized region (i.e., within a few centimeters).

7.3.5 Effect of External Electromagnetic Noise. Recall that the mic clock frequencies and their harmonics are in the lower MHz range, i.e., from 1 – 30 MHz. Hence, we evaluate TickTock in the presence of EM noise within our capture range, particularly from a radio frequency identification (RFID) reader, RFID-RC522, with a center frequency of 13.56 MHz. We test the effect on TickTock’s TPR by varying the distance of the reader from the near-field probes, i.e., the E-field probe or the H-field probe (with 5 mm loop diameter), placed on the laptops. For this experiment, we consider three laptops, namely Lenovo Thinkpad T430U, Dell Latitude E5570, and Lenovo X1 Extreme Gen 3, that are capable of detecting mic clock frequencies with both the above mentioned probes.

As depicted in Figure 21, we observe that the E-field probe remains unaffected in the presence of the RFID reader, by achieving an average TPR of 98.5% (across the three laptops) even at the closest distance of 1 cm. This is because the RFID readers create a magnetic field in the near-field region, and hence not influencing the E-field probe. On the other hand, we observe that the H-field probe is severely affected at close distances, with TickTock achieving an average TPR of 1.1% at a distance of 1 cm. However, we observe that the TPR increases with distance, hence at a distance of 9 cm, the average TPR increases to a high value of 99.43%. Figure 22 depicts the frequency spectrum of EM leakage for one of the laptops (i.e., Lenovo Thinkpad T430U) with the reader placed at distances of 1 cm, 5 cm, and 9 cm from the H-field probe. We observe that at the closest distance (i.e., 1 cm), the EM noise is broadband, i.e., covers a wide frequency band, and hence completely masks the underlying clock signals. However, as the distance increases, the frequency range of the noise decreases, leading to a more accurate detection of the harmonics of the mic clock frequency.

7.3.6 Real-time Performance. Recall that in the evaluations presented so far, we conduct TickTock’s detection in an offline manner, i.e., we compute the clock signals present in the trace, after all the EM traces are collected. In this evaluation, we test the practicality of TickTock by performing the detection in real-time. In particular, we compute the TPR by varying the detection time (i.e., time taken to output a prediction of mic status) from 0.15 – 2 seconds. We vary this indirectly by varying the rate at which we read from the SDR. Furthermore, we report the average detection time as its value depends both on the frequency-switching rate of the SDR hardware as well as TickTock’s computation time, both of which may change.

As depicted in Figure 23, TickTock achieves a high TPR of 97.5% and 99.7%, for average detection times of 0.26 s and 0.5 s, respectively, demonstrating the feasibility of TickTock as a real-time mic status indicator. For average detection time lower than 0.26 s, the increase in data read-rate results in significant data overflows from the SDR, and hence results in reduction in the TPR to as low as 0.27%, for an average detection time of 0.15 seconds.

7.3.7 Influence due to Speaker Access. We also test for potential false triggers that may result in mic clock frequency detection when the speaker is on. This is because the mic’s ADC and speaker’s DAC clock lines may be shared, especially if they are both controlled by the same audio codec IC. As this property of sharing clock lines is hardware-dependent, we perform this evaluation on all the 27 laptops. In Figure 24, we depict the true negative rate (TNR) of all laptops across Release Years.

Figure 21: Figure depicts the mean (denoted by dots) and standard deviation (denoted by error bars) in TPR for EM signals captured from the near-field probes in the presence of EM disturbance from RFID readers.

Figure 22: Figure depicts the EM leakage spectrum obtained when the RFID reader is (a) 1 cm, (b) 5 cm, and (c) 9 cm away from the H-field probe, while (d) depicts a case without an RFID reader (i.e., no noise). When the reader is 1 cm away, EM noise overshadows the mic clock signals, while the noise drops considerably as the distance increases to 9 cm.

Figure 23: Figure depicts the effect of different detection times on TickTock’s performance.

Figure 24: Figure depicts the TNR while accessing speakers for the 27 different laptop models, sorted by release year.
laptops, sorted chronologically by release year. We observe that in 20 out of the 27 laptops, access to speaker also triggers the same clock frequency, hence resulting in a low TNR of 26.2% on average across all laptops. However, we notice a significant increase in TNR, to an average of 83.4%, for all laptops released on or after 2019 (which includes 5 Lenovo and 1 HP laptops). This increase in TNR over the last three years seems promising, hence we believe TickTock has increased utility for laptops of the upcoming years.

8 DISCUSSION
We present important points related to TickTock’s detection.

**Mic Status Detection on Non-Laptop Devices.** We evaluate TickTock on 40 non-laptop devices, including smart phones, tablets, smart speakers and USB web-cameras. We report detailed findings in Appendix C of our extended version [50]. To summarize, we successfully detect mic clock frequency in 21 out of 40 devices. Of the successful devices, we obtain an average TPR and TNR of 86.2% (σ = 22.5%) and 100% (σ = 0%), respectively. We note three key reasons for TickTock’s lower performance on non-laptop devices:

- **Analog vs. Digital Mics:** Some smartphone models contain analog mics instead of digital mics. We believe there are several reasons for future devices to transition to digital mics: Digital mics—(1) host an ADC, hence require fewer components to function, (2) are highly integrable into systems only containing general purpose ICs as they output digital data; (3) are immune to EM interference compared to analog mics, hence robust to noise. Finally, (4) digital mics are known to be easier to design [4, 61].

- **Devices without Power Constraints:** Voice-enabled smart speaker devices (including Google Home and Echo Dot) do not have any power constraints as they are always plugged-in, and may not cut-off the clock frequency even when not recording. We observe such cases in four out of eight tested smart speakers.

- **Compact Form-Factor:** Devices with compact form-factors, e.g., smartphones enclose shorter cables (compared to laptops), and likely cause reduced EM leakage in lower radio frequencies [42].

**Miniaturizing TickTock’s Form-Factor.** TickTock’s current prototype (Figure 2) consists of a variety of components stacked to the side of the laptop, while our vision is a device with a small USB drive form-factor that can be placed in contact with the laptop’s exterior (Figure 1). One approach to reduce overall setup size is to leverage SDRs with smaller dimensions. Hence, we evaluate TickTock with different SDRs such as AirSpy HF+ (with small form-factor ~ 45 × 60 × 10 mm), and achieve high TPR above 98% (refer to Appendix D of our extended version [50]). Another approach would be to redesign the whole setup into a single printed circuit board, consisting of the RF amplifier, a high sampling ADC (50-60 Mps), as well as the controller IC which runs TickTock’s logic [20–22, 62].

**Absence of Leakage due to Clock Signals.** TickTock’s technique relies on the EM leakage from clock signals due to imperfection in hardware design including impedance mismatch at connectors, cables. Hardware designers are constantly improving the emissions from clock signals in their circuits, by incorporating techniques such as differential signaling, spread spectrum clocking, and reduction in trace length, in addition to physical methods such as shielding with metal [29, 36, 41, 51]. However, none of these approaches are foolproof, as they can only reduce the amount of leakage. As an example, metal shields around cables typically have slits to serve as heat vents, which can in-turn radiate EM signals in certain frequency ranges, subject to the dimensions of the slit.

9 RELATED WORK
We now present closely related work with TickTock.

**Eavesdropping Detection.** Researchers have proposed hardware and software-based approaches to detect mic eavesdropping [12, 38, 40, 57, 58, 63]. One of the works utilizes SDRs to detect audio bugs in the environment based on their wireless transmissions [63]. However, none of these approaches apply to detect eavesdropping mics in laptops. One representative work amongst the software-based approaches proposes a system for trustworthy mic-usage notification by inserting run-time checks in the kernel/hypervisor [39]. However, unlike these approaches, TickTock is resistant to kernel/hypervisor compromise, and its detection can easily be extended to work on devices with different specifications, e.g., different OSes.

**Acoustic Jamming.** One line of work explores generation of audio jamming signals – both audible and inaudible, in order to prevent mics in commodity devices from capturing meaningful audio [7, 31, 32, 53, 54, 71]. In particular, one of the works engineered an ultrasound array in a wearable bracelet form-factor that produced inaudible jamming signals to prevent any attacker device from recording [7]. Our work is complementary to these works in that TickTock detects eavesdropping mics, while they disable them.

**Electromagnetic Side Channels.** There are several attacks leveraging electromagnetic leakage signals to infer cryptographic keys, screen content, passcodes, USB data, neural network architecture, and even capture audio [5, 8, 24, 33, 34, 60]. However, unlike all the above, our work leverages the leaked EM signals for defense, rather than attack. One particular work utilises leaked electromagnetic signals from local oscillators to identify wireless eavesdroppers [6]. However, unlike this work that detects Wi-Fi receivers, our approach detects audio receivers, i.e., mics.

10 CONCLUSION
We present TickTock, a novel laptop mic on/off status detection, based on EM leakage of clock signals. We design and implement TickTock, as well as perform real-world evaluation on 30 popular laptops and observe mic detection in 27 laptops. Through this work, we explore a novel direction of utilizing EM side-channel information as part of a defense, rather than an attack. As part of future work, we hope to utilize TickTock to identify access to other sensors including cameras and inertial measurement unit (IMU) sensors.

11 ACKNOWLEDGEMENTS
We thank Wang Guench, Nitya Lakshmanan, Siddharth Rupavatharam and Niel Warren for valuable discussions and/or feedback on our paper. This work is supported by the Singapore Ministry of Education Academic Research Fund (R-252-000-B48-114), the Yonsei University Research Fund (2021-22-0337), the Institute of Information and Communications Technology Planning and Evaluation (IITP-2022-0-00420) grant funded by Ministry of Science and ICT (MSIT) in Korea, and the Google PhD Fellowship 2021.
[54] Nirupam Roy, Sheng Shen, Haitham Hassanieh, and Romit Roy Choudhury. 2018. Inaudible Voice Commands: The {Long-Range} Attack and Defense. In 15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18). 547–560.

[55] Jennifer Schlesinger. 2014. Inside Blackshades: Hackers are watching you on infected webcams. https://www.cnbc.com/2014/05/23/inside-blackshades-hackers-are-watching-you-on-infected-webcams.html.

[56] Katherine Schwab. 2019. The hardware trend Google and Apple are throwing their weight behind. https://www.fastcompany.com/90349731/the-hardware-trend-google-amazon-and-apple-are-throwing-their-weight-behind.

[57] Rahul Anand Sharma, Elahe Soltanaghaei, Anthony Rowe, and Vyas Sekar. 2022. Lumot: Identifying and Localizing Diverse Hidden {IoT} Devices in an Unfamiliar Environment. In 31st USENIX Security Symposium (USENIX Security 22). 1095–1112.

[58] Akash Deep Singh, Luis Garcia, Joseph Noor, and Mani Srivastava. 2021. I Always Feel Like Somebody’s Sensing Me! A Framework to Detect, Identify, and Localize Clandestine Wireless Sensors. In 30th USENIX Security Symposium (USENIX Security 21). 1829–1846.

[59] Do Son. 2018. 14-year-old hacker developed Fruttfly backdoor surveillance. https://securityonline.info/14-year-old-hacker-developed-fruttfly-backdoor-surveillance-mac-users-over-13-years/.

[60] Yang Su, Daniel Genkin, Damith Ranasinghe, and Yuval Yarom. 2017. {USB} snooping made easy: crosstalk leakage attacks on {USB} hubs. In 26th USENIX Security Symposium (USENIX Security 17). 1145–1161.

[61] Manuel Tagliavini. 2018. MEMS microphones generate new buzz, face resonant future. https://omdia.tech.informa.com/OM001889/MEMS-microphones-generate-new-buzz-face-resonant-future.

[62] Linear Technology. Accessed 2022. LTC1744 14-Bit, 50MSPs ADC Datasheet. https://www.analog.com/media/en/technical-documentation/data-sheets/1744f.pdf.

[63] Veronica Valeros and Sebastian Garcia. 2017. Spy vs. spy: A modern study of microphone bugs operation and detection. Chaos Computer Club-eV (2017).

[64] Stevens Vaughan-Nichols. 2020. Dell enables Linux hardware privacy controls. https://www.zdnet.com/article/dell-enables-linux-hardware-privacy-controls/.

[65] Paul Winkler. Accessed 2022. Arecord Command Line Tool. https://linux.die.net/man/1/arecord.

[66] Fahmida Y. 2020. Enterprise Laptops Still Vulnerable to Memory Attacks. https://duo.com/decipher/enterprise-laptops-still-vulnerable-to-memory-attacks.

[67] Milan Yogendrappa. Accessed 2022. Best EMI and RF Shielding Methods for Flex PCBs. https://www.protoexpress.com/blog/emi-rf-shielding-methods-flex-pcb/.

[68] YouTube. Accessed 2022. Macbook Pro 2014 15" Teardown. https://www.youtube.com/watch?v=N8qZiJnrH1M.

[69] YouTube. Accessed 2022. Macbook Pro 2017 13" Teardown. https://www.youtube.com/watch?v=aYXYJpA0eEY.

[70] YouTube. Accessed 2022. Macbook Pro 2019 15" Teardown. https://www.youtube.com/watch?v=a1YxyjA0eEY.

[71] Hongzi Zhu, Xiaosong Wang, Yi Jiang, Shan Chang, and Xudong Wang. 2021. Secure Voice Interactions with Smart Devices. IEEE Transactions on Mobile Computing (2021).