GhostType: The Limits of Using Contactless Electromagnetic Interference to Inject Phantom Keys into Analog Circuits of Keyboards

Qinhong Jiang*, Yanze Ren*, Yan Long†, Chen Yan*§, Yumai Sun†, Xiaoyu Ji*, Kevin Fu‡, Wenyuan Xu*
*Zhejiang University, †University of Michigan, ‡Northeastern University
{qhjiang, yzren, yanchen, xji, wyxu}@zju.edu.cn, {yanlong, yumai}@umich.edu, k.fu@northeastern.edu

***Abstract*—Keyboards are the primary peripheral input devices for various critical computer application scenarios. This paper performs a security analysis of the keyboard sensing mechanisms and uncovers a new class of vulnerabilities that can be exploited to induce phantom keys—fake keystrokes injected into keyboards' analog circuits in a contactless way using electromagnetic interference (EMI). Besides regular keystrokes, such phantom keys also include keystrokes that human operators cannot achieve, such as rapidly injecting over 10,000 keys per minute and injecting hidden keys that do not exist on the physical keyboard. The underlying principles of phantom key injections consist in inducing false voltages on keyboard sensing GPIO pins through EMI coupled onto matrix circuits. We investigate the voltage and timing requirements of injection signals both theoretically and empirically to establish the theory of phantom key injection. To validate the threat of keyboard sensing vulnerabilities, we design GhostType that can cause denial-of-service of the keyboard and inject random keystrokes as well as certain targeted keystrokes of the adversary's choice. We have validated GhostType on 48 of 50 off-the-shelf keyboards/keypads from 20 brands, including both membrane/mechanical structures and USB/Bluetooth protocols. Some example consequences of GhostType include completely blocking keyboard operations, crashing and turning off downstream computers, and deleting computer files. Finally, we glean lessons from our investigations and propose countermeasures, including shielding keyboards with metal materials and enhancing the keystroke sensing mechanism.**

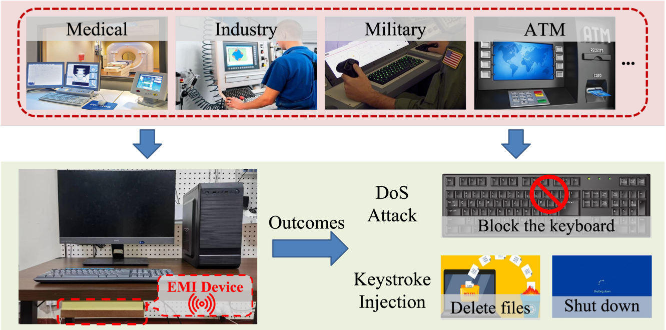


Fig. 1: Keyboards are widely used in medical, industry, military, ATM, and other applications. Exploiting the vulnerabilities of the keyboard sensing mechanisms, `GhostType` can perform DoS attacks to block the keyboard or inject random keystrokes and certain targeted keystrokes.

I. Introduction

Keyboard has been an indispensable input component of any computer setup since the 1970s [11]. As a fundamental peripheral input device, keyboard has been an object of security research for decades. The majority of studies focused on how to eavesdrop on the typed keystrokes (also known as keylogging) and their countermeasures [1]–[4], [6], [8], [19], [29], [31]–[33], [35], [36], [39], [42], [45], [46], [51], [55]. Others have tried injecting malicious fake keystrokes with reprogrammed USB devices masquerading as keyboards [5], [23], [40], [43]. To prevent these fake keystrokes from directly manipulating computers, keyboards are recommended to be vetted or authenticated in security-sensitive applications [7], [10], [18], [20], [26], [41], [53]. Our study[1] revisits keyboard security and asks one more fundamental question: *to which degree can we trust the keystroke sensing of unaltered legitimate keyboards?*

Trustworthy keystroke sensing lays the foundation to secure computer operations in various critical application scenarios, including medical [17], industry [25], military [9], ATM [52], etc. Untrusted keystroke inputs could disrupt the operation of downstream computers and result in unexpected consequences. However, the security of keystroke sensing mechanisms has hardly been investigated by the security community. The main reason is that keyboards are known for their high reliability, especially in comparison with the touchscreen alternatives, which have been demonstrated to be vulnerable to electromagnetic interference (EMI) [37], [49], [56]. Keyboards sense keystrokes based on a simple principle—a keypress turns on/off a physical switch and therefore changes the received voltage level indicator which is usually 3.3 or 5 V. The high voltage level is naturally more resistant to conductive and radiative interference and keyboards are normally designed and tested for electromagnetic compatibility (EMC). In addition, the long history of keyboard manufacturing has given birth to false keystroke-prevention designs such as debounce and anti-ghosting mechanisms. These factors seem to suggest a reduced attack surface of malicious exploits.

[§] Corresponding author

[1] Demos: https://sites.google.com/view/ghosttype-demo

Network and Distributed System Security (NDSS) Symposium 2024
26 February - 1 March 2024, San Diego, CA, USA
ISBN 1-891562-93-2
https://dx.doi.org/10.14722/ndss.2024.23015
www.ndss-symposium.org

Our work aims to perform a security analysis of the overarching keystroke sensing mechanisms on modern keyboards and keypads. Specifically, we investigate whether unaltered keyboards/keypads may sense adversary-controlled fake keystrokes other than the authentic physical keystrokes from human inputs. As illustrated in Fig. 1, if an adversary is able to inject fake keystrokes into a legitimate keyboard without touching it, she may stealthily manipulate the computer by disrupting normal user operations, deleting documents, shutting the computer down, etc., depending on the specific keystrokes that can be injected by the adversary.

Security Analysis of Keyboards. To investigate the feasibility and limits of such a threat, we disassembled 15 off-the-shelf keyboards to retrieve the characteristics of matrix circuits and scanning signals through reverse engineering. After overcoming the challenges of understanding diverse and complex implementations of these keyboards, we managed to uncover three shared vulnerabilities that could be exploited for keystroke injections:

- The simple keystroke scanning signals are not verified when received by the processor on all keyboards. We are able to inject *every key* on the keyboard by replaying the scanning signals into the sensing circuit.
- Keyboards are vulnerable to EMI signals at specific frequencies despite the existing EMC design, with which we are able to cause *denial-of-service (DoS) of the keyboard* or *inject fake keystrokes contactlessly*.
- All keyboards have "*hidden keys*" that do not exist in the physical layout but only exist in the keyboard matrix circuit. We are able to inject non-existing functional keys such as sleep, open file/web browser, and media control.

Contactless Keystroke Injection via EMI. Motivated by these observations, we further explore the principles behind the above phenomenon and investigate the voltage and timing requirements of injection signals both theoretically and empirically to establish the theory of effective keystroke injections. Additionally, we investigate the feasibility of injecting targeted keystrokes. To assess the level of potential real-world threats of keyboard sensing vulnerabilities, we design GhostType (GT in short), the first contactless keystroke injection attack against the keystroke sensing mechanisms based on EMI. GhostType achieves two types of attack outcomes:

- *DoS attacks* can completely block the sensing of authentic keystrokes and thereby disable user operations.
- *Keystroke injection* can inject random keystrokes to make the computer unresponsive and even crash, or inject certain targeted keystrokes of the attacker's choice.

Injecting targeted keystrokes is the most challenging as it requires a significant amount of reverse engineering and delicate design of the injection signals to achieve localized delivery of EMI with correct timing. We achieved localized delivery by selecting proper relative antenna-to-trace positions so that fake keystrokes can be injected to *only one certain* RX. To address the timing problem, we designed a synchronization-free injection method by understanding the keyboard's two-stage scanning mechanism and designing signals that trigger the keyboard's rescanning period. We evaluated GhostType on 50 off-the-shelf keyboards and keypads, including both membrane/mechanical structures and USB/Bluetooth protocols and found 48 out of them vulnerable to GhostType. Notable results include injecting hidden alphabetical and function keys on a numeric-only keypad, injecting random keystrokes at a maximum of 22,939 keys per minute to force the computer to sleep constantly, and injecting targeted keystrokes to delete/-close files and turn off the computer. Finally, we summarize security insights and propose hardware and software countermeasures. In summary, our contributions include:

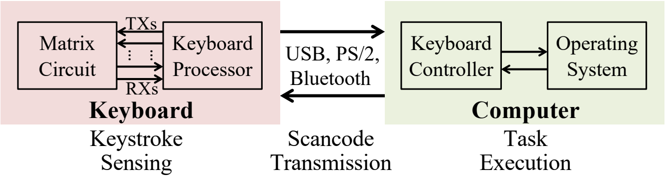


Fig. 2: Illustration of how a keyboard works. A keystroke is sensed through matrix scanning and encoded in a "scancode" to be transmitted to the computer via standard protocols. The computer then executes the corresponding tasks.

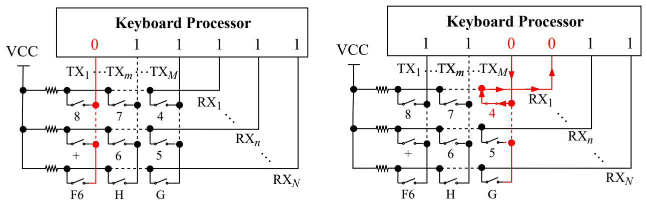
- To the best of our knowledge, we present the first signal integrity analysis of keystroke sensing mechanisms. Our analysis reveals vulnerabilities that could be exploited to manipulate keyboards and downstream computers.
- We develop the theory of contactless keystroke injections via EMI. Our theory and experiments show that adversaries can achieve synchronization-free attacks capable of blocking the keyboard, injecting random keystrokes to cause computers to crash, and injecting targeted keystrokes to delete files and turn off computers.
- We provide the assessment and analysis of the vulnerabilities on 50 off-the-shelf keyboards and insights for potential countermeasures gleaned from our investigations.

II. Background and Threat Model

A. Keyboard Overview

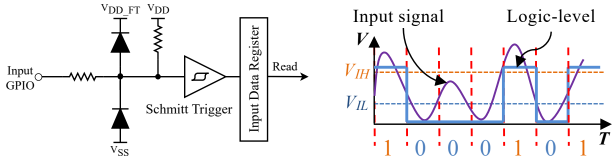
Keyboards are the most prevalent computer input device. There are several types of keyboards, including membrane, mechanical, dome, capacitive, buckling-spring, hall-effect, and optical keyboards. Membrane keyboards have been the most popular since the mid-1990s because they are cheap and easy for mass production. Keyboards can have different numbers of keys depending on the vendor and model, with most keyboards having 80 to 110 keys.

The typical workflow of a keyboard consists of three steps: keystroke sensing, scancode transmission, and task execution. As shown in Fig. 2, the keyboard processor employs the scanning algorithm to scan the matrix circuit and sense keystrokes. Each key is assigned a unique identifier called a "scancode" with a translation table stored in the keyboard processor's memory. When the processor detects a key being pressed, it compares the key's coordinate on the matrix circuit to the scancode translation table. It then reports the scancode to the host computer via standard communication protocols such as PS/2, USB, and Bluetooth. After receiving the scancode, the computer raises an interrupt to process the scancode and



(a) (b)

Fig. 3: (a) The keyboard arranges switches/keys in a grid-like array. (b) When a key is pressed, a closed circuit is formed, and a corresponding RX is dropped to the logical-low state.



(a) (b)

Fig. 4: (a) The structure of GPIO input pin. (b) The Schmitt Trigger determines the input logic state with two thresholds, V_{IH} and V_{IL}.

register a keystroke. Finally, the operating system (OS) passes the keystroke information to applications.

B. Keyboard Sensing Mechanism

Matrix Circuit Scanning. The keyboard is often designed in a special architecture known as the matrix circuit, as shown in Fig. 3(a). The matrix circuit is built by arranging switches/keys in a grid-like array of M scanning lines (TXs) and N receiving lines (RXs) with one switch/key at every intersection. There is no established standard for the design of the matrix circuit layout, and a matrix circuit with M TXs and N RXs can support up to $M * N$ keys in theory. Each RX is pulled up to remain in the logical-high state ("1") in the idle state, and the keyboard processor drops each TX to the logical-low state ("0") in sequence to scan the matrix circuit. When a key is pressed, as shown in Fig. 3(b), the circuit of the corresponding TX-RX pair is closed. The scanning signal on TX is received by RX, resulting in RX being dropped to the logical-low state.

Keystrokes Sensing. The majority of keyboards sense keystrokes on the principle of detecting the logical state on the input GPIO. The keyboard processor employs a Schmitt Trigger at the input GPIO to determine the input logic state, as shown in Fig. 4(a). The Schmitt Trigger determines the input logic state by applying two threshold voltages: the high threshold voltage V_{IH}, and the low threshold voltage V_{IL} (Fig. 4(b)). The keyboard processor detects a key as pressed when an RX is dropped below the low threshold voltage V_{IL} when a TX is scanned. The generic values for V_{IH}, and V_{IL} are 2.0-2.5 V and 1.2-1.5 V for a 5 V system, 1.2-1.5 V and 0.6-0.8 V for a 3.3 V system respectively. The exact thresholds depend on the processor's electrical characteristics.

Capability of Handling Simultaneous Keystrokes. *Key Rollover* is the term used to describe how many keys can be pressed simultaneously. A keyboard with n-key rollover (NKRO) can correctly detect and handle all keys being pressed simultaneously. Typical general-purpose keyboards are 3-KRO to 6-KRO, and gaming keyboards usually support NKRO. *Keyboard Ghosting* is the problem that some keyboard keys don't work when multiple keys are pressed simultaneously. This happens when three or more keys sharing rows and columns are pressed simultaneously, and the connected circuit permits the current to flow incorrectly. Keyboards typically use filtering logic to detect and block keystrokes before this happens in software or employ diodes at each key to prevent the incorrect current flow in hardware.

C. Known Keyboard Vulnerabilities

Previously, keyboards have mostly been investigated for keystroke logging side-channel attacks that aim to eavesdrop on keystroke inputs and violate keyboard confidentiality (Section VII). A few works analyzed how to inject keys using BadUSB attacks [23], [40]. Such attacks exploit keyboard USB's firmware vulnerabilities by reprogramming a connected USB device with malicious software that allows the device to masquerade as a malicious USB keyboard. However, countermeasures already exist for authenticating USB devices and detecting BadUSB attacks to ensure the legitimacy of connected USB devices [7], [10], [18], [20], [26], [41], [53]. Meanwhile, the security of keyboard sensing mechanisms has rarely been explored yet. Our work aims to characterize an orthogonal space of keyboard data integrity vulnerabilities by investigating the underlying sensing mechanisms of keyboards that are independent of data transmission protocols such as USB and Bluetooth.

D. Threat Model

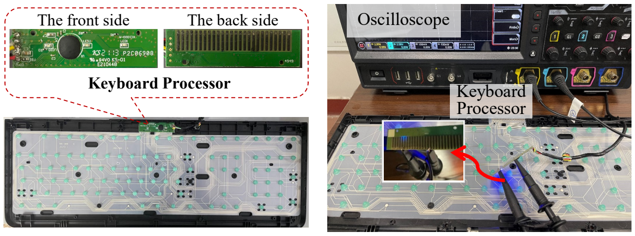
Adversary's Goal. The adversary aims to contactlessly inject keystrokes into a keyboard through intentional electromagnetic interference (IEMI), thus blocking keyboard inputs or input keys to manipulate the connected computer. Our work considers two types of attack outcomes:

(1) **Denial-of-Service (DoS) Attack**, where the adversary can completely block the sensing of authentic keystrokes to disable user operations.
(2) **Keystroke Injection**, where the adversary can inject random keystrokes to make the computer unresponsive and even crash, or inject certain targeted keystrokes of the attacker's choice.

We make the following assumptions for the adversary to achieve the aforementioned attack outcomes:

Capability of the Adversary. We assume it is only feasible for the attacker to inject keys using external EMI signals contactlessly. This happens when the attacker has no on-site controls over the target keyboard's hardware/software and cannot take apart or tap into the keyboard or physically connect a malicious USB device in the form of BadUSB.

Knowledge of the Victim Keyboard. We assume the adversary knows the target keyboard's model, and she may obtain



(a) (b)

Fig. 5: (a) The internal structure of a membrane keyboard. (b) The experimental setup for reverse-engineering the scanning characteristics.

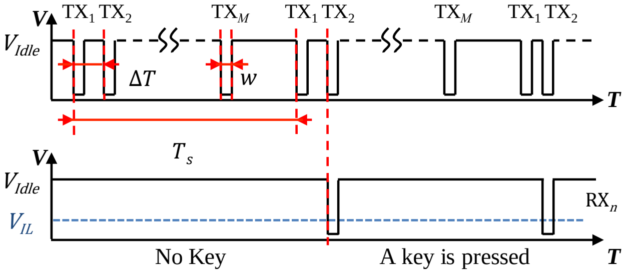


Fig. 6: The keyboard processor continuously pulses each TX for a short duration in sequence, and the scanning signal on the TX flows through the switch to RX when a key is pressed.

a similar keyboard for assessment beforehand. For example, she may disassemble the keyboard to systematically analyze the matrix circuit and scanning characteristics to retrieve the specifications by reverse engineering.

Attack Setup. We assume the adversary can hide the injection equipment by attaching it under the keyboard's desk or placing it at a distance from the keyboard. We also assume the adversary can control the equipment remotely.

III. Keyboard Sensing Security Analysis

In this section, we perform a systematic security analysis of 15 off-the-shelf keyboards through reverse engineering and uncover three vulnerabilities of keyboard sensing mechanisms.

A. Vulnerabilities of Matrix Scanning

We disassemble 15 off-the-shelf membrane keyboards. Fig. 5(a) shows an example of their internal structures, including a keyboard processor board and a three-layered plastic matrix circuit board. The keyboard processor board is linked to a USB cable with a magnetic ring to shield high-frequency electromagnetic interference. The processor's GPIO pins are connected to traces on the matrix circuit board through physical contact.

1) Keyboard Sensing Characteristics Revealing: We use an oscilloscope to monitor the signal on each GPIO pin of the keyboard processor in Fig. 5(b). The signals on TX and RX without and with a key pressed are illustrated in Fig. 6. The signal on each RX remains high in the idle state when no key is pressed, while the scanning signal on each TX is a pulse signal with width w and scanning period T_S. Thus, we first determine whether it is TX or RX by measuring whether there is a pulse or DC signal on each GPIO pin. The results in Table I indicate that the most commonly used keyboard matrix circuit employs 18 TXs and 8 RXs. We then measure each keyboard's scanning characteristics, including idle state voltage V_{Idle}, pulse width w, scanning period T_S and time difference ΔT between two adjacent TXs. The results are summarized in Table I, indicating that the scanning characteristics vary with the keyboard vendor and model. *We then hypothesize that the keyboard processor may be spoofed by replaying the scanning signal into an RX according to the following two reasons:* (1) the processor determines whether there is a key by sensing RX's logic state without authenticating the received signals, and (2) the scanning signal is a simple negative pulse signal that is not encrypted.

2) Wired Keystroke Injection: To validate our hypothesis above, we wire into an arbitrary RX and utilize a signal generator to inject a pulse signal with the same scanning characteristics as revealed in Table I. Several keys were successfully injected, indicating that the keyboard processor does not validate the authenticity and legitimacy of the received signals. According to the keystroke sensing mechanism in Fig. 4(b), keystrokes can be injected by dropping the RX's voltage to the low threshold V_{IL} when a TX is scanned. Thus, we change the values of the replayed pulse signal's amplitude V_{in}, period T_{in}, and pulse width w_{in} to test more diverse injection signals. We first decrease the amplitude V_{in} of the injection signal in a 0.1 V step from V_{Idle}. The result shows that keystrokes can be injected when V_{in} satisfies Eq. (1) to drop the voltage at an RX across the GPIO's low threshold voltage V_{IL} introduced in Fig. 4. For example, keystrokes can be injected into Cherry KC1000 keyboard and Logitech MK235 when V_{in} is higher than 3.4 V and 3.6 V, respectively.

$$V_{Idle} - V_{in} \leq V_{IL} \tag{1}$$

We then change the value of injection period T_{in} and pulse width w_{in}. The results indicate that keystrokes can be injected only when T_{in} satisfies Eq. (2).

$$T_{in} = \frac{T_s}{k}, k \in \mathbb{N}^* \tag{2}$$

This is because keyboards usually employ a *debounce delay* to ensure only one signal is acted upon each key-down or key-up event to prevent spurious keystrokes, i.e., a key press/release is only determined to be a keystroke if it is detected by two consecutive scanning cycles. Thus, the injected signals must hold for at least two consecutive scanning cycles. Additionally, keystrokes are injected at a higher speed when increasing the value of k. We also notice multiple keystrokes are injected simultaneously when we increase w_{in}, and keystrokes are injected occasionally or even not injected when we decrease w_{in}. The keyboard is blocked when the number of injected keystrokes exceeds the keyboard's key rollover capacity (Section II).

B. Vulnerabilities of Contactless Keystroke Injection

1) Potential Coupling Path for EMI Injection: The keyboard matrix circuit board in Fig. 5(a) is a three-layered plastic sheet board with dense traces exposed on the upper and lower sheets and cavities at each key location on the middle

TABLE I: Characteristics of matrix circuits and scanning signals retrieved through reverse engineering.

Vendor & Model	**Num. of Keys (TXs, RXs)**	**Scanning Characteristics**			
		V_{Idle}	T_S	w	ΔT
Cherry KC1000	108 (18,8)	5 V	3.6 ms	15 us	200 us
ACER YKB913	104 (18,8)	5 V	4.0 ms	120 us	120 us
ACER KM41-2K	104 (18,8)	3.3 V	8 ms	35 us	35 us
A4TECH MK100	104 (18,8)	5 V	3.8 ms	110 us	130 us
A4TECH FG1010	98 (18,8)	3.3 V	2.4 ms	45 us	130 us
Logitech MK235	104 (12,11)	3.3V	4.0 ms	8.5 us	10 us
Logitech MK220	100 (12,11)	3.3 V	4.0 ms	8.0 us	11 us
Rapoo K150	104 (18,8)	3.3 V	8.2 ms	142 us	250 us
Rapoo X125S	104 (18,8)	3.3 V	7.9 ms	140 us	250 us
Dell KB522P	116 (18,8)	5 V	3.2 ms	10 us	30 us
Dell KM2123D	104 (18,8)	3.3 V	7.8 ms	120 us	160 us
Lenovo KM4800S	107 (18,8)	5 V	7.8 ms	230 us	250 us
BOW MK610	79 (16,8)	3.3 V	7.2 ms	180 us	300 us

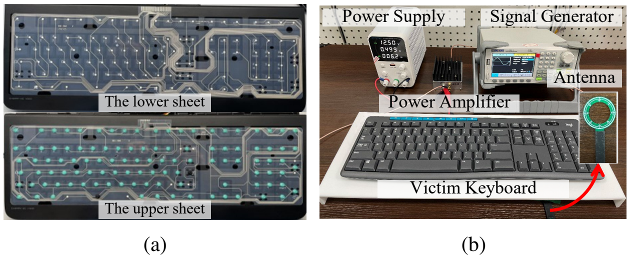


Fig. 7: Illustrations of (a) the traces on the upper and lower sheets, and (b) the experiment setup of contactless keystroke injection via EMI. The keyboard is placed on a 5 mm-thick acrylic sheet, and the antenna is hidden under the sheet.

sheet. These TX and RX traces are irregular and vary from keyboard vendor and model, as illustrated in Appendix A. An example of the 16 TX traces lower sheet and 8 RX traces upper sheet is shown in Fig. 7(a). Although keyboards are consumer electronics that are expected to have adequate EM shielding, we found no anti-interference design on the matrix circuit board for almost all keyboards except the magnetic ring of the USB cable protecting the USB instead of the sensing circuit. *We hypothesize these long exposed traces can be exploited as potential EM coupling paths for EMI injections.*

2) Feasibility Study of Contactless Keystroke Injection: We conducted a frequency sweep test on a Cherry KC1000 keyboard to test the hypothesis. The experiment setup is illustrated in Fig. 7(b). We employ a signal generator (SIGLENT SDG6032X), a power amplifier for EMI signal generation, and an antenna for EMI signal transmission. We place the antenna under the keyboard matrix circuit and conduct a frequency sweep test with a sinusoidal signal from 10 MHz to 100 MHz with a step of 10 MHz and an amplitude of 1 Vpp. During the test, we can randomly inject keystrokes into the keyboard at 30, 50, 60, 70, and 80 MHz *only* and block the keyboard at 20, 90, and 100 MHz. These results demonstrate that the matrix circuit traces act as the EM coupling path for contactless keystroke injections, resulting in different attack outcomes.

C. Vulnerabilities of Hidden Keys

During the experiments, we observed an interesting phenomenon: keys that don't exist on the keyboard's physical layout are injected. We call these keys "hidden keys". For example, we injected several hidden keys on a Cherry KC1000 keyboard, including function keys to open the file browser, turn the volume up/down, make the media play/previous, and possible ASCII codes for debugging such as "171", "233", "255", etc.

Prerequisites of Hidden Keys. We found that the hidden key phenomenon occurs because the keyboard matrix circuit is designed with a key at the intersection of TX and RX without a physical switch. The key sets of keys on the matrix circuit and the keyboard's physical switches are M_p and M_k, respectively. Theoretically, M_k should be equal to M_p, but in practice, M_k is a proper subset of M_p and $M_p - M_p \cap M_k \neq 0$, which is the prerequisites of hidden keys. The keyboard processor could handle all the input keys in M_k, and the set of hidden keys M_{hidden} can be expressed as $M_{hidden} = M_p - M_k$. We believe that hidden keys exist due to keyboard designers' negligence in inspecting and removing the non-existent keys from the keyboard processing firmware. This could be because it is more cost-effective for manufacturers to develop one matrix for various products. Under normal circumstances, these hidden keys will not be triggered because a human cannot close a nonexistent switch. However, the adversary could inject every key on the keyboard matrix circuit to trigger hidden keys, which may cause unexpected consequences to the downstream OS system and software. We evaluated the full spectrum of hidden keys on 10 keyboards and discussed potential threats of hidden keys in Section V-E.

D. Summary of Observed Vulnerabilities

We summarize the following three observed vulnerabilities of the keyboard sensing mechanisms based on the above security analysis and keystroke injection experiments:

- **Observation 1**: The keystroke scanning signals are simple and not verified after being received by the processor. We are able to inject *every key* on the keyboard by replaying the scanning signals in the sensing circuit.
- **Observation 2**: The long and dense traces on the keyboard matrix circuit are coupling paths for EM injections. We are able to inject keystrokes contactlessly via EMI.
- **Observation 3**: We are able to inject hidden keys that do not exist in the keyboard's physical layout but only exist in the matrix circuit.

IV. Contactless Keystroke Injection via EMI

After verifying the feasibility of contactless keystroke injections via EMI, we investigate the following two research questions to design GhostType.

Q1: How to design the injection signals to achieve effective contactless keystroke injections? Although our preliminary study confirmed the feasibility of contactless injections, keystrokes were injected inefficiently. To characterize more effective attacks, we investigate the voltage and timing requirements of injection signals to establish the theory of contactless keystroke injections in Section IV-A.

Q2: How can adversaries inject a targeted keystroke? Even if we can inject keystrokes effectively, targeted keystroke

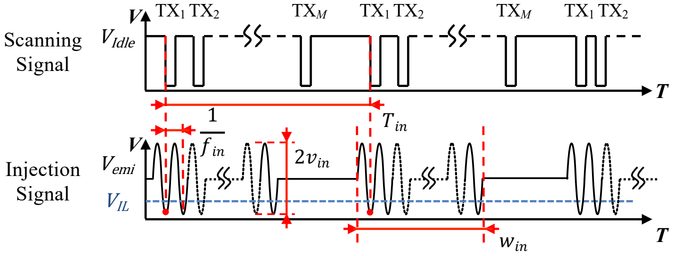


Fig. 8: The injection signal designed for effective keystroke injections is a pulse-modulated sinusoidal signal with frequency f_{in}, amplitude v_{in}, pulse width w_{in} and period T_{in}.

injection is challenging as it requires a significant amount of reverse engineering and delicate design of the injection signal for localizing delivery of injection signals into the targeted RX and synchronizing the injection with the targeted TX. We explore the strategies to overcome the challenges and assess the feasibility of such injections in Section IV-B.

A. *Effective Keystroke Injection via EMI*

Fig. 8 illustrates the injection signal we designed for `GhostType`, where four parameters can be configured, including frequency f_{in}, amplitude v_{in}, pulse width w_{in} and period T_{in}. We design the injection signal as a pulse-modulated sinusoidal signal for two reasons: (1) the pulse-modulated sinusoidal signal is the most commonly used in state-of-the-art EMI injections [12]–[15], [21], [22], [47], [56], [57], and (2) the feasibility of changing the reading of GPIO pins by injecting sinusoidal signals has been demonstrated in [12], [47], [57] and our preliminary study in Section III-B. As briefly mentioned in Section III-A, keystrokes can be injected when the injection signal satisfies two constraints:

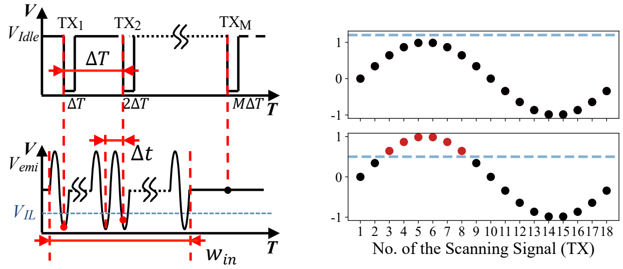
- **Constraint 1:** The induced sinusoidal voltage $V_{emi}(t)$ at an RX needs to drop below V_{IL} when a TX is scanned to be sensed as a keystroke.
- **Constraint 2:** The injection signal needs to be injected during at least two consecutive scanning cycles because of the debounce mechanism.

To understand how to satisfy these constraints, we investigate the voltage and timing requirements of the injection signal to establish the theory of effective keystroke injections via EMI. First, we analyze the requirements of frequency f_{in} and amplitude v_{in} to inject keystrokes effectively. Then, we analyze the requirements of width w_{in} and period T_{in} to perform the single- and multiple-keystroke injections.

1) Requirements of frequency f_{in} and amplitude v_{in}: For a sinusoidal signal with frequency f_{in} and amplitude v_{in}, the induced voltage coupled into the keyboard's RX is an AC signal $V_{emi}(t)$ that varies with time t, which can be expressed as Eq. (3).

$$V_{emi}(t) = E_c v_{in} sin(2\pi f_{in} t + \varphi_0) \tag{3}$$

where E_c is the coupling efficiency since the injection signal is coupled into the victim matrix circuit by means of the magnetic coupling mechanism. And the signal S_{in} in Fig. 8 can be



(a) (b)

Fig. 9: (a) The timing relationship between the injection and scanning signal, where $\Delta T = kt_{in} + \Delta t$. (b) Illustration of the requirements of the injection signal in Eq. (7): the blue dashed line represents the successful injection threshold (the right-hand side of Eq. (7)), and the red and black dots represent successful and failed keystroke injections.

expressed as Eq. (4).

$$S_{in} = \begin{cases} V_{emi}(t) & kT_{in} < t \le w_{in} + kT_{in} \\ 0 & otherwise \end{cases} \tag{4}$$

where S_{in} is a pulse-modulated sinusoidal signal and T_{in} is the period of the injection signal and $k \in \mathbb{N}$.

To meet the first constraint to inject a keystroke when the m-th TX is scanned, the voltage requirement of the injection signal $V_{emi}(m\Delta T)$ can be expressed as Eq. (5) by combining Eqs. (1) and (3).

$$E_v v_{in} sin(2\pi f_{in} m\Delta T + \varphi_0) \ge V_{Idle} - V_{IL} \tag{5}$$

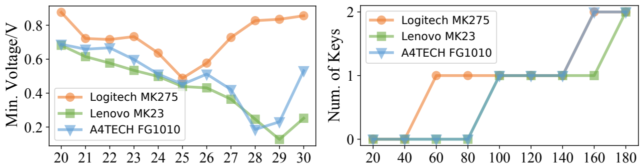
where $m\Delta T$ represents the scanning time of the m-th TX, $m = 1, 2, ..., M$, $V_{Idle} - V_{IL}$ is a constant, and M is the number of TXs. The specific values of V_{Idle} and V_{IL} are keyboard-dependent, which can be measured through reverse engineering. Since the frequency of injection signal f_{in} (several MHz) is more than three orders of magnitude greater than the frequency of the scanning signal f_s (several kHz), the timing relationship between the attack and each TX's scanning signal is shown in Fig. 9(a), which can be expressed as Eq. (6).

$$\Delta T = kt_{in} + \Delta t, 0 \le \Delta t < t_{in} \ and \ k \in \mathbb{N} \tag{6}$$

Substitute Eq. (6) into Eq. (5) and simplify, the voltage requirement of the injection signal can be expressed as Eq. (7).

$$sin(2\pi f_{in} \Delta tm + \varphi_0) \ge \frac{V_{Idle} - V_{IL}}{E_c v_{in}}, \ m = 1, 2, ..., M \tag{7}$$

where kt_{in} vanishes because $f_{in} t_{in} = 1$, and $0 \le f_{in} \Delta t \le 1$. Thus, $sin(2\pi f_{in} \Delta tm + \varphi_0)$ becomes a discrete function of m. Fig. 9(b) illustrates the meaning of Eq. (7) on a keyboard with 18 TXs when $\varphi_0 = 0$. The solid dots represent the injected voltage on the RXs when a TX is scanned, and the blue dashed line represents the threshold $(V_{Idle} - V_{IL})/E_c v_{in}$. Since $max(sin(\cdot))$ = 1, the sinusoidal signal has no intersection with the blue dashed line when $V_{Idle} - E_c v_{in} > V_{IL}$, i.e., Eq. (7) is unsolvable and there is no keystroke injected. When $V_{Idle} - E_c v_{in} \le V_{IL}$, the blue dashed line gradually moves down, and more keystrokes are injected as v_{in} increases. Thus, the minimum injection voltage is $v_{in} = V_{Idle} - V_{IL}$. The red dots in Fig. 9(b) represent successful keystroke injections



(a) Injection Frequency/MHz (b) Pulse Width/us

Fig. 10: (a) The minimum voltage v_{in} required for keystroke injections at different frequencies f_{in}. (b) The number of simultaneously injected keys with different pulse widths w_{in}.

when the corresponding TX is scanned. Besides, we can change the value of φ_0 to inject keystrokes from different TXs. Faraday's law of induction states that the coupling efficiency E_c strongly depends on the injection signal frequency f_{in}. Thus, we must choose an appropriate combination of f_{in} and v_{in} to satisfy Eq. (7) to inject keystrokes contactlessly.

Prior works usually try to maximize E_c by analyzing the resonant coupling frequency f_{res}. However, analyzing f_{res} is both difficult and unnecessary for our attack for the following two reasons: *(1) Difficult:* The resonant coupling frequency f_{res} is determined by the geometry of traces and the keyboard's matching network impedance, which is difficult to calculate theoretically because traces are complex and different on each keyboard, and there are no publicly available high-frequency keyboard processor models. *(2) Unnecessary:* Keystrokes can be injected at a wide range of frequencies, not just one. Although the coupling efficiency E_c fluctuates with frequency, it can be easily overcome by increasing the amplitude v_{in}. Therefore, we can sweep the frequency across a wide band, determine the injection frequency candidates $\{f_1, f_2, \dots\}$ for relatively high power transfer, and then increase v_{in} to satisfy Eq. (7). It is important to note that the high-power EMI attack is power-hungry and may interfere with other devices to make them easily detectable. Thus, to make the injection energy-efficient and undetectable, attackers must select optimal injection frequency candidates with higher E_c to perform keystroke injections with a relatively low v_{in}.

We validate this by conducting a frequency sweep experiment on three keyboards and changing the amplitude v_{in} at different injection frequencies to determine the minimum required amplitude under a successful keystroke injection frequency. The results in Fig. 10(a) show that keystrokes can be injected at a wide range of frequencies and the minimum required amplitude varies in a non-linear pattern at different frequencies, indicating that some frequencies are more advantageous to an EMI injection than others. We can choose these more advantageous frequency candidates to perform a more powerful keystroke injection. The effect of f_{in} and v_{in} on keystroke injection behaviour is further evaluated in Section V-C.

To meet the second constraint, Eq. (5) and Eq. (8) must be satisfied simultaneously.

$$V_{emi}((m+M)\Delta T) \geq \frac{V_{Idle} - V_{IL}}{E_c v_{in}} \tag{8}$$

where M is the number of a keyboard's TXs. Keystrokes are continuously injected when satisfying Eq. (9).

$$sin(2\pi f_{in}\Delta t m + \varphi_0) = sin(2\pi f_{in}\Delta t(m+M) + \varphi_0) \tag{9}$$

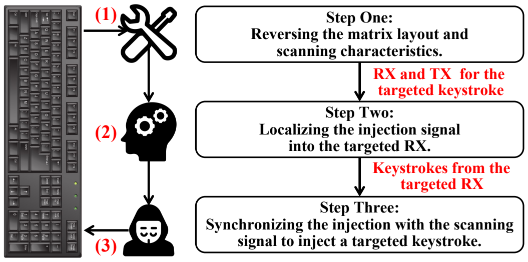

(1)
(2)
(3)
Step One:
Reversing the matrix layout and
scanning characteristics.
RX and TX for the
targeted keystroke
Step Two:
Localizing the injection signal
into the targeted RX.
Keystrokes from the
targeted RX
Step Three:
Synchronizing the injection with the scanning
signal to inject a targeted keystroke.

Fig. 11: Targeted keystroke injection requires three steps: (1) retrieve the coordinate of targeted key $key(m, n)$, (2) localize the injection signal into the n-th RX, and (3) synchronize the injection with the m-th TX.

When $\Delta t = 0$, i.e., $\Delta T = kt_{in}$, Eq. (9) holds. The injection constraint of the debounce mechanism is automatically satisfied. When $\Delta t \neq 0$, a sufficient and necessary condition to satisfy Eq. (9) is

$$f_{in} \cdot \Delta t \cdot M = C, \ C \in \mathbb{N} \tag{10}$$

Combining Eqs. (6) and (10), t_{in} and ΔT need to satisfy the relationship in Eq. (11) to inject a keystroke.

$$\frac{\Delta T}{t_{in}} = k + \frac{C}{M}, \ k \in \mathbb{N}^* \tag{11}$$

Since ΔT is two orders of magnitude greater than t_{in}, many solutions exist for Eq. (11). This conclusion is further demonstrated in Table V, where keystrokes can be injected at a wide range of frequencies with the 48 off-the-shelf keyboards. When $V_{emi}(m\Delta T) \neq V_{emi}((m+M)\Delta T)$, keystrokes can only be injected periodically because Eq. (5) and Eq. (8) can be partially satisfied simultaneously in a sinusoidal signal period, which is inefficient and not the goal in this paper.

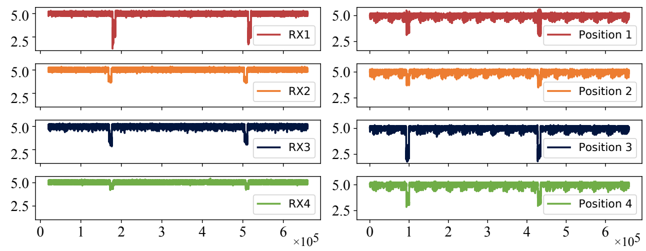
2) Requirements of width w_{in} and period T_{in}: We investigate the requirements of w_{in} and T_{in} to perform single- and multiple-keystroke injections. The red dots below the blue threshold dash line in Fig. 9(a) illustrate successful keystroke injections at different TXs. We can configure the value of width w_{in} and period T_{in} to change the number of injected TXs. w_{in} can be expressed as Eq. (12) and T_{in} satisfies Eq. (2).

$$w_{in} = kt_{in}, \ k \in \mathbb{N} \tag{12}$$

where k is the cycle of a sinusoidal wave. When $T_{in} = T_s$, single and multiple keystrokes can be injected by satisfying Eq. (13) and Eq. (14), respectively.

$$Single\ Keystroke: \ w \leq w_{in} < \Delta T \tag{13}$$
$$Multiple\ Keystrokes: \ w_{in} \geq w + k\Delta T \tag{14}$$

where ΔT is the time difference between two adjacent TXs and $k \in \mathbb{N}$. When $T_{in} = T_s/k$ and $k = 2, 3, \dots$, multiple keystrokes are injected . We validate this by conducting keystroke injections on three different keyboards. The results in Fig. 10(b) demonstrate that we can configure the value of w_{in} to inject single or multiple keystrokes simultaneously. The larger w_{in} is, the more keystrokes are injected simultaneously.



(a)

(b)

Fig. 12: The injected signals measured on (a) different RXs with the same antenna position, and (b) an RX with different antenna positions P_{in}.

B. Feasibility of Targeted Keystroke Injection

The core idea of injecting a targeted keystroke $key(m, n)$ involves injecting the injection signal into n-th RX while the m-th TX is scanned. There are three required steps in this injection pipeline, as illustrated in Fig. 11: (1) acquiring the victim keyboard's matrix circuit to determine the targeted RX and TX and scanning characteristics, (2) localizing the injection signal into the targeted RX to inject keystrokes from the targeted RX line and (3) synchronizing the injection with the targeted TX to inject a targeted keystroke. To perform a targeted keystroke injection, it is important to address the following two technical challenges:

- **Challenge 1: Localized delivery of injection signals into the targeted RX.** The dense traces on the matrix circuit make it challenging to localize EMI signals to *only one certain* RX without interfering with other RXs.
- **Challenge 2: Synchronizing the injection with the targeted TX.** Synchronizing with the victim device is always required by the most state-of-the-art EMI injections on a specific target.

We investigate the opportunities to handle the aforementioned challenges to inject a targeted keystroke.

1) Opportunities for RX selection: The dense and irregular matrix circuit traces make it challenging to inject signals into *only one certain* RX without interfering with other RXs. Generating an injection signal with a small focusing area is a common challenge in EMI injections [12], [38], [49]. Nevertheless, we discover that the susceptibility to EMI injections differs substantially among traces when injecting the same injection signal at the same position, as illustrated in Fig. 12(a). Although the injection signal is coupled into different RXs, there is a significant difference in the amplitude of the coupled signals on different RXs. Similarly, as demonstrated in Fig. 12(b), there is also a significant difference in the amplitude of the coupled signals measured on the same RX with different antenna positions. Our simulations in Ansys HFSS in Appendix B also demonstrated that the relative antenna-to-trace position has a significant impact on the trace's susceptibility to EMI injections. Thus, we envision that it is possible to make the coupled sinusoidal voltage V_{in} satisfy the requirement in Eq. (7) on *only one certain* RX.

We conduct keystroke injections on a Cherry KC1000 keyboard at twenty-one antenna positions and analyze the injected keystrokes' RX distributions to validate this strategy. These twenty-one evenly distributed antenna positions $P_1, \cdots, P_{21}$ are illustrated in Appendix C. We perform injections at each position for 30 s with an amplitude of 1 Vpp and different frequencies f_{in}. The frequency f_{in} is swept from 10 MHz to 50 MHz in 10 MHz steps for 105 sets of tests. The results in Fig. 13 demonstrate that the relative antenna-to-trace position significantly impacts the RX distributions of injected keystrokes, i.e., the injected keystrokes at a certain position are mostly distributed over a specific RX. For example, we can inject keystrokes from the 1-th RX and the 6-th RX at P_3 and P_{11}, respectively. Thus, we can localize the injection signal into the targeted RX by determining the injection position along the targeted RX trace.

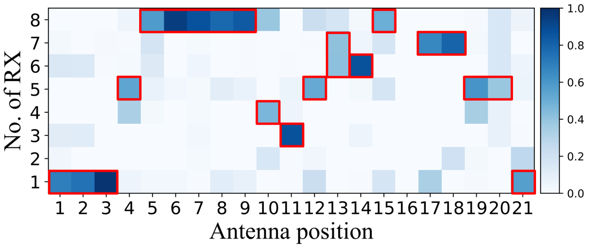
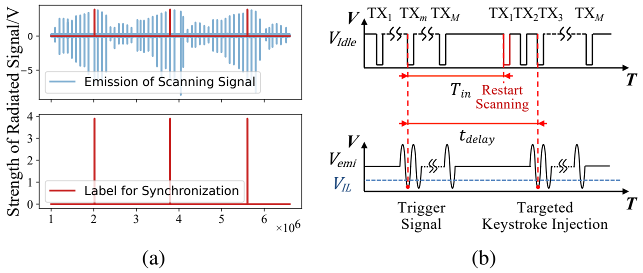


Fig. 13: RX distributions of the injected keystrokes at 21 antenna positions.

2) Opportunities for TX selection: We investigate the opportunities for TX selection by synchronizing the injection with the scanning signal or employing a synchronization-free injection strategy.

Synchronizing with the Scanning. The keyboard scanning process unintentionally emits electromagnetic signals [55], which can be received to synchronize the injection with the scanning. The adversary can employ an antenna and low noise amplifier (LNA) like [21], [56] to receive and amplify the radiated scanning signals from victim devices. Fig. 14(a) (top) shows an example of the received emission on a Cherry KC1000 keyboard, from which the timing information of the scanning signals can be extracted. For example, we can observe 17 TXs between the two strongest emissions, corresponding to the Cherry KC1000 keyboard's 18 TXs. The strength of the radiated scanning signals changes between TXs because of the varying TX-trace-to-antenna positions. The adversary can synchronize the injection with the scanning signal utilizing any TX's emission as a flag to synchronize the injection with the scanning signal and adjust the timing of the injection signal into the targeted TX based on the scanning sequence. Fig. 14(a) (bottom) illustrates an example of utilizing the strongest emission as the flag for synchronization. We also demonstrated in Appendix D that the EM emission characteristics used for synchronization are the same for two keyboards of the same model.

Synchronization-free Injection Strategy. In addition to this synchronization strategy, we discover that certain keyboards employ a two-stage scanning mechanism, which can be exploited to build a synchronization-free injection strategy. The keyboard processor is configured in an interrupt-driven method by default at the first stage to detect a key press to save energy. When a key press is detected, the processor *restarts the scanning* and switches to the second scanning stage to



(a)

(b)

Fig. 14: (a) The strategy for locating the strongest emanation as the flag for synchronization. (b) The strategy of synchronization-free injection with the scanning state transition triggered by the detection of a key press.

detect which key is pressed. The specifics of this two-stage scanning mechanism are detailed in Appendix E. Based on this two-stage scanning mechanism, we develop a synchronization-free strategy to inject a targeted keystroke without sensing the emanations for synchronization, as illustrated in Fig. 14(b). We first inject a trigger signal $V_{emi}(t_{trig})$ to trigger the scanning restart. Based on our previous analysis in Section IV-A, the trigger time t_{trig} needs to satisfy Eq. (15) in order for the trigger signal to be identified as a key press.

$$t_{trig} = w + \Delta T \tag{15}$$

We can then adjust the delay time t_{delay} to determine the injection timing to inject a targeted keystroke, where $t_{delay} = T_S + (m-1)\Delta T, m = 1, 2, 3, ..., i$. We validate this synchronization-free strategy with the same experiment setup as Section IV-B1. The results show that we can inject the function key “PrintScrn” at P_3 in Fig. 13 when $t_{trig} = 215\ us$ and $t_{delay} = 800\ us$, and inject the function key “sleep” at P_{11} when $t_{trig} = 215\ us$ and $t_{delay} = 600\ us$. Furthermore, this synchronization-free strategy can also be applied to all Bluetooth keyboards since they don't scan the matrix until a key press is detected to save energy.

C. Attack Prerequisites for Different Attacks

The attack prerequisites are determined by the type of attack, and our investigations show that targeted keystroke injection attacks have more prerequisites than others.

For DoS attack and random keystroke injection, the adversary is required to determine the appropriate combination of injection frequency and position to perform the injection effectively. Our evaluations in Sections V-C and V-D will demonstrate that these attacks can succeed at a wide range of injection frequencies and positions.

For targeted keystroke injection, the adversary needs to localize the delivery of injection signals into the targeted RX and synchronize with the keyboard via its EM emission, or she can employ a synchronization-free attack strategy if the keyboard supports a two-stage scanning mechanism or is a Bluetooth keyboard.

V. Evaluation

In this section, we evaluate GhostType on 50 commercial keyboards across different manufacturers, models, structures, and protocols. We then explore the factors affecting injection performance, including the antenna position, injection distance, table material, and thickness. Finally, we evaluate hidden keys on 10 keyboards and elaborate on potential attack scenarios and case studies.

A. Experiment Setup

Apparatus. We use a similar experimental setup as Fig. 7 by default if not otherwise specified, which consists of a signal generator, a power amplifier, and a near-field antenna. We explain the specific setups at the beginning of each subsection for the experiments that evaluate performance with various factors.

Targeted Devices. We evaluate GhostType on 50 off-the-shelf keyboards and keypads from 20 popular brands released in the last five years, including Acer, Dell, HP, Lenovo, Logitech, Microsoft, Philips, etc., as shown on the website in [16]. The specifications of these keyboards are presented in Table V. There are 40 membrane and 10 mechanical keyboards, with 35 USB and 15 Bluetooth keyboards.

Metrics. We define two evaluation metrics to evaluate GhostType's overall performance:

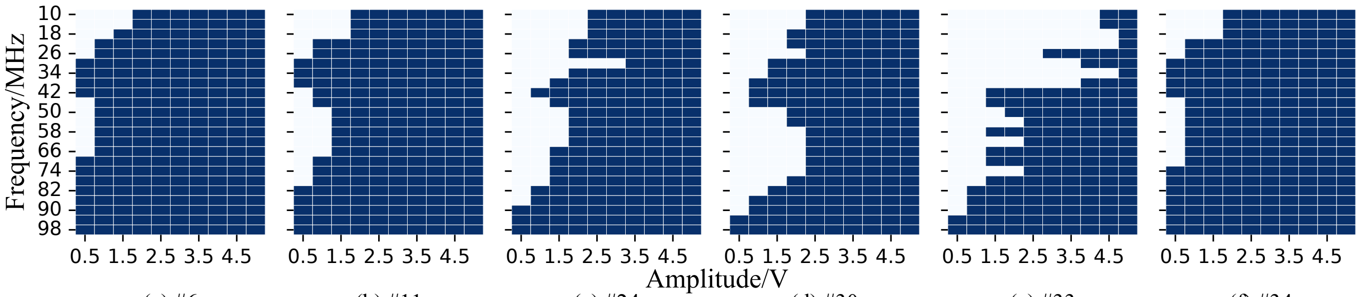
- *Success Rate (SR)* is the percentage of attacks that successfully achieve the targeted attack outcomes.
- *Actions per Minute (APM)* is the number of injected keystrokes per minute to evaluate the injection speed.

B. Different Keyboards

We placed the injection antenna directly under the keyboard and conducted a frequency sweep experiment from 10 MHz to 100 MHz with a step of 0.1 MHz with an amplitude of 1 Vpp. We present the results of injection frequencies at which each keyboard is vulnerable to GhostType.

1) Overall Performance: The results are shown in Table V, indicating that 48 of these 50 keyboards are susceptible to GT attacks. We disassembled these two unsusceptible keyboards and found steel plates underneath the matrix circuit as shown in Appendix F, which we believe were acting as additional EMI shielding for these two keyboards.

DoS Attack. We manually pressed the keyboard to determine if the keyboard was blocked during the injection. The results show that the DoS attack is successful on 36 of the 48 vulnerable keyboards. The detailed injection frequencies are reported in Table V. We found that DoS attack can be conducted in a wide range of frequencies rather than a specific frequency, validating our theory established in Section IV. These 12 unsuccessful keyboards are all mechanical or membrane gaming keyboards that are NKRO and can correctly handle all keys being pressed simultaneously and hence will not be blocked by numerous keystrokes injected simultaneously. We evaluated the success rate of DoS attack with 30 repeated trials on these 36 keyboards. The results are displayed in Table V, indicating a success rate of nearly 100%.



(a) #6 (b) #11 (c) #24 (d) #30 (e) #33 (f) #34

Fig. 15: Results of GhostType's performance on DoS attack over different combinations of injection frequency f_{in} and amplitude v_{in}, where the solid blue areas represent successful attacks.

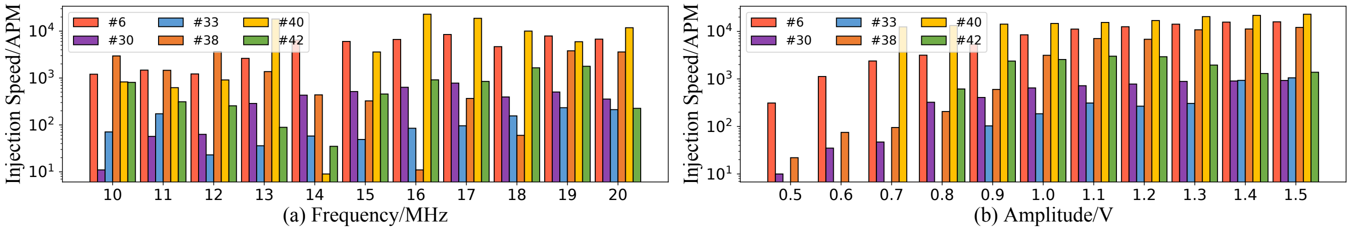


Fig. 16: GhostType's performance on keystroke injections over different injection frequencies f_{in} and amplitudes v_{in}.

Keystroke Injection. We ran a keystroke monitoring program on the computer to record the injected keystrokes during the injection. The results show that we can inject fake keystrokes via EMI on 39 of the 48 vulnerable keyboards. The detailed range of injection frequencies for each keyboard is reported in Table V. We found that keystrokes can be injected at over 10,000 APM on mechanical and membrane gaming keyboards, *which is significantly faster than what human users can type*. For example, GT can inject 12,110 keystrokes per minute into a membrane gaming keyboard #38 and 22,939 into a mechanical keyboard #40. We evaluated the success rate of random injection with 30 repeated trials on each device. The success rate is also nearly 100%. Additionally, we investigated the feasibility of injecting targeted keystrokes for these 39 keyboards using the strategy established in Section IV-B. The number of targeted keystrokes that could be injected on each keyboard is reported in Table V. We believe a well-sourced adversary can implement a targeted keystroke injection attack to achieve a damaging attack outcome. We validate the threat of targeted keystroke injection with two case studies in Section V-F and further discuss the challenges of targeted keystroke injection implementation in Section VI.

2) Summary: The above results show that the uncovered vulnerabilities of the keyboard sensing mechanisms are widespread, and each keyboard has a different level of susceptibility to GT due to its unique design. In addition, we had the following three insights:

Membrane vs. Mechanical: Membrane keyboards are vulnerable to both keystroke injection and DoS attacks, while mechanical keyboards are only vulnerable to keystroke injection attacks because of their NKRO capacity.

USB vs. Bluetooth: Bluetooth keyboards are more vulnerable to GT attack than USB keyboards. This is because Bluetooth keyboards are often powered by 3.3 V batteries so the required injection amplitude v_{in} to satisfy Eq. (7) is relatively lower than 5 V-powered USB keyboards.

Vendor vs. Security: The uncovered vulnerabilities of sensing mechanisms are not specific to individual keyboard vendors. Keyboards are vulnerable to GT in a wide range of EMI frequencies. Higher-end keyboards are typically superior in performance but not security. The adversary can inject keystrokes at a higher speed on high-end keyboards than on the more common office keyboards from the same vendor.

C. Injection Frequencies and Amplitudes

To evaluate the impact of injection frequencies and amplitudes on GhostType, we conducted experiments using a similar setup as Fig. 7. The injection antenna is placed at the same position under the key "5" on the numeric keypad.

1) DoS Attack: We selected six representative keyboards from Table V. Three keyboards (#11, #24, and #34) are vulnerable to DoS attack, and three (#6, #30, and #33) are vulnerable to DoS attack and keystroke injection attack. The injection frequency f_{in} and voltage v_{in} were swept from 10 MHz to 98 MHz with a step of 4 MHz and 0.5 Vpp to 5 Vpp with a step of 0.5 Vpp, with a total of 230 injection experiments that take more than 30 minutes on each keyboard. Fig. 15 shows the results with a combined view of the frequency and amplitude dependency for successful EMI attacks. As we can see in this figure, certain frequencies outperform other frequencies (require smaller amplitude v_{in} to trigger GT), which validated our previous theory of attack in Section IV-A. Besides, we found that increasing the amplitude v_{in} allows a keyboard to be blocked at a frequency that is not blocked before, and these six keyboards are blocked at almost all frequencies when $v_{in} \geq 2.5$ Vpp.

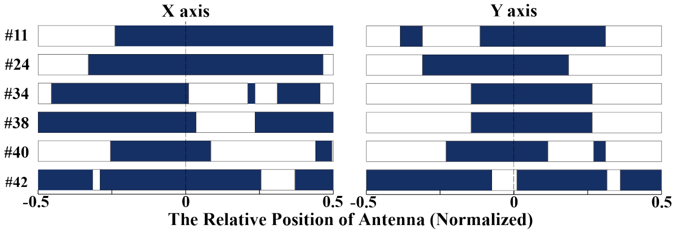


Fig. 17: `GhostType`'s robustness against antenna positions. The solid blue area represents successful injections relative to the keyboard's center position (normalized by keyboard length). Positive and negative values denote movement to the right/upward and left/downward.

2) Keystroke Injections: Similarly, we also selected six representative keyboards from Table V. Three keyboards (#6, #30, and #33) are as same as the DoS attack, and three (#38, #40, and #42) are vulnerable to only keystroke injections. We first swept the injection frequency f_{in} from 10 MHz to 20 MHz with a step of 1 MHz when v_{in}=1 Vpp. We captured the injection keystrokes over 3 minutes to calculate the average value of actions per minute (APM) to evaluate the injection speed. Fig. 16(a) shows that keystrokes can be injected at various frequencies, but the injection speed greatly varies with injection frequency, indicating certain frequencies outperform other frequencies. We then selected the frequency with the highest APM and swept the amplitude v_{in} from 0.5 Vpp to 1.5 Vpp with a step of 0.1 Vpp. Fig. 16(b) indicates that the injection speed of keystrokes increases as v_{in} increases.

D. Environmental Factors

To quantify the `GhostType` performance under the impact of antenna position, injection distance, table material and thickness, we conducted experiments using a similar setup as presented in Fig. 7(b), except there is a table top sample between the victim keyboard and antenna to evaluate the impact of different table materials and thicknesses.

1) Impact of Antenna Positions: Section IV-B1 points out that the relative antenna-to-trace position can affect the performance of `GT`. To evaluate the impact, we shifted the antenna horizontally and vertically along the keyboard plane. Specifically, we first put the antenna under the keyboard's geometric center and adjusted injection frequency f_{in} and amplitude v_{in} to conduct `GT` successfully. We then shifted the antenna in horizontal and vertical directions, respectively, without changing the injection signal. We recorded the positions where `GT` were still successfully conducted. We normalized the horizontal and vertical positions with respect to the keyboards' dimensions since the size of each keyboard is different. The results are shown in Fig. 17. We observed successful DoS and injection attacks in a wide range of horizontal and vertical directions. We also notice that the robustness to different injection positions is device-dependent because the traces on different keyboards are different, as shown in Appendix A. We also discovered that we could inject the same targeted keystroke when shifting the 14 mm and 12 mm away in the X and Y axes, respectively, without changing the injection parameters. In general, targeted keystroke injections are robust to the displacement of injection points within the keyboard region along the target RX trace as long as the signal does not interfere with other RX traces.

2) Impact of Injection Distance: We conducted the experiments with antenna-keyboard distances of 0 mm, 10 mm, 20 mm and 30 mm. As we can see in Table II, the success rate of DoS attack is up to 100% when the injection distance is 10 mm, and keystroke can still be injected when the injection distance is 30 mm. The injection performance reduces as distance increases because the electromagnetic signal intensity rapidly degrades with distance. In practice, a resourceful attacker can extend the attack distance by employing high-power amplifiers and directional antennas. We extended the attack distance for both DoS attack and keystroke injection to over 1 m using an Ettus LP0410 PCB directional antenna with a Mini-Circuits ZHL50W-63+ power amplifier. The demo of injecting keystrokes from a distance over 1 m is shown in [16]. Since the radiated scanning signal rapidly attenuates with distance, the adversary may only be able to synchronize with the keyboard via EM emission with a nearby attack device that is hidden underneath or inserted into the table. In comparison, the adversary can achieve a longer injection distance using a synchronization-free strategy if the keyboard employs a two-stage scanning mechanism or is a Bluetooth keyboard. For example, we demonstrated an example of a targeted keystroke injection at a distance of 1 m using the synchronization-free strategy in [16]. We believe that the achievable distance can be extended further by employing a professional antenna and amplifier with superior directionality and gain.

Both DoS attack and random keystroke injection can be performed at a distance of up to 1 m in this paper using a preliminary set of injection equipment, and the achievable distance can be extended further by employing a professional antenna and amplifier with superior directionality and gain.

3) Impact of Table Material: We selected four typical tabletop samples (solid wood, acrylic, medium density fiberboard (MDF), and glass) as the insulation material between the victim device and antenna and conducted injection experiments. The thickness of these table material samples is 10 mm. Table II shows that `GT` can achieve a similar success rate and injection speed performance. This suggests that contactless injection can be relatively robust to different table materials.

4) Impact of Table Thickness: We further evaluated the success rate and injection speed with different thicknesses of tables to understand the practicality of `GT`. We conduct the experiments when the thickness of the acrylic sheets is 10 mm, 15 mm, 20 mm, and 25 mm. Table II shows the success rate of DoS attack is up to 100% when the table thickness is 15 mm, and keystrokes can be injected into most keyboards when the table thickness is up to 25 mm. Note that the 25 mm effective attack distance is larger than the common table top thickness (less than 20 mm). The demo of injecting keystrokes under a 25 mm-thick table is shown in [16].

E. Hidden Keys

As mentioned in Section III-C, the hidden key phenomenon occurs due to the insecure implementation of the matrix circuit. To understand the full spectrum of these hidden keys, we investigated hidden keys on 10 keyboards from 6 vendors.

TABLE II: The Results of the Impact of Different Injection Distances, Table Materials, and Table Thickness.

Environmental Factors		**DoS Attack**			**Keystroke Injection**		
		#11	**#24**	**#34**	**#38**	**#40**	**#42**
Injection Distance	0 mm	100%	100%	100%	6804.0	23612.4	3962.6
	10 mm	100%	100%	100%	4448.2	19453.8	2403.0
	20 mm	42%	62%	100%	2951.8	10475.2	1635.6
	30 mm	0	0	100%	1017.8	2632.2	393.8
Table Material	Solid Wood	100%	100%	100%	3808.2	18464.8	2311.2
	Acrylic Sheet	100%	100%	100%	4248.4	16901.4	2336.0
	MDF	100%	100%	100%	4492.2	17310.4	2423.0
	Glass	100%	100%	100%	4035.8	14964.6	2375.4
Table Thickness	10 mm	100%	100%	100%	4248.4	16901.4	2336.0
	15 mm	100%	100%	100%	3570.0	11105.0	1828.6
	20 mm	100%	100%	100%	2509.2	7781.2	1064.6
	25 mm	70%	0	100%	1349.0	3579.8	617.2

Specifically, we ran a keystroke detection program on Windows and connected each TX and RX pin in pairs to reverse the key sets of the matrix circuit M_p. Then, we compared M_p with the keyboard's physical switches M_k and got hidden keys according to the analysis in Section III-C. The matrix circuits on different keyboards from the same vendor can be very similar (#34 vs. #35 from Rapoo) or different (#25 vs. #27 from Logitech). Matrix circuits on different keyboards from different vendors can be similar (#9 from Cherry vs. #34 from Rapoo). We discovered that hidden keys exist on every keyboard, and the specific number and type of hidden keys vary because of the different matrix circuits. These hidden keys could be divided into two main categories:

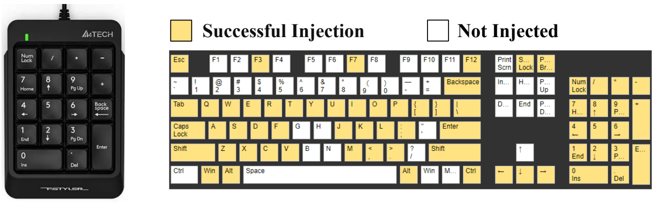
- *Function Keys:* We can inject hidden keys with different functionalities, such as opening the file browser/web browser/menu/mail app/music app/calculator, turning the media volume up/down, playing/pausing media, and putting the computer to sleep, etc.
- *Possible Debug Codes:* There are certain debug ASCII codes, including 110, 167, 170, 171, 193, 194, 235, 255, etc., and vendor-specific debug sequences such as "alt-8-2-1-0", "alt-insert-end-keydown-keyup", etc.

Potential Threat of Hidden Keys. Hidden keys are expected to cause security problems if the downstream OS system or software also neglects to identify and disable keystroke inputs that should not be accepted, e.g., alphabetical keystrokes from a numeric-only keypad. For example, a numeric-only keypad (#45) we examined has 44 hidden keys, including the alphabetical keys, function keys, and control keys (Fig. 18), which can be exploited to turn off a computer in our demonstration [16]. We hope our investigation can motivate stakeholders, particularly the manufacturers of critical devices, e.g., POS/ATM/public-facing terminals and kiosks, to examine their products against unexpected keystroke inputs.

F. Potential Attack Scenarios and Case Studies

We envision that GhostType works for both user-like adversaries that are granted use of the keyboards and adversaries that want to disrupt the operation of legitimate keyboard users.

For user-like adversaries, there are two potential scenarios: (1) GT can help them input keystrokes at a much higher speed than human inputs to outperform competitors in computer-based tests, gaming, etc. For example, a GT adversary in DOTA gaming may quickly inject the "QWER" hotkeys to dramatically increase their damages per second and gain advantages over others without being detected by software anti-cheating programs. (2) GT can help them inject hidden keys to trigger hidden functionalities of the downstream software systems and may cause unexpected consequences on system administrators. For example, we could inject hidden keys on a numeric-only keypad to *turn off the computer*.



Physical Layout

Keystrokes injected by GhostType

Fig. 18: GhostType injected 44 hidden keys on a numeric-only keypad, including alphabetical, function, and control keys.

For adversaries that want to disrupt the operation of legitimate keyboard users, we envision two main scenarios: (1) Manipulating keyboards on critical equipment such as medical devices, control terminals of industrial devices, and military weapons. Such critical keyboards are usually limited-accessible or even physically inaccessible since any untrusted or incorrect keystrokes can cause disastrous consequences. Despite the need for proximity access for limited-accessible keyboards, we envision that the adversary can install the disguised GT equipment near the target computer desk or keyboard tray via one-time access and then launch the attack remotely when other legitimate users or system administrators unlock and operate the computer. For physically inaccessible keyboards, the adversary may launch GT from a distance using a directional high-gain antenna. As a proof-of-concept, we demonstrate in [16] some examples of the adversary's capability in such scenarios, where we could inject certain targeted keystrokes to *close unsaved files*, *delete files*, *force the computer to shut down*, and inject random keystrokes *at a distance of up to 1 m* to disrupt and modify computer inputs. (2) Causing irreversible loss in time-sensitive scenarios. Some examples include a trader trading securities and stocks, a secretary typing to record key information of an important meeting, a gamer playing a world competition, etc. Our study shows adversaries can perform DoS attack to *completely block the keyboard* or inject random keystrokes to *keep the computer in hibernation*. Demos of case studies can be found on [16].

VI. Discussion

A. Countermeasures

To mitigate the vulnerabilities of keystroke sensing mechanisms, we provide insights into potential hardware and software mitigations gleaned from our investigations.

Shield Keyboards with Metal Materials. According to the findings in Section V-B, keyboards with a steel plate underneath the matrix circuit are less susceptible to EMI injections when the injection antenna is placed underneath the keyboard. It is worth noting that adversaries can still use the antenna above the keyboard to attack keyboards shielded with merely a metal plate underneath. We recommend that keyboard manufacturers employ metal enclosures as a straightforward countermeasure to protect both sides of the keyboard from EMI injections.

Enhance the Keystroke Sensing Mechanism. We believe keyboard manufacturers could improve the keystroke sensing mechanism in four ways. (1) Randomize the scanning signal waveform. The keyboard sensing mechanism can be spoofed primarily because the keyboard processor does not verify whether the received keystroke scanning signals came from the keyboard's TX. To ensure trustworthy keystroke sensing, we propose that the keyboard randomize the scanning signal waveform to be employed as the "verification signal". When a pressed key completes a circuit, the keyboard controller checks if that, and only that signal, is received on the appropriate RX pin. (2) Redesign the scanning signal's parameters. Our simulations in Appendix G revealed that decreasing the value of time difference ΔT between the two adjacent TXs considerably reduced the success rate of phantom keystroke injections. As a result, keyboard engineers can design appropriate scanning parameters to make the keyboards less vulnerable to GhostType. (3) Randomize the scanning sequence to make it difficult for adversaries to predict when and which TX is scanned to inject specific keystrokes into the targeted RX. (4) Detect and remove hidden keys using the proposed test method in Section V-E to avoid unexpected consequences.

B. Limitations and Future Work

Stealthy Attack Setups. As the first work investigating keyboards' analog sensing vulnerabilities, GhostType focuses on modeling and characterizing the attack limits and factors to lay the groundwork for further studies and keyboard engineering. Therefore, our proof-of-concept experiments used the existing laboratory equipment to explore the potential attack impact. Practical attacks in real-world scenarios can benefit from stealthy attack setups that can be concealed from victims. A future direction for the adversary is to employ professional (e.g., military-grade) amplifiers and directional antennas with superior directionality and gain to perform the attack at a long distance. The adversary may also build a small attack device that can be hidden underneath or inserted into the victim's table, as has been demonstrated in [21], [49].

Antenna Array. We proposed a methodology for injecting targeted keystrokes using a single antenna by localizing the injection into a specific RX at the timing of a specific TX. However, this method requires the adversary to maintain a roughly similar antenna position to the initial benchmarking setting, which may be difficult in some scenarios. A potential future work is to design an antenna array similar to [37], [49], [56], which can locate the victim device and adjust the antenna position for the targeted RX. We believe such an antenna array can also enable injecting a long sequence of keys, which is theoretically possible by combining single keystroke injection at different antenna positions.

High-security Keyboards. In this paper, we evaluated 50 off-the-shelf consumer-grade keyboards that are the most common types in everyday life. In addition to them, there are other types of keyboards, such as rugged or metal-plated keyboards used in high-security applications, including POS/ATM/public terminals, industry, medical, etc. These keyboards are generally protected by hardened cases with better shielding capabilities and robustness in tough environments. It will be worthwhile to investigate whether these keyboards are also susceptible to high-power EMI in future work.

VII. Related Work

Keyboard Security. The majority of previous studies on keyboard security focused on methods to eavesdrop on the typed keystrokes (also known as keylogging), which employs various information-derived compromising emanations to eavesdrop on keystrokes. Either the emanations from the keyboard [1], [3], [8], [19], [29], [31], [36], [55], or the user's physical state when typing [2], [6], [32], [35], [42], [45], [46], [51] are exploited to perform keylogging attacks. Others have tried injecting malicious fake keystrokes with reprogrammed USB devices masquerading as keyboard, such as BadUSB [23], Teensy [40], Mousejack [43] and Rubber Ducky [5]. Considering these threats, keyboards are recommended to be vetted or authenticated in security-sensitive applications to prevent these fake keystrokes from directly manipulating computers [7], [10], [18], [20], [26], [41], [53] In comparison to connecting a BadUSB device, this paper uncovers the vulnerabilities of the keyboard sensing mechanism and demonstrates the contactless injection of fake keystrokes into legitimate keyboards using EMI. To the best of our knowledge, this paper presents the first signal integrity analysis of the keystroke sensing mechanism.

EMI Attacks against Electronic Systems. Since analog signals are more susceptible to EMI than digital signals, there are various EMI attacks against analog sensors: microphones [30], [58], [59], implantable cardiac devices [30], magnetic speed sensors [50], smartphones [24], [58], light sensors [48], temperature sensors [34], [54], CCD and CMOS image sensors [21], [27], [60], touchscreens [37], [49], [56] and smart lock [38]. Besides, several EMI attacks against digital signals have been demonstrated, including bit-flip attacks on serial communications [12], [47], [48], PWM driving signals manipulations of AC-DC converters and servo motors [13]–[15], data falsifications targeting CAN frame [44] and vehicle charging communication [28]. The EMI attacks [37], [49], [56] against virtual keyboards on touchscreens are the closest to GhostType. In comparison, there are significant differences between GhostType and EMI attacks against touchscreens regarding the voltage level, signal waveform, and signal parameters (Appendix H). This paper uncovers the vulnerabilities of keyboard sensing mechanisms, which requires quantifying a unique set of theories and limits pertaining to keyboards' sensing mechanisms.

VIII. Conclusion

We present the first signal integrity analysis of keystroke sensing mechanisms and reveal a new class of vulnerabilities that could be exploited to manipulate keyboards and downstream computers. We develop the theory of contactless keystroke injections via EMI and design GhostType that can cause DoS of the keyboard and inject random keystrokes and certain targeted keystrokes. We provide the assessment and analysis of the vulnerabilities on 50 off-the-shelf keyboards and insights for potential hardware- and software-based countermeasures gleaned from our investigations.

Acknowledgments

We sincerely appreciate our anonymous reviewers and shepherd for their valuable comments and suggestions. This work was supported by China NSFC Grant 62201503, 62222114, 61925109, and 62071428.

References

[1] D. Asonov and R. Agrawal, "Keyboard Acoustic Emanations," in *Proceedings of the 2004 IEEE Symposium on Security and Privacy (IEEE SP 04)*.

[2] D. Balzarotti, M. Cova, and G. Vigna, "Clearshot: Eavesdropping on Keyboard Input From Video," in *Proceedings of the 2008 IEEE Symposium on Security and Privacy (SP)*.

[3] A. Barisani and D. Bianco, "Sniffing keystrokes with lasers/voltmeters," *Black Hat USA*, 2009.

[4] C. Bolton, Y. Long, J. Han, J. Hester, and K. Fu, "Characterizing and Mitigating Touchtone Eavesdropping in Smartphone Motion Sensors," in *Proceedings of the 26th International Symposium on Research in Attacks, Intrusions and Defenses*, 2023.

[5] B. Cannoles and A. Ghafarian, "Hacking experiment by using usb rubber ducky scripting," *Journal of Systemics*, 2017.

[6] B. Chen, V. Yenamandra, and K. Srinivasan, "Tracking Keystrokes Using Wireless Signals," in *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*, 2015.

[7] J. Choi, "Countermeasures for BadUSB Vulnerability," in *Proceedings of the Conference on Information Security and Cryptology*, 2015.

[8] A. Compagno, M. Conti, D. Lain, and G. Tsudik, "Don't Skype & Type! Acoustic Eavesdropping in Voice-Over-IP," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security (ACM ASIA CCS 17)*.

[9] Cortron, "Rugged keyboards for military industrial applications," https://www.cortroninc.com/rugged-keyboards-for-military-and-industrial-applications/, 2023, [Online; accessed 17-April-2023].

[10] P. Cronin, X. Gao, H. Wang, and C. Cotton, "Time-print: Authenticating usb flash drives with novel timing fingerprints," in *Proceedings of the 2022 IEEE Symposium on Security and Privacy (SP)*.

[11] Das Keyboard Blog, "Typing through time: Keyboard history," https://www.daskeyboard.com/blog/typing-through-time-the-history-of-the-keyboard/, 2011, [Online; accessed 1-Dec-2022].

[12] G. Y. Dayanıklı, A. Z. Mohammed, R. Gerdes, and M. Mina, "Wireless Manipulation of Serial Communication," in *Proceedings of the 2022 ACM Asia Conference on Computer and Communications Security (ACM ASIACCS 22)*.

[13] G. Y. Dayanikli, "Electromagnetic interference attacks on cyber-physical systems: Theory, demonstration, and defense," Ph.D. dissertation, Virginia Tech, 2021.

[14] G. Y. Dayanikli, R. R. Hatch, R. M. Gerdes, H. Wang, and R. Zane, "Electromagnetic Sensor and Actuator Attacks on Power Converters for Electric Vehicles," in *Proceedings of the 2020 IEEE Security and Privacy Workshops (SPW)*.

[15] G. Y. Dayanıklı, S. Sinha, D. Muniraj, R. M. Gerdes, M. Farhood, and M. Mina, "Physical-Layer Attacks Against Pulse Width Modulation-Controlled Actuators," in *Proceedings of the 31st USENIX Security Symposium (USENIX Security 22)*, 2022.

[16] GhostType, "Demos of ghosttype attacks," https://sites.google.com/view/ghosttype-demo, 2022, [Online; accessed 20-April-2023].

[17] Gloria Cascarino, "Medical equipment for outpatient care," https://www.hfmmagazine.com/articles/1349-medical-equipment-for-outpatient-care, 2014, [Online; accessed 17-April-2023].

[18] F. Griscioli, M. Pizzonia, and M. Sacchetti, "USBCheckIn: Preventing BadUSB Attacks by Forcing Human-device Interaction," in *Proceedings of the 2016 14th Annual Conference on Privacy, Security and Trust (PST)*. IEEE.

[19] T. Halevi and N. Saxena, "Keyboard acoustic side channel attacks: Exploring realistic and security-sensitive scenarios," *International Journal of Information Security*, 2015.

[20] G. Hernandez, F. Fowze, D. Tian, T. Yavuz, and K. R. Butler, "Firmusb: Vetting USB Device Firmware Using Domain Informed Symbolic Execution," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (ACM CCS 17)*.

[21] Q. Jiang, X. Ji, C. Yan, Z. Xie, H. Lou, and W. Xu, "GlitchHiker: Uncovering Vulnerabilities of Image Signal Transmission with IEMI," in *Proceedings of the 32st USENIX Security Symposium (USENIX Security 23)*, 2023.

[22] Y. Jiang, X. Ji, K. Wang, C. Yan, R. Mitev, A.-R. Sadeghi, and W. Xu, "WIGHT: Wired Ghost Touch Attack on Capacitive Touchscreens," in *Proceedings of the 2022 IEEE Symposium on Security and Privacy (SP)*.

[23] N. Karsten, K. Sascha, and L. Jakob, "Badusb-on accessories that turn evil," *Black Hat USA*, 2014.

[24] C. Kasmi and J. L. Esteves, "IEMI Threats for Information Security: Remote Command Injection on Modern Smartphones," *IEEE Transactions on Electromagnetic Compatibility*, vol. 57, no. 6, pp. 1752–1755, 2015.

[25] Keetouch, "What challenges does industrial equipment solve?" https://keetouch.eu/en/news/what-challenges-does-industrial-touch-equipment-solve.html, 2020, [Online; accessed 17-April-2023].

[26] A. Kharraz, B. L. Daley, G. Z. Baker, W. Robertson, and E. Kirda, "USBESAFE: An End-Point Solution to Protect Against USB-Based Attacks," in *Proceedings of the 22nd International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*, 2019.

[27] S. Kohler, R. Baker, and I. Martinovic, "Signal Injection Attacks against CCD Image Sensors," in *Proceedings of the 2022 ACM ASIA Conference on Computer and Communications Security (ACM ASIACCS 22)*.

[28] S. Köhler, R. Baker, M. Strohmeier, and I. Martinovic, "Brokenwire: Wireless disruption of ccs electric vehicle charging," *arXiv preprint arXiv:2202.02104*, 2022.

[29] M. G. Kuhn and R. J. Anderson, "Soft Tempest: Hidden Data Transmission Using Electromagnetic Rmanations," in *Proceedings of the International Workshop on Information Hiding*. Springer, 1998.

[30] D. F. Kune, J. Backes, S. S. Clark, D. Kramer, M. Reynolds, K. Fu, Y. Kim, and W. Xu, "Ghost talk: Mitigating emi signal injection attacks against analog sensors," in *Proceedings of the 2013 IEEE Symposium on Security and Privacy (SP)*.

[31] J. Liu, Y. Wang, G. Kar, Y. Chen, J. Yang, and M. Gruteser, "Snooping Keystrokes with mm-level Audio Ranging on a Single Phone," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, 2015.

[32] X. Liu, Z. Zhou, W. Diao, Z. Li, and K. Zhang, "When Good Becomes Evil: Keystroke Inference with Smartwatch," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (ACM CCS 15)*, 2015.

[33] Y. Long, P. Naghavi, B. Kojusner, K. Butler, S. Rampazzi, and K. Fu, "Side Eye: Characterizing the Limits of POV Acoustic Eavesdropping from Smartphone Cameras with Rolling Shutters and Movable Lenses," in *Proceedings of the 2023 IEEE Symposium on Security and Privacy (SP)*.

[34] Y. Long, S. Rampazzi, T. Sugawara, and K. Fu, "Protecting covid-19 vaccine transportation and storage from analog cybersecurity threats," *Biomedical Instrumentation & Technology*, vol. 55, no. 3, pp. 112–117, 2021.

[35] A. Maiti, O. Armbruster, M. Jadliwala, and J. He, "Smartwatch-based Keystroke Inference Attacks and Context-aware Protection Mechanisms," in *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security (ACM ASIA CCS 16)*, 2016.

[36] P. Marquardt, A. Verma, H. Carter, and P. Traynor, "(sp)iphone: Decoding vibrations from nearby keyboards using mobile phone accelerometers," in *Proceedings of the 18th ACM conference on Computer and Communications Security (ACM CCS 11)*, 2011.

[37] S. Maruyama, S. Wakabayashi, and T. Mori, "Tap'n ghost: A Compilation of Novel Attack Techniques against Smartphone Touchscreens," in *Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP)*.

[38] A. Z. Mohammed, A. Singh, G. Y. Dayanıklı, R. Gerdes, M. Mina, and M. Li, “Towards Wireless Spiking of Smart Locks,” in *Proceedings of 2022 IEEE Security and Privacy Workshops (SPW)*.

[39] J. V. Monaco, “SoK: Keylogging Side Channels,” in *Proceedings of the 2018 IEEE Symposium on Security and Privacy (SP)*.

[40] S. M. Nasution, Y. Purwanto, A. Virgono, and G. C. Alam, “Integration of Kleptoware as Keyboard Keylogger for Input Recorder Using Teensy USB Development Board,” in *Proceedings of the 8th International Conference on Telecommunication Systems Services and Applications (TSSA)*. IEEE, 2014.

[41] S. Neuner, A. G. Voyiatzis, S. Fotopoulos, C. Mulliner, and E. R. Weippl, “Usblock: Blocking USB-based Keypress Injection Attacks,” in *Proceedings of the IFIP Annual Conference on Data and Applications Security and Privacy*. Springer, 2018.

[42] A. Neupane, M. Rahman, N. Saxena *et al.*, “Peep: Passively Eavesdropping Private Input via Brainwave Signals,” in *Proceedings of the International Conference on Financial Cryptography and Data Security*. Springer, 2017.

[43] M. Newlin, “Mousejack, keysniffer and beyond: Keystroke sniffing and injection vulnerabilities in 2.4 ghz wireless mice and keyboards,” *DEFCON*, 2016.

[44] H. Ogura, R. Isshiki, K. Iokibe, Y. Kodera, T. Kusaka, and Y. Nogami, “Electrical Falsification of CAN Data by Magnetic Coupling,” in *Proceedings of the 2020 35th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC)*.

[45] G. Oligeri, S. Sciancalepore, S. Raponi, and R. Di Pietro, “Brokenstrokes: On the (in) security of wireless keyboards,” in *Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2020.

[46] M. Sabra, A. Maiti, and M. Jadliwala, “Zoom on the Keystrokes: Exploiting Video Calls for Keystroke Inference Attacks,” in *Proceedings of the Network and Distributed System Security (NDSS) Symposium*, 2021.

[47] J. Selvaraj, “Intentional electromagnetic interference attack on sensors and actuators,” Ph.D. dissertation, Iowa State University, 2018.

[48] J. Selvaraj, G. Y. Dayanıklı, N. P. Gaunkar, D. Ware, R. M. Gerdes, and M. Mina, “Electromagnetic Induction Attacks against Embedded Systems,” in *Proceedings of the 2018 ACM Asia Conference on Computer and Communications Security (ACM ASIACCS 18)*. ACM.

[49] H. Shan, B. Zhang, Z. Zhan, D. Sullivan, S. Wang, and Y. Jin, “Invisible Finger: Practical Electromagnetic Interference Attack on Touchscreen-based Electronic Devices,” in *Proceedings of the 2022 IEEE Symposium on Security and Privacy (SP)*.

[50] Y. Shoukry, P. Martin, P. Tabuada, and M. Srivastava, “Non-invasive Spoofing Attacks for Anti-lock Braking Systems,” in *Proceedings of the International Conference on Cryptographic Hardware and Embedded Systems*. Springer, 2013.

[51] D. X. Song, D. Wagner, and X. Tian, “Timing Analysis of Keystrokes and Timing Attacks on {SSH},” in *Proceedings of the 10th USENIX Security Symposium (USENIX Security 01)*, 2001.

[52] The Fintech Times, “Three ways to get the most out of your atm: a spotlight on lac,” https://thefintechtimes.com/3-ways-to-get-the-most-out-of-your-atm-a-spotlight-on-lac/, 2023, [Online; accessed 17-April-2023].

[53] D. J. Tian, A. Bates, and K. Butler, “Defending against Malicious USB Firmware with GoodUSB,” in *Proceedings of the 31st Annual Computer Security Applications Conference*, 2015.

[54] Y. Tu, S. Rampazzi, B. Hao, A. Rodriguez, K. Fu, and X. Hei, “Trick or Heat? Manipulating Critical Temperature-Based Control Systems Using Rectification Attacks,” in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (ACM CCS 19)*.

[55] M. Vuagnoux and S. Pasini, “Compromising Electromagnetic Emanations of Wired and Wireless Keyboards,” in *Proceedings of the 18th USENIX Security Symposium (USENIX Security 09)*, 2009.

[56] K. Wang, M. Richard, C. Yan, X. Ji, S. Ahmad-Reza, and W. Xu, “GhostTouch: Targeted attacks on touchscreens without physical touch,” in *Proceedings of the 31st USENIX Security Symposium (USENIX Security 22)*.

[57] D. A. Ware, “Effects of intentional electromagnetic interference on analog to digital converter measurements of sensor outputs and general purpose input output pins,” Ph.D. dissertation, Utah State University, 2017.

[58] Z. Xu, R. Hua, J. Juang, S. Xia, J. Fan, and C. Hwang, “Inaudible attack on smart speakers with intentional electromagnetic interference,” *IEEE Transactions on Microwave Theory and Techniques*, vol. 69, no. 5, pp. 2642–2650, 2021.

[59] C. Yan, H. Shin, C. Bolton, W. Xu, Y. Kim, and K. Fu, “Sok: A Minimalist Approach to Formalizing Analog Sensor Security,” in *Proceedings of the 2020 IEEE Symposium on Security and Privacy (SP)*.

[60] H. Zhang, Q. Jiang, Y. Cheng, X. Ji, and W. Xu, “Intentional electromagnetic interference attack against infrared thermal imaging sensor,” in *Proceedings of the 2022 IEEE 6th Conference on Energy Internet and Energy System Integration (EI2)*.

Appendix

A. The Irregular TX and RX Traces on Different Keyboards

Examples of the traces on the lower and upper sheet on six keyboards. We found that these TX and RX traces are irregular and vary from keyboard vendor and model. For example, as shown in Fig. 19, the traces on two Logitech keyboards (Logitech MK 220 and Logitech MK 235) are quite different. These traces differ amongst keyboards, resulting in different vulnerabilities to `GhostType` and making it difficult to inject long sequences of targeted keystrokes.

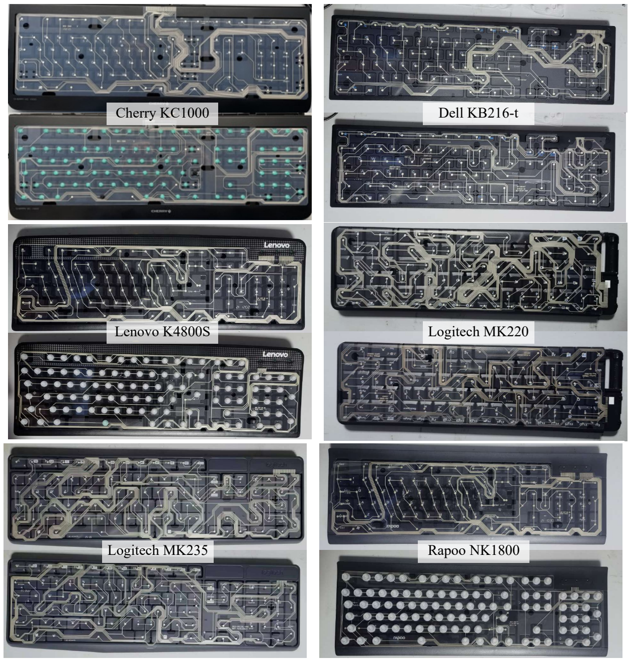


Fig. 19: Illustrations of the TX and RX traces on matrix circuits of 6 keyboards. We can find that these TX and RX traces are irregular and vary from keyboard vendor and model.

B. Simulations in Ansys HFSS.

We create a proof-of-concept model in Fig. 20, where a loop antenna is utilized for EMI injection attacks, and two microstrips are used to emulate the traces. To fully describe a trace’s geometry, three parameters are used: length L_t, width W_t, and angle θ_t. The antenna plane is parallel to the microstrip plane, the vertical distance between the two planes

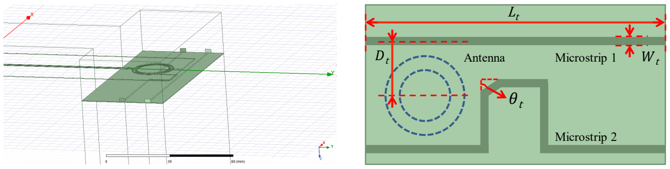


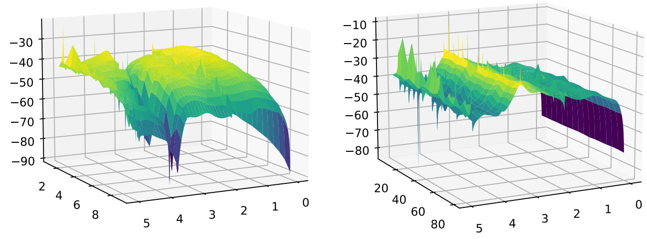
Fig. 20: Illustrations of the 3D perspective and parameter definitions for the proof-of-concept simulation model in Ansys HFSS.

is d_t, and the horizontal distance between the microstrip 1 and the antenna center position is D_t. The relative antenna-to-trace position can be expressed as $\sqrt{D_t^2 + d_t^2}$. Since a quantitative evaluation of these four factors is out of the scope of this paper, we design four sets of simulations to investigate the impact of each factor on a trace's susceptibility to EMI and determine a dominant factor to design our attack strategy. Specifically, the frequency of the injection signal is swept from 10 MHz to 5 GHz at a step of 10 MHz, and the value of four factors are set as indicated in Table III.

TABLE III: Value of simulation parameters.

No.	Parameters	Value of Start, End and Step	Simulation Points
1	f_{in}	10 MHz, 5 GHz, 10 MHz	500
2	L_t	1 mm, 9 mm, 0.5 mm	17
3	W_t	1 mm, 9 mm, 0.5 mm	17
4	θ_t	5°, 85°, 5°	17
5	D_t	1 mm, 9 mm, 0.5 mm	17

The simulation results are illustrated in Fig. 21. We discovered that the strength of the coupled signal decreased when increasing the distance of the relative antenna-to-trace position in Fig. 21(a) and the effect of angle on the coupled signal's amplitude is not significant in Fig. 21(b). These results reveal that the relative antenna-to-trace position plays a dominant role in a trace's susceptibility to EMI.



(a)

(b)

Fig. 21: Results of the induced signal's amplitude on the microstrip when (a) sweeping the signal frequencies with different relative antenna-to-probe positions, and (b) sweeping the signal frequencies with different angles.

C. Designing of the Twenty-One Injection Positions

We created a 4*8 grid in Fig. 22, with a total of 21 intersection positions $P_1, P_2, \cdots, P_h, \cdots, P_{21}$. We performed keystroke injection attacks at each position for 30 s with the same injection amplitude V_{emi} and a different frequency f_{in} that is swept from 10 MHz to 50 MHz in 10 MHz steps, for a total of 105 sets of tests. The result of the injected keystrokes' RX distributions is illustrated in Fig. 13.

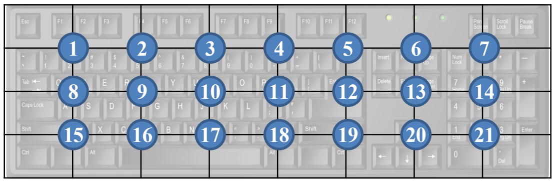


Fig. 22: Illustration of the injection positions.

D. The Feasibility of Synchronizing with Scanning

The strength of the radiated scanning signals changes between TXs because of the varying TX-trace-to-antenna positions. Since the traces on the matrix circuits are the same for two keyboards from the same manufacturer and batch, when placing the receiving antenna in the same position underneath these two keyboards, the strongest emission from keyboard scanning corresponds to the same TX in theory. To demonstrate this, we conducted an experiment on two Cherry KC1000 keyboards from the same batch. One keyboard served as a benchmark, while the other served as the victim. We placed the receiving antenna underneath the key "G" on the benchmark keyboard and found that the strongest emission came from the 15-th TX trace. We then used this emission as a synchronizing signal to inject the targeted keystrokes "3", "E", and "D" at positions P3, P6, and P14 in Fig. 22, respectively. We repeated the experiment 10 times at each position, and the success rate was calculated when the target keystroke was successfully injected. Next, we placed the receiving antenna underneath the key "G" on the victim keyboard and repeated the same experiment at positions P3, P6, and P14. The results are shown in Table IV. The same key was injected with a similar success rate, indicating that the strongest EM emission from keyboard scanning mostly coincides with the same key on both the victim and the benchmark keyboard despite subtle variations between individual keyboards.

TABLE IV: Results of GhostType on the Benchmark and Victim Keyboard.

#	Antenna Position	Benchmark Keyboard		Victim Keyboard	
		Key	Succ.	Key	Succ.
1	P3	3	8/10	3	6/10
2	P6	E	9/10	E	8/10
3	P14	D	7/10	D	7/10

E. Keyboard's Two-Stage Scanning Mechanism

The keyboard employs a two-stage scanning mechanism to detect key presses in the interrupt-column mode and identify the pressed key in the polling mode. The flowchart of the scanning mechanism is illustrated in Fig. 23. The keyboard processor is configured in the *interrupt-driven mode* by default at the first state to detect a key press and switch to the second state as soon as a key press is detected. In the *polling mode*, the processor scans a key for the first time and starts the timer for debounce delay; when the debounce delay elapses, the timer

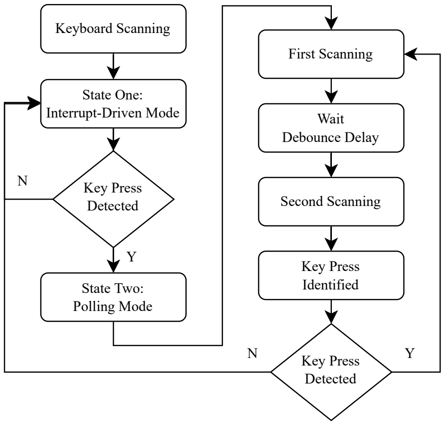


Fig. 23: Illustration of the flowchart of the keyboard's two-stage scanning mechanism. The keyboard employs a two-stage scanning mechanism to detect key presses in the interrupt-column mode and identify the pressed key in the polling mode.

triggers an interrupt for the processor to read the logic state of RX pins again to identify the pressed key. The keyboard processor remains in the polling mode to identify each pressed key until no keys are pressed, at which point it returns to the first scanning mode. This two-stage scanning mechanism conserves energy because the computational overhead of the first state is significantly lower than that of the second state.

Fig. 24 illustrates an example of this kind of two-stage scanning mechanism on a Cherry KC1000 keyboard. The scanning process of detecting a key press can be divided into three segments. The scanning state switching between two scanning modes is visible at the beginning of the second segment. Based on this two-stage scanning mechanism, we can develop a synchronization-free strategy to inject a targeted keystroke without sensing the emanations for synchronization. We can inject a trigger signal to trigger the scanning restart and then adjust the delay time to determine the injection timing to inject a targeted key without synchronizing the injection with the scanning signal.

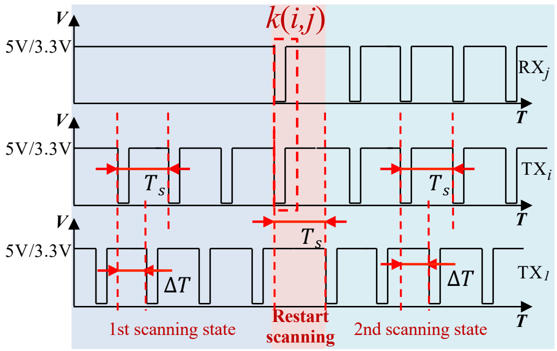


Fig. 24: Illustrations of the keyboard's two-stage scanning mechanism, where detecting a key press triggers the scanning state transition. This two-stage scanning mechanism provides opportunities for designing the synchronization-free injection strategy to inject a targeted keystroke without synchronizing the injection with scanning.

F. The Steel Plate Under the Matrix Circuit

We disassembled the two keyboards that are susceptible to GhostType attack during our evaluation in Section V-B. Their inners were shown in Fig. 25, and we found a steel plate under the matrix circuit. We envision the metal plate acting as additional EMI shielding to protect these two keyboards from our alias attacks. We propose this insight as a hardware-based countermeasure for keyboard manufacturers.

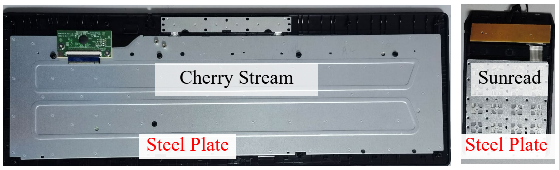


Fig. 25: Illustrations of the steel plate underneath the matrix circuit, which shields the matrix circuit from GhostType attack.

G. Redesigning the Parameters of the Scanning Signal

We found in our evaluation in Section V-B that although keyboards employ a similar scanning mechanism, the specific value of scanning parameters varies from vendor and model (Table I), resulting in different sensitivity to GhostType. Inspired by this, we investigated the keyboards that are relatively less vulnerable to our GhostType attacks and found that the keyboard with a lower value of time difference ΔT between the two adjacent TXs was relatively less vulnerable to our GhostType attacks. We envision this is because decreasing the value of ΔT reduces the probability that the injection signal dropped an RX below V_{IL} when a TX is scanned in at least two consecutive scanning cycles, thus reducing the success rate of the attack. We validated this through simulations by only changing the value of ΔT. The other scanning signal parameters do not change. The results of the simulation in Fig. 26 showed that the success rate of phantom keystroke injections was significantly reduced when the value of time difference ΔT between the two adjacent TXs decreased, validating our hypothesis. When ΔT =40 us, the injection success rate is only 19.5%. Thus, keyboard manufacturers can carefully choose their scanning parameters to make the keyboards less vulnerable.

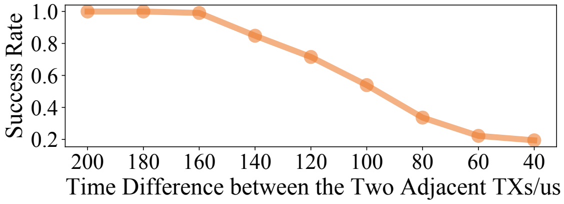


Fig. 26: Simulation results indicate that the success rate of phantom keystroke injections was significantly reduced when the value of time difference ΔT between the two adjacent TXs decreased.

TABLE V: Overall Performance of GhostType on 50 Off-the-shelf Keyboards.

#	Specificaiton			DoS		Keystroke Injection			#	Specification			DoS		Keystroke Injection		
	Vendor and Model	Structure	Protocol	Fre.	Succ.	Fre.	Succ.	Targeted Keys		Vendor and Model	Structure	Protocol	Fre.	Succ.	Fre.	Succ.	Targeted Keys
1	A4TECH MK100	Mem.	USB	76-78	30/30	33-37.7	30/30	14	26	Logitech MK275	Mem.	BLE	39.2-98.2	30/30	19.1-37.6	30/30	4
2	A4TECH KBN-8510	Mem.	USB	68.2-80.1	30/30	42-48	30/30	3	27	Logitech MK220	Mem.	BLE	37.7-43.3 49.4-100	30/30	17.1-37.6 43.4-49.3	30/30	4
3	A4TECH FG1010	Mem.	BLE	88.3-100	30/30	17.3-88.2	28/30	8	28	Logitech G610	Mech.	USB	✗	-	96.9-97.8	30/30	3
4	A4TECH KB-N9100	Mem.	USB	75.2-86.3	30/30	23.5-58.7	30/30	2	29	Microsoft 850	Mem.	BLE	36.3-52.2 75.4-100	30/30	52.3-75.3	30/30	3
5	ACER YKB913	Mem.	USB	100	30/30	44.8, 82.5 93.3	30/30	5	30	Microsoft 900	Mem.	BLE	32.4-46.5 82.6-100	30/30	32.4 82.6	30/30	1
6	ACER KM41-2K	Mem.	BLE	17.8-19.5 31.5-33.7	30/30	15.1-17.7 27.5-31.4	30/30	5	31	Philips SPK6234	Mem.	USB	98.8	30/30	✗	-	-
7	BOW MK610	Mem.	BLE	80-90	30/30	10-80 90-100	30/30	6	32	Philips SPT6103	Mem.	BLE	29.5-57 63-100	30/30	17.5-29.5 57-63.2	30/30	3
8	BOW HW098A	Mem.	USB	10-100	30/30	✗	-	-	33	Philips SPK6212B	Mem.	USB	79.1-100	29/30	10-79	30/30	4
9	Cherry KC1000	Mem.	USB	17.8-23.1 83.3-100	30/30	23.2-31.3 42.1-83.2	30/30	10	34	Rapoo K150	Mem.	USB	66.1-100	30/30	✗	-	-
10	Cherry Stream	Mem.	USB	✗	-	✗	-	-	35	Rapoo X125S	Mem.	USB	74.1-100	30/30	✗	-	-
11	Dell KB216-t	Mem.	USB	23.9-27.7 37.7-44.2 73.3-100	30/30	✗	-	-	36	Rapoo 8050T	Mem.	BLE	19.3-20	30/30	20.1-100	30/30	1
12	Dell KM17	Mem.	BLE	15-96	30/30	96.1	30/30	1	37	Razer RZ03-0146	Mem.	USB	✗	-	10-100	30/30	3
13	Dell KM2123D	Mem.	BLE	10-26.5 27-74.8	30/30	26.6 75-100	30/30	2	38	Razer RZ03-0147	Mem.	USB	✗	-	10-100	30/30	3
14	Dell KB3022D	Mech.	USB	✗	-	93.3-100	29/30	1	39	Thunderobot KG3089R	Mech.	USB	✗	-	62-89	30/30	4
15	Dell KB522P	Mem.	USB	24.0-24.5 92.7-97.6	30/30	✗	-	-	40	Thunderobot KG3104R	Mech.	USB	✗	-	38-39.8 91-100	30/30	2
16	HP GK400F	Mech.	USB	✗	-	43.6-100	30/30	5	41	Thunderobot KM400	Mem.	BLE	87-100	30/30	✗	-	-
17	HP KM10	Mem.	USB	10-27.6	28/30	27.7-85.9	30/30	3	42	Xiaomi HZJP01YM	Mech.	USB	✗	-	10-100	30/30	5
18	HP CS10	Mem.	BLE	28.7-57.1	30/30	18.3-28.6 57.2-99	30/30	3	43	Xiaomi WXJS01YM	Mem.	BLE	42-85	29/30	10-40 86-100	30/30	3
19	IKBC W200	Mech.	BLE	✗	-	86-100	30/30	3	44	Xiaomi JXJP01MW	Mech.	USB	✗	-	33.3-33.7	30/30	1
20	Keycool K-9	Mech.	USB	✗	-	37.2-62.3 74.9-96.1	30/30	5	45	A4TECH FK13P	Mem.	USB	33.4 41.9	30/30	22-33 34-38.4	30/30	2
21	Lenovo K4800S	Mem.	USB	93-100	30/30	38.9-44.8 76.6-90	30/30	2	46	CoolSpeed	Mem.	USB	30.1-42.8 68.1-100	30/30	19.6-30.0 42.9-68.0	30/30	2
22	Lenovo MK23	Mem.	BLE	33.6-42 77.6-100	30/30	17.3-33.5 42-77.5	30/30	2	47	Hiz	Mem.	USB	87.9-90	30/30	29.6-43.3	30/30	3
23	Lenovo K104	Mech.	USB	✗	-	35-45 70.5-87.9 95-100	30/30	2	48	IBM	Mem.	USB	38.1-42.2 74-77 83.1-100	30/30	26.9-38 42.3-45.5 77.1-83	30/30	3
24	Lenovo EKB-536A	Mem.	USB	38.7-45.2 88.0-96.8	30/30	✗	-	-	49	Rapoo K10	Mem.	USB	27.6-45.7 67.3-100	30/30	✗	-	-
25	Logitech MK235	Mem.	BLE	73-100	30/30	18-48	30/30	3	50	Sunread SKB886S	Mem.	USB	✗	-	✗	-	-

H. Comparison with EMI injection attacks on Touchscreens

There are significant differences between GhostType and EMI injection attacks against virtual keyboards on touchscreens [37], [49], [56]:

- **Voltage Level:** Keyboards detect keystrokes by reading the on/off status of a 3.3 V/5 V circuit via GPIOs, which is more difficult to interfere with EM signals than touchscreens that use millivolt-level voltages.
- **Signal Waveform:** GhostType injects AC signals that are sampled as DC signals to spoof GPIO readings which requires the design of AC signals that can be interpreted as DC signals by the victim circuits, whereas touchscreen attacks often directly inject AC signals in a similar waveform as the encoded touchscreen scanning signals.
- **Signal Parameter:** Compared with touchscreens' regular TX-RX matrices, keyboard circuits have irregular forms that vary with the model. Thus, GhostType must consider complex parameter combinations such as signal frequency, voltage, timing, and position, whereas touchscreen attacks can use the same signal parameters and only change the antenna position.

I. Overall Performance on 50 Off-the-shelf Keyboards

The experiment results of GhostType on 50 off-the-shelf-keyboards are shown in Table V, indicating that 48 of these 50 keyboards are susceptible to GT attacks.