DualStrike: Accurate, Real-time Eavesdropping and Injection of Keystrokes on Commodity Keyboards

Xiaomeng Chen[†], Jike Wang[†], Zhenyu Chen[†], Qi Alfred Chen[‡], Xinbing Wang[†], Dongyao Chen[†✉]
[†]Shanghai Jiao Tong University, China
[‡]University of California, Irvine, USA
{sjtu_chenxm, jikewang, zhenyu.chen, xwang8, chendy}@sjtu.edu.cn, alfchen@uci.edu

***Abstract*—We discover that enabling both eavesdropping and non-invasive, per-key injection is viable on keyboards, in particular, the fast-emerging commodity Hall-effect keyboards. This paper introduces `DualStrike`, a new attack system that allows attackers to remotely listen to victim input and control any key on a Hall-effect keyboard. This capability opens doors to severe attacks (e.g., file deletion, private key theft, and tampering) based on the victim's input and context, all without requiring hardware or software modifications to the victim's computer. We present several key innovations in `DualStrike`, including a novel, compact electromagnet-based hardware design for high-frequency magnetic spoofing, a synchronization-free attack scheme, and a magnetometer-based listening mechanism using commercial off-the-shelf components. Our real-world experiments demonstrate that `DualStrike` can reliably compromise arbitrary keys across six recent Hall-effect keyboard models. Specifically, `DualStrike` achieves over 98.9% keystroke injection accuracy across all tested models. In an end-to-end test, the eavesdropping module achieves a high listening accuracy (i.e., above 99%). To improve the robustness of `DualStrike`, we implement a calibration algorithm to account for keyboard displacement, allowing it to maintain 98.5% injection accuracy even with offsets up to 4 cm. We also identified `DualStrike`'s immunity to existing magnetic shielding mechanisms and proposed a novel shielding approach for Hall-effect keyboards. The demo video of `DualStrike` can be found in [30].**

I. Introduction

Keyboards have been essential input devices for computers. Therefore, the integrity of the keyboard is of paramount importance. Most studies have focused on keystroke inference attacks, which can only eavesdrop on typed keystrokes to gather sensitive information but cannot compromise the victim's inputs. In contrast, keystroke injection attacks aim to actively input malicious commands. However, most efforts involve invasive methods, such as reprogrammed USB devices masquerading as keyboards [39], which are now well-defended at the software level. Recently, GhostType [41] demonstrated the feasibility and challenges of keystroke injection on mechanical and membrane keyboards by injecting electromagnetic interference (EMI) into their scanning circuits. However, existing injection attacks are not aware of the victim's inputs. As a result, these attacks either blindly inject messages into the keyboard or require the attacker's physical presence.

[✉]Dongyao Chen is the corresponding author.

Network and Distributed System Security (NDSS) Symposium 2026
23–27 February 2026, San Diego, CA, USA
ISBN 979-8-9919276-8-0
https://dx.doi.org/10.14722/ndss.2026.230046
www.ndss-symposium.org

Therefore, we argue that a practical attack on keyboards should integrate both keystroke inference and injection.

Non-invasive, per-key injection. The attacker should have precise control over most keys on the victim's keyboard without any physical contact or internal modifications. This allows injection of malicious contents *as if typed by the victim*. Such capability enables a wide range of harmful actions, such as ultra-fast file deletion, remote malware installation, and document tampering.

Real-time eavesdropping. The attacker can leverage eavesdropping to enhance and guide injection attacks. For example, after a "sudo" keystroke, the inputs between two "ENTER" keystrokes are likely to be the user's password. With this information and injection capability, the attacker can control the victim's computer, thus leading to severe consequences. As we will elaborate in Sec. II, the eavesdropping capability can also help infer the context information (e.g., user's liveness and keyboard placement), thus boosting the injection accuracy.

In this paper, we discover that the fast-emerging, commodity Hall-effect keyboard is susceptible to *both* keystroke eavesdropping and injection. Specifically, we present `DualStrike` attack — attackers can remotely access the victim's keyboard input and conduct corresponding malicious activities. To the best of our knowledge, this presents the first time that keyboard devices are demonstrated vulnerable to both eavesdropping and injection. As we will elaborate in Sec. III, achieving this can introduce new, severe, and practical threats to users.

The Hall-effect keyboard, also known as the magnetic switch keyboard, differs from traditional keyboards mainly in its switch-sensing mechanism. Traditional keyboards require a key to be fully pressed to register a keystroke. In contrast, the Hall-effect keyboard can continuously detect keystroke motion using a passive magnet and an underlying Hall sensor in each switch. Specifically, when the user presses the magnetic keycap, the key will be triggered as soon as the Hall sensor detects that the magnetic signal has reached the preset threshold. This provides unique advantages, including much shorter response times, adjustable key travel, and longer lifespan. As we will

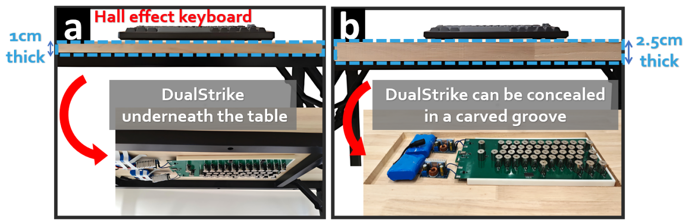

a
Hall effect keyboard
1cm thick
DualStrike underneath the table
b
2.5cm thick
DualStrike can be concealed in a carved groove

Fig. 1: Real-world deployment of `DualStrike`. Note that the complete setup of `DualStrike` can be concealed underneath a desk or embedded within a carved groove under the desk.

elaborate in Sec. II-A, within the last year, the number of new Hall-effect keyboard models has shown exponential growth — the average growth rate reaches *82.60%* every three months.

In the design of `DualStrike`, we first analyze the sensing mechanisms of Hall-effect keyboards via reverse engineering (Sec. IV). The observation helps us identify the root cause of Hall-effect keyboards' security issues, which reside in the keystroke sensing and scanning mechanisms. Ironically, we find that the most distinctive feature of these keyboards, the Hall switches, can be the *new root cause* for severe keystroke eavesdropping and injection attacks. When a keystroke occurs, the resulting magnetic field changes can serve as side-channel information, enabling attackers to monitor keyboard activity. Similarly, these magnetic variations can be disrupted or mimicked by external magnetic fields, facilitating keystroke injection. Furthermore, attackers can exploit the fast responsiveness of Hall sensors to significantly enhance injection speed.

The design of `DualStrike` is comprised of an injection module, an eavesdropping module, and a calibration module. Specifically, we propose the following key design elements:

Electromagnet-based magnetic injection. First, we use an electromagnet to spoof the magnetic field strength. To minimize the form factor of the electromagnet while maintaining sufficient magnetic field strength for spoofing, we conduct simulations using finite element analysis. Our findings show that suction electromagnets outperform solenoid ones in concentrating the magnetic field. Next, we design our attack parameters based on the scanning parameters of the Hall-effect keyboard, which can be obtained via a one-time reverse engineering effort for each keyboard model. By controlling the start voltage and the activation/deactivation timing of the electromagnet, we can accurately inject up to 12,553 keystrokes per minute. Our empirical study indicates that `DualStrike`'s injection module can adapt to various Hall-effect keyboard configurations (e.g., key travel distances, scanning circuits, parameters, and sizes).

Magnetometer-based keystroke eavesdropping. Despite the compact size of keycap-integrated magnets, we find that common commercial off-the-shelf (COTS) magnetometers (e.g., MLX90393) can detect magnetic field fluctuations at a typical desktop thickness distance. As we will elaborate in Sec. V-A, we place MLX90393 sensors within the gaps of the electromagnet array. By integrating our hardware configuration with real-time magnetic field analytics pipeline, `DualStrike`'s eavesdropping module can accurately identify the key that is being pressed with high accuracy (i.e., 99.54%).

Calibration of keyboard displacement. As shown in existing works, compromising keyboards accurately have stringent requirements for keyboard placement. For example, GhostType requires the keyboard displacement to be less than *14 mm*. As we will elaborate in Sec. V-C, powered by the eavesdropping module, `DualStrike` can infer the precise displacement, thus realigning the electromagnets. Our empirical results (Sec. VI) indicate `DualStrike` can sustain the injection performance with displacement as large as 4 cm.

As shown in Fig. 1, our attack device can be concealed under a desk during actual use and does not require any additional equipment (e.g., power supply, power amplifier). This ensures the independence and stealthiness of `DualStrike`. Once a sensitive context (e.g., a password) is obtained through eavesdropping, `DualStrike` can autonomously trigger the injection sequence, keeping the attacker fully hidden from the victim. We also discuss how existing magnetic shielding schemes *cannot* defend against `DualStrike`. We then propose an efficient, low-cost hardware-based countermeasure against our attacks.

In `DualStrike`, we make the following contributions:

- We proposed `DualStrike`, a new attack that can perform both keyboard listening and non-invasive, per-key keystroke injection on Hall-effect keyboards, while also incorporating a calibration mechanism to mitigate real-world disruptions such as keyboard displacement.
- We performed the first reverse-engineering effort to help design practical attacks on Hall-effect keyboards;
- We find that existing magnetic shielding schemes *cannot* effectively defend against `DualStrike` attack. We further provide new approaches as defense methods.

II. Background

A. The Fast-emerging Hall-effect Keyboards

Due to their high sensitivity, durability, and reliability, Hall-effect switches have long been used in safety-critical industries such as aerospace [13], underwater devices [24], and the military [23], for sensing magnetic field strength. In 1968, Honeywell integrated Hall-effect switches into keyboard circuits, introducing the world's first Hall-effect keyboard [1], [10]. However, mass production was impeded due to the bulky size and high cost of Hall-effect switches at that time.

Recently, following successful launches by major keyboard OEMs, e.g., SteelSeries and Wooting, and Logitech [3], more keyboard manufacturers are fast advancing to Hall-effect technology. As shown in our survey results (Fig. 2), since August 2023, the Hall keyboard market has experienced exponential growth, with an average growth rate of 82.60% *every three months*. This rapid brand expansion has driven down prices (around $14 [4]), making prices of new models comparable to mechanical counterparts. As a result, Hall-effect keyboards are not only popular among performance enthusiasts but are also increasingly favored by everyday users.

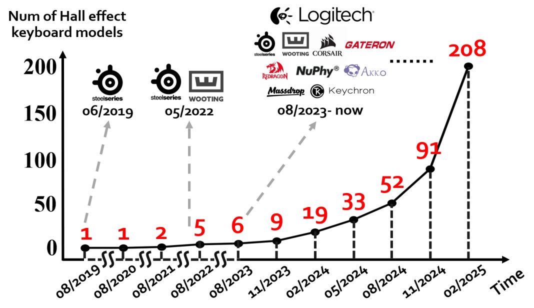


Fig. 2: The rapid adoption of Hall-effect keyboards.

B. Premier on Hall-effect Keyboards

A Hall-effect keyboard consists of three main components: Hall-effect switches, a scanning circuit, and a communication port. When a key is pressed, the switch detects the press, which is then transmitted to the keyboard circuit and processed by the scanning mechanism. The detected input is sent to the computing device via a wired (e.g., USB) or wireless (e.g., Bluetooth) communication port.

Compared to traditional keyboards, the core difference in Hall-effect keyboard lies in the keystroke sensing mechanism. In a Hall-effect keyboard, each switch contains a permanent magnet and a Hall sensor. As depicted in Fig. 3, the permanent magnet is positioned within the stem connected to a keycap and moves up and down as the keycap is pressed and released. The Hall sensor is placed beneath the switch on the keyboard's PCB. Utilizing the Hall effect, the sensor measures the magnetic field strength induced by the permanent magnet, which is perpendicular to the sensor's surface, and outputs a voltage proportional to that magnetic field strength. In contrast, traditional keyboards employ switches that must be fully pressed until switches contact with a conductive base to register a keystroke. The contactless-sensing feature of Hall-effect switches offers three unique advantages:

1) **Customizable key travel.** The key travel refers to the distance a switch must be pressed to register a keystroke. The high sensitivity of the Hall sensor allows customization of the key travel distance. For example, the SteelSeries Apex Pro [11] Hall-effect keyboard allows key travel adjustment between 0.1 mm and 4 mm with a granularity of 0.1 mm.
2) **Quicker response time.** By setting a shorter key travel, the time from press to actuation is reduced, thus enabling faster response time. This is crucial in high-performance keyboard applications, such as esports, where players need to quickly press specific keys for actions.
3) **Longer lifespan.** The contactless design of Hall-effect switches significantly extends their lifespan, with SteelSeries keyboards rated for up to 100 million key-presses, twice that of mechanical switches [18].

We argue that, despite these advantages, the magnetic switch is also the root cause of Hall-effect keyboards for eavesdropping and injection attacks. Specifically, the Hall-effect switch cannot differentiate the source of the magnetic field.

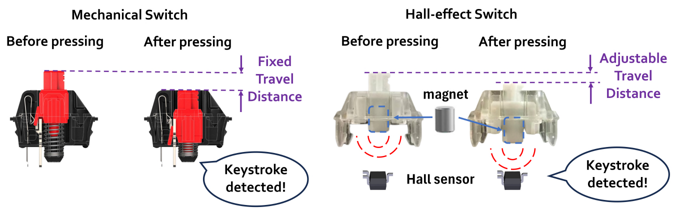


Fig. 3: Comparison between the mechanical keyboard switches and Hall-effect keyboard switches.

The external magnetic field may interfere with the magnetic field incurred by a keystroke, thus providing potential avenues for injection attacks. Moreover, the movement of magnets can lead to distinctive patterns for magnetic field sensors, e.g., low-cost, off-the-shelf MEMS magnetometers.

C. Existing Attacks on Keyboards

Existing attacks on keyboards are mostly keystroke inference side-channel attacks that do not directly affect the victim's computer [58], [56], [43], [60], [46]. To alter the keystroke information, existing methods rely on implanting reprogrammed USB devices [48]. However, this invasive approach is well-studied and can be prevented via software detection [39], [37]. Recently, GhostType [41] demonstrated the feasibility of contactless keystroke injection attacks, focusing on membrane and mechanical keyboards. GhostType exploits the scanning process of keyboards to inject keystrokes via electromagnetic interference (EMI). Specifically, GhostType requires (i) synchronization with the keyboard's scanning matrix. (ii) use of a modulated EMI signal to carry out the attack based on pre-known scanning circuit parameters. Although this achieves non-invasive keystroke injection, GhostType fell short in eavesdropping the keystroke, unable to achieve per-key injection, and its performance can be severely undermined by subtle displacement of the keyboard.

III. Attack Model and Challenges

A. Attack Model

Attackers aim to both eavesdrop on the victim's input and manipulate the victim's typing inputs. We consider a common scenario where a Hall-effect keyboard on a desk is connected to a computer, such as in public spaces like libraries or cafes. The attacker positions the `DualStrike` attack device under the desk. To further conceal the device, the attacker can carve a rectangular groove in the desk to fit the `DualStrike` device. Note that this change can be practical as replacing desk surface requires minimum cost and is common in the remodeling process. We illustrate this setup in Fig. 1. Similar settings have been explored in prior works [44], [59]. We assume the attacker knows the Hall-effect keyboard's model. By purchasing a similar keyboard, the attacker can study the characteristics of its sensing circuit. Configurations such as the desk's thickness and material can also be easily measured.

With its eavesdropping module, `DualStrike` can capture sensitive information (e.g., passwords) during active typing and infer user liveness to decide whether to proceed with

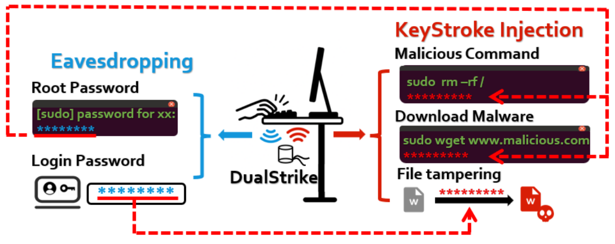


Fig. 4: Attack scenarios of `DualStrike`. Note that sensitive information obtained via eavesdropping can be exploited in subsequent, more critical injection attacks, e.g., root password.

the injection attack, thus ensuring collision-free execution and enhancing the attack's practicality. A lightweight, on-device detection algorithm can enable `DualStrike` to locally locate and record such critical information. For example, after detecting a long period of inactivity, the initial keystrokes are likely login credentials; on Linux systems, keystrokes entered after a "sudo" command are likely to be administrative passwords.

With the sensitive context information, the attacker can implement several practical attack vectors as shown in Fig. 4. (1) Leveraging the captured root password, `DualStrike` can inject root-level *malicious commands*, such as deleting the root directory or logging out the current user. According to our evaluation, `DualStrike` can inject up to 40 characters within 200 ms — faster than the human reaction time [40]. This enables the execution of short malicious commands before the victim can respond. (2) `DualStrike` can covertly download malware from a remote attacker-controlled server. (3) With `DualStrike`'s accuracy and coverage of keystrokes, attackers can *modify sensitive files*, e.g., high-impact public documents, computer programs, and machine learning models, via key injections. Compared with existing file tampering schemes [38], [50], [57], [47], `DualStrike` does not require any modification of the targeted computer's hardware/software.

B. Design Challenges

Designing the attack device. Existing works of magnetic injection [36] commonly rely on large electromagnets to improve the magnetic field strength. For example, the height and diameter of the electromagnet in existing work [36] are 3.8 cm and 7 cm, respectively. However, in Hall-effect keyboards, each switch corresponds to a Hall sensor with a typical spacing of 19.05 mm in ANSI standard [15]. This confined space makes it challenging to achieve per-key keystroke injection. Therefore, we need to carefully design the layout of electromagnets, thus (1) producing sufficient magnetic field strength and (2) minimizing interference with adjacent switches to reduce false triggers.

High-frequency magnetic injection. Previous works that inject magnetic spoofing signals have only achieved low-frequency injections, e.g., 2 Hz in [36]. Improving the injection speed can make the attack more stealthy, thus more practical in the real world. As we will elaborate in Sec. V-B2, the inductive

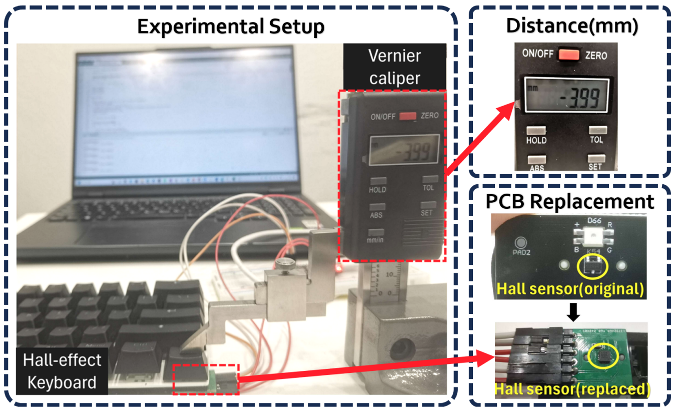


Fig. 5: Analysis of Hall-effect switch vulnerabilities.

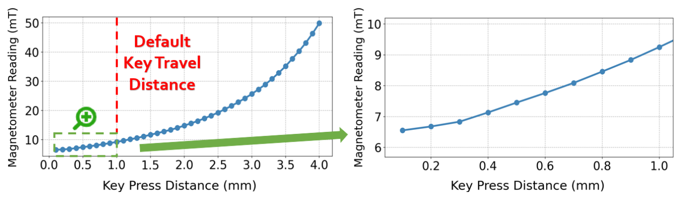


Fig. 6: Magnetic field strength under varying press distances.

characteristics of electromagnets cause a delay in current rise, which limits the ability to increase the injection frequency.

Alignment of attack device. In normal keyboard usage, users may adjust the keyboard's position based on personal habits. Therefore, successfully executing keystroke injection attacks despite misalignment between the attack device and the targeted keyboard poses a significant challenge.

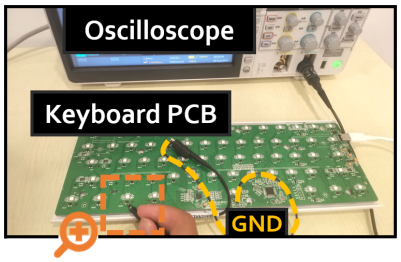
IV. Characterizing Keyboards

A. Hall-effect Switches

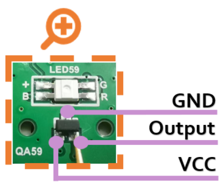
We ask: *How much magnetic field strength is needed to trigger a keystroke?* and *can magnetometers sense the movement of magnetic switches?* After disassembling Hall-effect keyboards, we found it challenging to answer due to the use of *proprietary Hall sensors*, which lack accessible specifications. Although we can measure sensor's output voltage, we can't directly determine the corresponding magnetic field strength.

To overcome this obstacle, we developed a lightweight reverse engineering method. Specifically, we used a commercial off-the-shelf Hall sensor MLX90393 [21] to quantify the magnetic field. As shown in Fig. 5, we replaced the original PCB of the Hall-effect keyboard with a PCB that contains the MLX90393 sensor. We used the Wooting 60 HE keyboard, which senses key travel distances from 0.1 to 4 mm, and placed our PCB under the "Fn" key to collect data. By simulating key presses with a vernier caliper (Nscing Es 889-101), which allows 0.01 mm increments, we recorded magnetic field data at various key travel distances.

Fig. 6 shows the correlation between magnetometer readings and key press distance, indicating the magnetic field strength needed to trigger a keystroke. At the maximum key travel distance of 4 mm, the sensor reading reaches 50 mT. As the

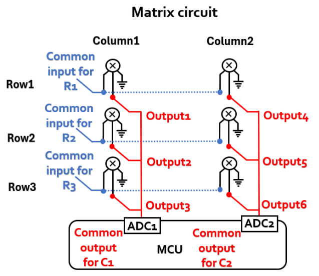


(a)

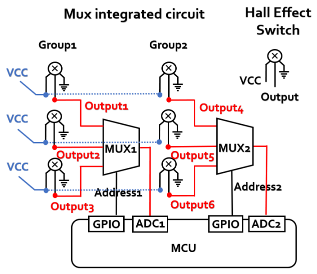


(b)

Fig. 7: (a) The setup for reverse-engineering Hall-effect keyboards' scanning circuits. (b) A common three-pin Hall sensor is used in the Hall-effect keyboard.



(a) Matrix Circuit



(b) MUX-Integrated Circuit

Fig. 8: Two types of scanning circuit.

travel distance decreases, the magnetic field strength drops rapidly due to the inverse cube law [54]. This experiment reveals: (1) when the key travel distance is set below the default value, i.e., 1 mm, only 9 mT is needed to trigger a keystroke. (2) commodity magnetometers can capture the variance of the magnetic field of a keystroke behavior.

Note that this sub-1mm key travel distance is often recommended by keyboard manufacturers [29] because shorter key travel allows for faster response times, which increases the keyboard's sensitivity.

B. Scanning Modules

Compared with traditional keyboards, Hall-effect keyboards also include a new scanning mechanism — essential for supporting the faster response time.

To investigate the scanning circuit of Hall-effect keyboards, we disassembled six popular off-the-shelf Hall-effect keyboards and reverse-engineered their scanning circuits. As shown in Fig. 7, the linear Hall-effect sensor used in these keyboards typically has three pins: VCC, GND, and the output voltage pin. We used an oscilloscope to monitor the signals on the VCC and output pins of the Hall sensor. By analyzing these signals, we identified two scanning circuits in Hall-effect keyboards as shown in Fig. 8 (a) and (b). One is the matrix circuit, and the other incorporates multiplexers (MUXs):

Matrix circuit. In this design, all keys are organized into rows and columns. Each row is connected to a shared power circuit trace, and each column is linked to a shared output circuit trace. In the scanning process, power is applied to each row at a time. If a key in the currently powered row is pressed, the voltage on the corresponding column changes, allowing the circuit to detect the pressed key.

MUX-integrated circuit. Instead of the complex matrix traces across the keyboard, we observed that some Hall-effect keyboard models use MUX-integrated scanning circuits. This type of circuit uses multiplexiers to divide keys into groups and queries the address to detect voltage outputs.

TABLE I: Scanning circuit types and cycle periods of different Hall-effect keyboards.

Keyboard Type	Scanning Type	Cycle
Wooting 60 HE	MUX-integrated	375 μs
Steelseries Pro	Matrix	990 μs
Keydous NJ98-CP	MUX-integrated	5160 μs
k70 Pro Max	Matrix	900 μs
Reddragon M61	Matrix	1000 μs
DrunkDeer A75 Pro	Matrix	880 μs

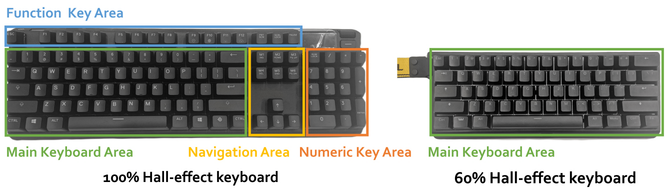


Fig. 9: Different layouts of Hall-effect keyboards.

We measured the scan cycle period—the time required to scan all keys— and circuit types for different keyboards, as shown in Table. I. Note that Hall-effect keyboards have shorter scan cycles compared to traditional keyboards (2.4 ms to 8.2 ms [41]). These differences make the existing EMI-based injection infeasible for Hall-effect keyboards. In Sec. V-B, we will leverage these findings in Table. I for designing `DualStrike`'s injection engine.

C. Keyboard Layouts

Hall-effect keyboard manufacturers introduce different keyboard sizes. A full-sized keyboard includes four key areas: the main keyboard area, function key area, navigation key area, and numeric key area. 80%, 75%, 65%, and 60%-sized keyboards (as shown in Fig. 9) crop certain areas of the 100% keyboard. Despite the diversity in keyboard sizes, we found that all keyboards include at least the main typing area. Moreover, the layout of the main typing area is highly consistent: they follow the standard "QWERTY" [26] layout and comply with the ANSI keyboard spacing standard.[1]

V. Design of DualStrike

Fig. 10 illustrates the overall system design of `DualStrike`, which consists of three key components: (1) Keystroke eavesdropping. `DualStrike` employs a magnetometer array to continuously monitor the victim's keystrokes to collect the desired information. (2) Keystroke injection. Through novel designs in four subsystems—attack

[1]The former ensures a consistent key arrangement, while the latter ensures that the key spacing adheres to a standard (i.e., typically a 19.05 mm interval).

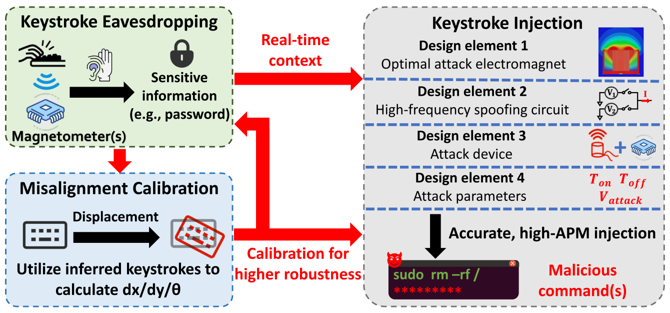


Fig. 10: The system overview of `DualStrike`.

unit, circuit, device, and parameters—`DualStrike` enables accurate, high-APM keystroke injection. (3) Misalignment calibration. To address possible keyboard displacement during victim usage, `DualStrike` uses eavesdropped data to infer the keyboard's posture relative to the attack device.

A. *Magnetometer-based Eavesdropping*

As described in Sec. II-B, due to the lack of shielding in current Hall-effect keyboards, this creates a potential vulnerability where the leaked magnetic field information can be used to infer the victim's keystrokes.

To exploit this vulnerability, we use the COTS magnetometer MLX90393 to measure the magnetic field strength. We divide the main keyboard area into a 4×2 grid and place an MLX90393 magnetometer near the center of each grid cell, forming a 4×2 array, as shown in Fig. 14. Based on our empirical study (Sec. VI), this layout is sufficient for the attack device to detect all keystrokes in the main keyboard area at a standard desk thickness.

In the continuous sensor data stream, the critical information resides in the readings captured during a victim's keystroke. Our goal is to mitigate the effects of surrounding magnetic fields (e.g., geomagnetic fields) and sensor noise, accurately detect the timing of keystrokes, and isolate the magnetic field changes caused by the keystrokes. Using the magnetometer data, we finally predict specific keystrokes. Our eavesdropping process is divided into three steps: preprocessing, peak detection, and keystroke classification, as shown in Fig. 11.

Step 1. Preprocessing. For each sensor and its axes, we first select non-triggered data recorded at the beginning of the operation as the offset. This offset is subtracted from the raw data to eliminate environmental noise. To further smooth the data, we apply a Savitzky-Golay (SG) filter with a window size of 15 and a polynomial order of 3.

Step 2. Peak detection. Using the preprocessed data from three axes, we calculate the overall magnetic field strength of each sensor: $B_{overall} = \sqrt{\text{mag}_x^2 + \text{mag}_y^2 + \text{mag}_z^2}$. To mitigate remaining noise, a sliding window with a size of 5 is applied to $B_{overall}$. For each window, we extract the local maximum: $\max V_m(t') \quad (t' \in [t, t + \Delta t])$, resulting in the final envelope curve. Next, we apply the peak detection algorithm [55] to identify peaks that correspond to keystroke events.

Step 3. Keystroke classification. When any of the eight

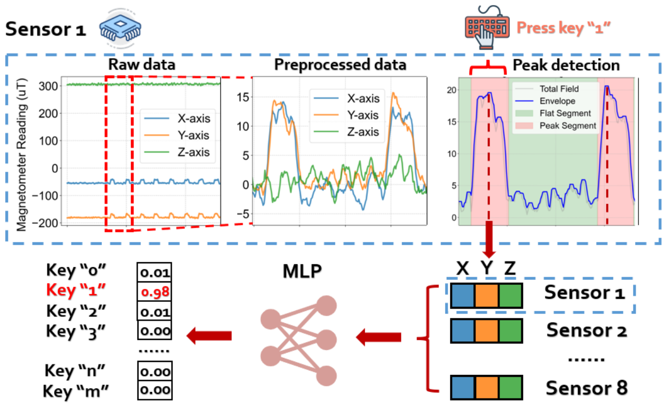


Fig. 11: Illustration of the eavesdropping algorithm.

sensors detects a peak, we record the timestamp corresponding to the *highest envelope value*. At this time, the preprocessed three-axis data from all eight sensors are used as input for prediction. We employ a lightweight MLP model with a single hidden layer of 32 nodes to predict all key categories. As shown in the evaluation (Sec. VI.2), our classification algorithm obtains high accuracy (i.e., above 99%).

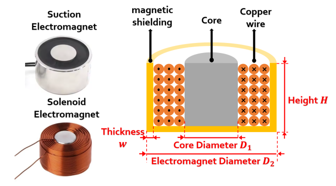
The eavesdrop algorithms can be adapted for real-time operation. With a 250 Hz sampling rate and a window size of 15, the algorithm delay is only $(15 - 1)/2 \times 1/250 \approx 0.028$ seconds, enabling real-time eavesdropping on victim's input.

B. *Electromagnet-based Injection Design*

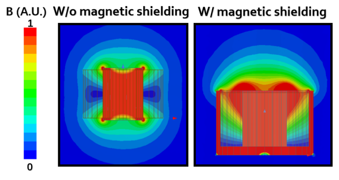
1) Electromagnet Design: To achieve controllable magnetic signal spoofing, we select electromagnets as the attack units. The design follows three principles: generating sufficient magnetic field strength, stealthiness, and cost efficiency.

We begin by evaluating two electromagnet designs: solenoid and suction electromagnets, as depicted in Fig. 12(a). Both types share a copper coil and a soft-iron core. The suction electromagnet includes an additional outer shell of soft iron, designed to enhance the magnetic field by concentrating it at the top. To compare the magnetic field distribution of these two types, we conduct simulations using Ansys (Maxwell 2022 R1), a widely used finite element analysis (FEA) tool [16]. As shown in Fig. 12(b), the suction electromagnet produces a more concentrated magnetic field on its top surface due to the shell, making it the preferred choice for `DualStrike`. By factory-bonding its non-mechanical components into a rigid structure and leveraging upward magnetic flux to suppress crosstalk between adjacent units, the suction electromagnet operates without vibration during magnetic spoofing, enabling fully static and silent attacks.

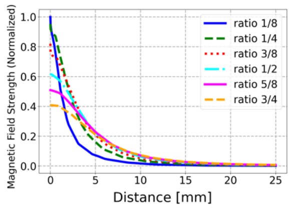
Our next step is optimizing the suction electromagnet configuration in `DualStrike`. Specifically, we focus on four key parameters: shell thickness (W), height (H), core diameter (D_1), and overall diameter (D_2). Using Ansys, we simulate the electromagnet's behavior with the same experimental settings, e.g., the same voltage applied. We compare the magnetic field strength at 25 mm above the top surface.



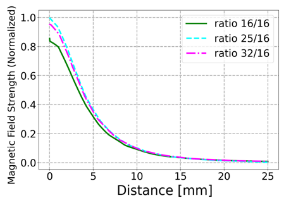
(a) Electromagnet and cross-sectional view



(b) Impact of magnetic shielding



(c) Simulation for different core ratios.



(d) Simulation for diff. configurations

Fig. 12: Electromagnets design and simulation results.

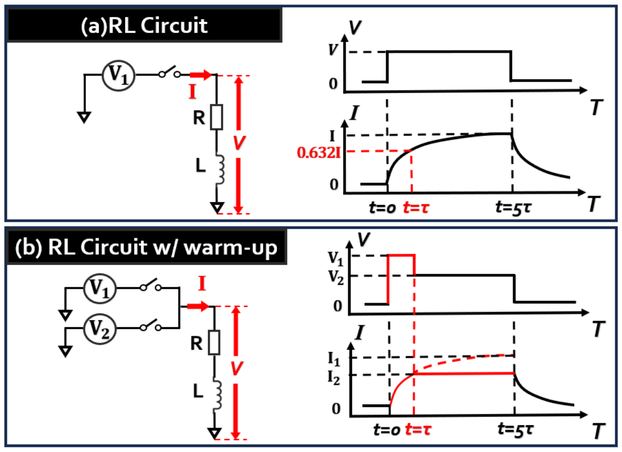


Fig. 13: (a) illustrates the characteristics of an RL circuit, and (b) depicts the RL circuit with a warm-up phase.

Diameter design. An electromagnet usually has a high permeability core, e.g., steel, to enhance the magnetic field strength. We vary the core-to-diameter ratio D_1/D_2 from 1/8 to 3/4, as shown in Fig. 12(c). Based on the magnetic field distribution at various distances, we select a core ratio of 1/2, which offers strong magnetic field strength across different distances while minimizing the core diameter to reduce costs. Considering the spacing constraints of keyboard (i.e., 19.05 mm) and to avoid collisions, we determine D_2 as 16 mm and D_1 as 8 mm.

Height design. To determine the height, we customize three electromagnets with different heights: 16 mm, 25 mm, and 32 mm. Fig. 12(d) shows that the 25 mm height produces the strongest field, so we finalize this height for our design.

2) *High-frequency Magnetic Spoofing:* We now analyze the electromagnet's circuit characteristics. As shown in Fig. 13(a), the electromagnet can be modeled as an RL circuit, with a voltage source and switch controlling the on/off state. The current in the RL circuit increases with a delay, which also delays the magnetic field strength as it is proportional to the current (Ampere's law [14]). Specifically, the transient response of the current is given by:

$$I = \frac{V}{R}\left(1 - e^{-t/\tau}\right), V = V_0 * u(t) \tag{1}$$

In Eq. 1, L and R represent inductance and resistance. The unit step function $u(t)$ indicates the voltage is applied at $t = 0$. The time constant $\tau = \frac{L}{R}$ determines how fast the circuit responds to input changes. After τ, the current reaches 63.2% of its maximum value. And typically, after 5τ[12], it is considered to have reached its full value $\frac{V_0}{R}$. As elaborated in Sec. IV-B, Hall-effect keyboards typically have a scanning period of around 1 ms. However, the transient effects of the RL circuit can hinder fast switching, potentially causing significant delays in the attack.

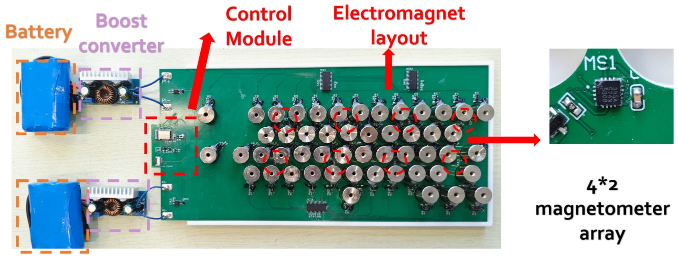


Fig. 14: The attack device of `DualStrike`.

To address this issue, we propose a high-frequency switching circuit with a novel warm-up strategy. Specifically, we focus on two key characteristics of the RL circuit: (i) τ is determined by the circuit parameters and is independent of the applied voltage; (ii) the fastest current rise occurs during the initial τ period. Based on these two characteristics, we discover that by applying a higher voltage during this phase, we can quickly bring the circuit to the desired current level. This *warm-up phase* can effectively reduce the circuit's startup time. As shown in Fig. 13(b), the applied voltage can be described as follows:

$$V = V_1 * (u(t) - u(t - \tau)) + V_2 * u(t - \tau) \tag{2}$$

To implement this warm-up phase, we use two voltage sources. As shown in Fig. 13(b), if the desired current is $\frac{V_2}{R}$, we apply a higher voltage V_1 during the first τ to rapidly increase the current. When $\frac{V_2}{V_1} = 0.632$, the current reaches the desired level after τ. At this point, V_1 is switched off and V_2 is switched on, maintaining the current. This reduces the rise time from 5τ to τ, enabling a $5\times$ switching frequency while maintaining the required magnetic field strength.

3) *The Attack Device:* `DualStrike`'s attack device is shown in Fig. 14. It comprises five key components: a universal electromagnet layout capable of attacking different size Hall-effect keyboards; a switching module for electromagnet selection; a magnetometer array for eavesdropping;

TABLE II: Keys covered in our design.

Type	Details
Letters	abcdefghijklmnopqrstuvwxyz
Symbol Keys	,./;'-
Control Keys	Shift,Ctrl,alt,OS,Esc,Backspace,Enter,CapsLk
Numeric Keys	1234567890
Whitespace	' '

a control module that manages switching and executes local computation; and a power supply. The entire attack device measures 35 cm in length, 16 cm in width, and 2.6 cm in height, with a total weight of approximately 750 g. Compared to existing work [41] that requires bulky external equipment such as power supplies and power amplifiers, `DualStrike` occupies minimal space. The attacker can further downsize the attack device by customizing the target keys based on actual usage. For example, omitting unused numeric keys can reduce both size and weight by 20%. Targeting specific malicious commands can also mitigate the overhead of the hardware. For example, only executing `sudo rm -rf /` requires only 20% of the full hardware footprint. This flexibility enables targeted injection and eavesdropping, further enhancing `DualStrike`'s stealthiness.

Universal electromagnet layout: We place our electromagnets based on Sec. IV-C, aligning their arrangement and spacing with the corresponding keys in the main keyboard area. This design ensures compatibility across various Hall-effect keyboard models. As validated in Sec. VI, the layout is effective across multiple keyboards. We target 51 keys listed in Table II, covering representative input types such as text, URLs, and bash commands. Note that an attacker can further reduce the size by selecting only essential keys. For instance, targeting a subset of left-side function keys occupies minimal space, while still enabling impactful injections and facilitating more stealthy deployment.

Switching module: Each electromagnet is controlled by a switch unit using a MOSFET (AO3442) for switching and a diode (DSK16) for surge protection. To manage all switches, we use three 16-bit GPIO expanders (MCP23017), each provide 16 additional I^2C-controlled ports. This allows a standard MCU to control all keys, and enables simultaneous key activation for combinations like `OS + X`.

Magnetometer array: We use eight MLX90393 magnetometers for eavesdropping, each configured at 250 Hz and read via the SPI protocol. With our lightweight algorithm, keystroke inference is performed on-device. In future designs, if the attacker aims to eavesdrop over longer distances, the MLX90393 can be replaced with magnetometers that offer higher resolution and lower noise levels, such as the RM3100.

Control module and power supply: All computing tasks of `DualStrike` are performed *locally*. Specifically, we use an MDBT42Q microcontroller to control electromagnet switching, and to execute eavesdropping and calibration algorithms. A 3D-printed case holds the PCB and electromagnets to ensure uniform height for reliable injection. For power, two

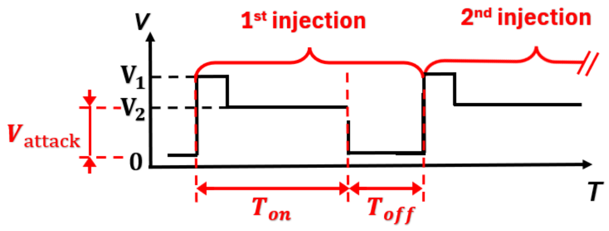


Fig. 15: The illustration of attack parameter designs.

12V batteries are used, each connected to a boost converter (XL6012) that outputs between 12V and 65V. The attacker presets the voltage, ensuring a ratio of 0.632 between two voltages to enable high-frequency switching. Thus, we can support an attack voltage of up to 40 V. An attacker can extend the attack range by using a higher-voltage boost converter. For instance, replacing the XL6012 with a same-size LT8361[32] (up to 100V) supplies a higher voltage and thus a stronger field.

4) Attack Parameters: For accurate injection attacks, based on the analysis of the keyboard scanning circuit in Sec. IV-B, the magnetic field must exceed the threshold when the target key is scanned. However, to prevent accidental keystrokes, manufacturers usually use a debounce mechanism[2], requiring the keystroke to be detected in two consecutive scan cycles. Existing methods usually require synchronizing with the scan cycle to identify the start of each cycle and the phase for the targeted key. However, the fast scan rates of Hall-effect keyboards make it difficult and cumbersome to achieve synchronization over two consecutive cycles. To solve this, we propose a sync-free keystroke injection method.

As shown in Fig. 15, each injection consists of two phases: voltage activation (T_{on}) and voltage deactivation (T_{off}). The voltage activation includes the warm-up phase mentioned in Sec. V-B2, followed by the sustained voltage phase, which we define as V_{attack} (equal to V_2). By keeping the sustained voltage active long enough to cover two complete scan cycles, we enable sync-free keystroke injection.

To achieve accurate keystroke injection, we analyze three key parameters. Specifically, a keystroke results in one of four possible outcomes: (i) *match*: "A" $\rightarrow$ "A"; (ii) *overtrigger*: "A" $\rightarrow$ "AA"; (iii) *missing*: "A" $\rightarrow$ "" (empty); (iv) *mismatch*: "A" $\rightarrow$ "B". Only match-type outcomes are considered accurate.

$\mathbf{T_{on}}$: As discussed earlier, the sustained voltage must last for at least two scan cycles. Considering the warm-up phase, the minimum duration is $T_{\text{min}} = \tau + 2 \times T_{\text{cycle}}$. Note that there is also an upper limit for T_{on} to avoid overtrigger. We found that this limit is set by the operating system rather than the keyboard. For example, Windows has a "keyboardDelay" parameter with four options (0-3), corresponding to delays of 300 ms, 550 ms, 800 ms, and 1050 ms. We also investigated MacOS and Ubuntu, finding that the smallest over-trigger threshold is 300 ms. Therefore, $T_{\text{max}} = 300$ ms, and T_{on} must satisfy $T_{\text{min}} < T_{\text{on}} < T_{\text{max}}$ for accurate injection.

$\mathbf{V_{attack}}$: V_{attack} depends on environmental factors (e.g., desk thickness, material) and the keyboard type. To ensure sufficient magnetic strength, the attacker can first reverse-engineer the

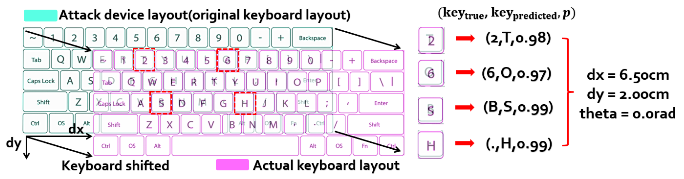


Fig. 16: The illustration of displacement calibration.

setup to determine the minimum required voltage V_{min}. Accurate injection requires $V_{\text{attack}} > V_{\text{min}}$; otherwise, keypresses may fail. Notably, key height can vary across rows (e.g., 10–16 mm on Wooting keyboards). As shown in Sec. VI, a higher V_{attack} can adapt to all keys on the keyboard while maintaining high performance without causing interference.

$\mathbf{T_{off}}$: The above settings ensure accurate single-key injection. For two consecutive keystrokes, T_{off} must be considered. When keys differ, T_{off} can be zero. However, for consecutive identical keystrokes, setting $T_{\text{off}} = 0$ may cause them to be interpreted as a single long press rather than two distinct inputs. In this case, T_{off} should equal T_{cycle}, allowing the first keypress to end before the next begins.

The conditions for accurate injection parameters are:

$$\begin{cases} T_{\text{min}} < T_{\text{on}} < T_{\text{max}},\ V_{\text{attack}} > V_{\text{min}},\ T_{\text{off}} = 0 & \text{Key}_{\text{now}} \neq \text{Key}_{\text{next}} \\ T_{\text{min}} < T_{\text{on}} < T_{\text{max}},\ V_{\text{attack}} > V_{\text{min}},\ T_{\text{off}} = T_{\text{cycle}} & \text{Key}_{\text{now}} = \text{Key}_{\text{next}} \end{cases} \quad (3)$$

To achieve high injection speed, we set T_{on} to T_{min}. As shown in Sec.VI-C1, our sync-free method reduces overhead due to the fast scan cycles of Hall-effect keyboards, achieving up to injecting 12,553 keystrokes per minute for effective swift attack injection.

C. Displacement Calibration

After the attack device is deployed, the victim may shift or rotate the keyboard. This poses challenges to both DualStrike's eavesdropping module and, especially, its injection model, which relies on precise per-key alignment. For example (Fig. 16), due to displacement, a victim's keystroke of "2" may be eavesdropped as "T". Similarly, during injection, to trigger key "2" the attacker must adjust the target to "T". Therefore, a calibration module needs to first estimate this displacement, then correct (i.e., remap) it accordingly.

To ensure accurate eavesdropping and injection with displacement, DualStrike employs a novel calibration scheme consisting of two steps: (1) estimating the displacement position and angle via keystroke inference and (2) remapping both the eavesdropped keystrokes and the intended injection targets, to recover the actual keys pressed and adjust the injected keys.

Step 1: Displacement estimation. Attackers can estimate displacement by prompting victims to input specific predetermined keys. Previous research has demonstrated similar setups [35], [44]. For instance, an attacker could set up public Wi-Fi access, requiring the victim to type a predetermined sequence as the Wi-Fi password. Notably, this calibration process is non-intrusive and does not require any software or hardware installation on the victim's device. Assuming the victim inputs a sequence of length n, and we process the data using the method described in Sec. V-A, we can obtain n key pairs: $(\text{key}_{\text{true}}, \text{key}_{\text{predicted}}, p)$, where key_{true} denotes the ground-truth keystroke in the sequence, key_{pred} is the corresponding inferred keystroke, and p represents the prediction confidence output by the MLP inference model. According to Sec. IV, the fixed positions of keys in the main keyboard area allow us to map them to a 2D coordinate system and derive key pair positions as $(\text{pos}_{\text{true}}, \text{pos}_{\text{predicted}}, p)$. We then apply a weighted least-squares (WLS) algorithm to estimate the displacement:

$$\min_{\text{dx},\text{dy},\theta} \sum_{i=1}^{n} \|\mathbf{pos}_{\text{predicted},i} - (\mathbf{R}(\theta)\cdot \mathbf{pos}_{\text{true},i} + [\text{dx}, \text{dy}]^\top)\|^2 \cdot \text{p}_i \quad (4)$$

Here, dx, dy, and θ represent the keyboard's displacement along the horizontal, vertical axes, and rotation angle, respectively. $\mathbf{R}(\theta)$ is the corresponding rotation matrix. We use p_i as the weight, assigning higher importance to key pairs with higher prediction probabilities.

Step 2: Remapping. Next, we use the displacement parameters dx, dy, and θ to calibrate both the eavesdropping and keystroke injection modules.

Eavesdropping: Because of displacement, a press intended for "2" may be sniffed as "T". We therefore apply an inverse transformation to the sniffed coordinate $\mathbf{pos}_{\text{eavesdrop}}$ to obtain the calibrated position $\mathbf{pos}^{\text{cali}}_{\text{predicted}}$:

$$\mathbf{pos}^{cali}_{\text{eavesdrop}} = \mathbf{R}^{-1}(\theta)\cdot(\mathbf{pos}_{\text{eavesdrop}} - [\text{dx}, \text{dy}]^\top)$$

We then compare $\mathbf{pos}^{\text{cali}}_{\text{eavesdrop}}$ with all key coordinates and select the nearest one as the actual key pressed(e.g., key "2").

Injection. Likewise, if the attacker intends to press key "2", we forward-transform its coordinate $\mathbf{pos}_{\text{inject}}$:

$$\mathbf{pos}^{cali}_{\text{inject}} = \mathbf{R}(\theta)\cdot \mathbf{pos}_{\text{inject}} + [dx,\, dy]^\top.$$

The key nearest to $\mathbf{pos}^{\text{cali}}_{\text{inject}}$ is chosen as the actual injection target (e.g., key "T").

VI. Evaluation

A. Experiment Setup

In our experiments, we evaluated six commercial off-the-shelf Hall-effect keyboards from different manufacturers. Their specifications are listed in Table III. Fig. 17 shows the specific setup of experiments. The attack device (Sec. V-B3) was powered solely by two 12 V batteries, with no additional hardware required. We first benchmark DualStrike's performance by placing the Hall-effect keyboard on the 3D-printed case. We further evaluate DualStrike's robustness in realistic scenarios by introducing varying material and thicknesses between the keyboard and attack device, as illustrated in Sec. VI-C2.

Evaluation sequences. We considered all letters, symbol keys, and whitespace, as shown in Table II, covering a total of 51 physical keys. Note that by using the "CapsLock" and "Shift" keys, one can expand the set of injectable characters. For example, pressing "CapsLock" followed by letters

TABLE III: Specifications of six Hall-effect keyboards.

ID	Type	Year	Size	Protocol	Key Travel	Selected Distance
1	Wooting 60HE [28]	2022	60%	Wired	0.1-4.0 mm	0.1 mm
2	SteelSeries Apex Pro [11]	2019	100%	Wired	0.1-4.0 mm	0.4 mm
3	Keydous NJ98-CP [19]	2023	100%	Wireless	0.2-3.7 mm	0.2 mm
4	Corsair K70 MAX [17]	2023	100%	Wired	0.4-3.6 mm	0.4 mm
5	Redragon M61 [27]	2023	60%	Wired	0.1-4.0 mm	0.1 mm
6	Drunkdeer A75 Pro [8]	2024	75%	Wired	0.2-3.8 mm	0.2 mm

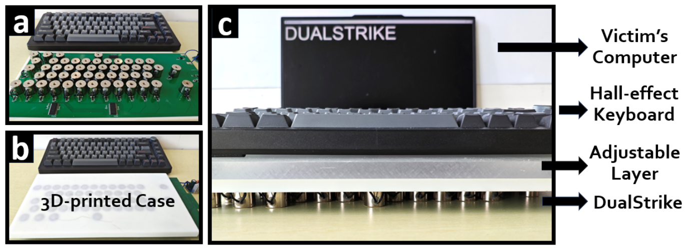


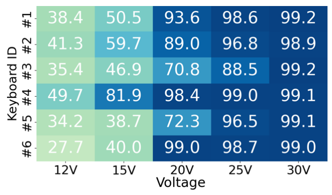
Fig. 17: (a) and (b) show the Hall-effect keyboard and the `DualStrike` device with a 3D-printed case. (c) illustrates the experimental setup, note that the adjustable layer can be replaced with different thickness and material. Here shows a 1-cm thickness acyclic plate.

enables uppercase letters, and pressing "Shift" along with symbol or numeric keys enables shifted characters such as "<>?:"—!@#$%^&*()". In total, we achieved 93 injectable characters in `DualStrike`. To evaluate `DualStrike`, we consider two types of sequences:

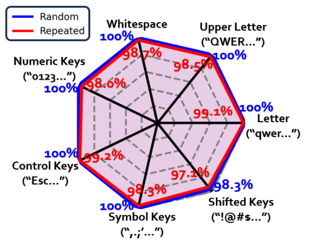
Type 1: Benchmark sequence. To cover all possible characters, we utilized a 930-keystroke sequence to benchmark the performance of `DualStrike`. Our benchmark sequence contains two components, i.e., a random sequence and a repeated characters sequence. Specifically, we first randomly shuffled the 93 characters five times to construct the random sequence. Next, we repeated each character five times to form the repeated sequence. The benchmark sequence enables analysis of key-wise performance under both random and repeated patterns. This benchmark sequence is used to evaluate key injection performance in Sec. VI-C1, Sec. VI-C2, and Sec. VI-D.

Type 2: Real-world sequence. To emulate real-world attacks, we evaluate `DualStrike` using the sequences introduced in Sec. III-A The corresponding experiments are conducted in Sec.VI-C3 and Sec. VI-E.

Evaluation metrics. We evaluate `DualStrike`'s performance using three metrics: **1) Benchmark accuracy:** We evaluated performance of benchmark sequence using *per-key accuracy*, defined as the proportion of correctly injected keystrokes. **2) Injection speed:** We used APM, i.e., the number of injected keystrokes per minute, to assess injection speed. **3) Real-world success rate:** For real-world sequences, we used *success rate* as the metric. A test instance is considered successful only if the entire attack sequence is reproduced without errors. The success rate is then computed as the ratio of successful instances to the total number of attempts.



(a) Per-key injection accuracy.



(b) Per-key accuracy across different key types in the benchmark sequence.

Fig. 18: The injection performance of `DualStrike`.

B. Performance of Eavesdropping

To test the performance of keystroke inference, we evaluated two Hall-effect keyboards (#1 and #6). Each keyboard and attack device were separated by a 10 mm thick wooden board to mimic real-world spacing, and data from the MLX90393 magnetometer array were collected to train the model. For each keyboard, the 51 keys were pressed 30 times each, and the dataset was divided into training and testing sets with a 7:3 split. The results show that our model achieves an accuracy of 99.41% and 99.54% on keyboards #1 and #6, respectively, demonstrating the feasibility of inferring keystrokes through magnetometer readings. The few misclassifications observed were primarily due to confusion between spatially adjacent keys, such as "N" and "M."

C. Performance of Keystroke Injection

1) Injection Accuracy and Speed: We first evaluate the accuracy of the attack and the speed of the keystroke injection. Specifically, as mentioned in Sec. V-B4, for each Hall-effect keyboard, T_{on} and T_{off} were determined through reverse engineering, and V_{attack} was adjusted using the buck converter to meet the Hall-effect keyboard's activation threshold.

Accuracy. We set V_{attack} to 12 V, 15 V, 20 V, 25 V, and 30 V. As shown in Fig. 18(a), we observe that the injection accuracy increased with the voltage. This is because keys in different rows sit at varying heights, and those positioned higher require more voltage to be triggered. At 30 V, the accuracy exceeds 98.9% for all keyboards, with keyboard #6 achieving a 99.2% accuracy. Note that this accuracy surpasses the level required by professional stenographers (96% in [6]).

We analyze the error pattern and find that all errors fell into the "missing" category, where the applied attack voltage (V_{attack}) is lower than the required activation threshold (V_{required}), resulting in failure to trigger the keystroke. Based on the results across different key types (Fig. 18(b)), nearly all types of random keys achieve an accuracy of 100%. Minor errors are observed in shifted keys (98.3%), which require simultaneous activation of "Shift" key and another key. For repeated keys, accuracy remains above 97.1%. Occasional errors stem from rapid consecutive keystrokes, e.g., "aaaaa" interpreted as "aaaa". For real-world input sequences, which typically follow a random key distribution, most key types achieve an injection accuracy of 100%. This enables

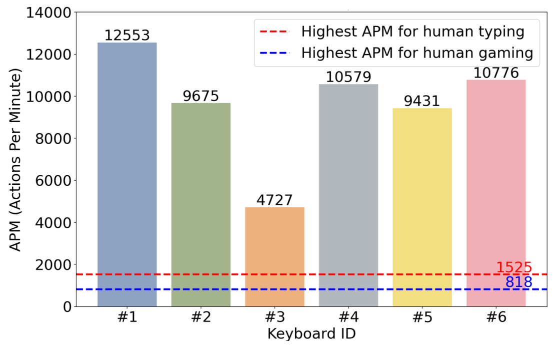


Fig. 19: APM results of six different Hall-effect keyboards.

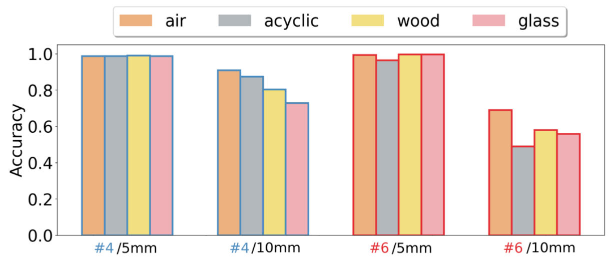
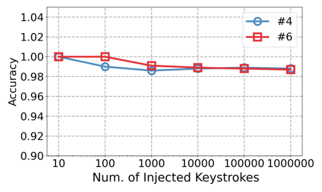
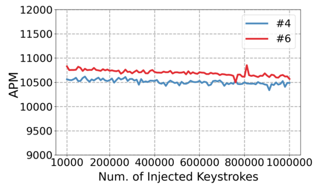


Fig. 20: `DualStrike`'s accuracy under different materials with thicknesses of 5 mm and 10 mm. "#4/5 mm" denotes #4 keyboard tested with material of 5 mm.



(a) Acc. during long-term typing



(b) APM during long-term typing

Fig. 21: `DualStrike` can maintain accuracy and APM in long-term typing, i.e.,injecting 1,000,000 keystrokes.

`DualStrike` to perform *high-precision sequence injection* even over long inputs, as we further illustrate in Sec. VI-C3.

APM. Based on the accuracy results in Fig. 18(a), we chose 30 V as the attack voltage to test `DualStrike`'s injection speed using a five-minute continuous typing session with the benchmark sequence. As shown in Fig. 19, we observed that `DualStrike` can achieve high injection speed. Specifically, all keyboards achieve an injection rate of over 4,000 APM, with keyboard #1 reaching as high as 12,553 APM. In contrast, the highest APM recorded in gaming by humans is 818. Thanks to our high-frequency spoofing design and the shorter scan cycles of Hall-effect keyboards compared to traditional keyboards, the proposed sync-free injection method enables rapid injections without additional hardware. This allows attackers to inject malicious commands faster than the typical human reaction times (e.g., 200 ms), leading to severe consequences, as we discussed in Sec. III-A.

2) *Varying Factors:* To evaluate `DualStrike`'s robustness, we assessed its injection performance under various conditions, including environmental factors such as table material and thickness, as well as long-term injection stability. We used two representative keyboards (#4 and #6), which differ in scanning circuit type and are among the latest Hall-effect models available.

Table thicknesses and materials. Since attackers can put `DualStrike` underneath the table to conduct the attack, we studied the impact of different table specifications, i.e., materials and thicknesses. We conducted experiments with three commonly used materials: wood, acrylic, and glass. Each material was tested at two different thicknesses: 5 mm and 10 mm. The keystroke injection performance in the air was used as the baseline. To accommodate a thicker desk, we set V_{attack} to 40 V.

As shown in Fig. 20, `DualStrike` achieves nearly 100% accuracy across all tested materials at a thickness of 5 mm, and maintains over 85% accuracy even at 10 mm. This performance degradation, compared to the baseline in air, is primarily due to the higher magnetic permeability of the materials, which leads to increased attenuation of the magnetic field. Note that by increasing the voltage V_{attack}, the attacker can enhance injection performance, as discussed in Sec. V-B3. Alternatively, the attacker can carve a rectangular groove to seat the device, as illustrated in Fig. 1, thus accommodating greater desk thickness.

Sequence length. We conducted a long-term keystroke injection attack to validate the stability of `DualStrike`. We kept V_{attack} at 30 V. By continuously typing the benchmark sequence, we recorded accuracy at 10, 100, 10,000, 100,000, and 1,000,000 keystrokes. As shown in Fig.21, `DualStrike` maintains consistent accuracy and APM throughout this stress test. Notably, even with 1,000,000 keystrokes, keyboards #4 and #6 still maintain 98.8% and 98.7% accuracy.

3) *Real-world Sequence Injection:* We further assess `DualStrike`'s injection performance using three representative real-world attack sequences:

Malicious command: We emulated a file-deletion attack in which the attacker issued a six-key `rm -rf` command to delete a critical directory.

Private key leakage: Private keys are highly sensitive. We modeled an attack where the attacker executed a 77-key command to steal the victim's private key and transfer it to a remote server: scp /main/user/RSA/private_key.rsa https://badman.com/file/collected-victim/

File tampering: We emulated tampering with sensitive documents (e.g., press release drafts), similar to attacks reported in [25], [5]. First, we injected `Backspace` to erase the original content, then inserted English text of varying lengths (200–600 characters).

For each scenario, we executed the commands on keyboards #4 and #6, repeating each injection ten times. As shown in Fig. 22, `DualStrike` maintains a near-100% accuracy across varying sequence lengths. Even for the complex 600-character file-tampering attack, it achieves success rates of 100% and 90% on keyboards 4 and 6, respectively.

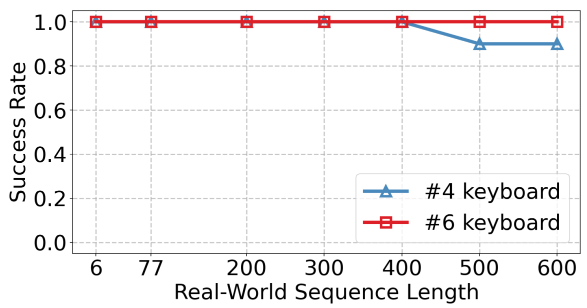


Fig. 22: DualStrike's flawless injection success rate for injecting real-world sequences of varying lengths.

D. Performance of Misalignment Calibration

We evaluated the misalignment calibration in two aspects: (1) displacement estimation accuracy and (2) injection accuracy after calibration. Tests were conducted on two representative Hall-effect keyboards (#1 and #6), under three misalignment scenarios: 1) Displacement along the x-axis: offsets of 2 cm and 4 cm. 2) Combined displacement: 3 cm in the x-direction and 2 cm in the y-direction. 3) Rotation about the keyboard center: angles of 5°. For predetermined keys, we selected eight well-distributed keys: `q, r, u, p, z, v, m, /`, and conducted experiments using subsets of these keys.

Accuracy of displacement estimation. Table IV presents the errors for two keyboards using eight calibration keys under different displacements. We measure the displacement error by the absolute distance between estimated and true displacements, i.e., $\sqrt{(dx_{\text{pred}} - dx_{\text{true}})^2 + (dy_{\text{pred}} - dy_{\text{true}})^2}$, and the angular error by the absolute difference from the true angle. As shown in the results, even when the displacement reaches 4 cm, the estimation error for both Keyboard #1 and Keyboard #6 remains below 0.5 cm, demonstrating the accuracy of our algorithm. To investigate how the number of calibration keys affects actual calibration, we vary the number of keys from two to eight. For each number less than eight, we consider all subsets of that size and compute the mean and STD of their errors. As shown in Fig. 23, the error gradually decreases and stabilizes as more calibration keys are used. We also observed that selecting keys with wider spatial distribution results in improved calibration accuracy. To balance calibration effectiveness with stealth, we select six keys—"qruzvm"—as the default inputs for our calibration algorithm.

Keystroke injection accuracy. After determining the calibration sequence, we evaluated the accuracy of keystroke injection following the application of calibration algorithm, as described in Sec.V-C. We selected keys from the overlapping region and composed the benchmark sequence as described in Sec. VI-A for injection verification. The result is shown in Table IV. We notice a slightly inferior performance compared with the performance with zero displacement. This is because even with alignment calibration, the attack key and the actual key still have errors of less than *half the key spacing*, requiring higher voltage for activation. Nevertheless, through calibration, we achieve accurate injection performance under large displacement (e.g., 4 cm), whereas current state-of-the-art methods [41] are only effective within a displacement range of less than 14 mm.

TABLE IV: Displacement calibration results. '→2cm' denotes a horizontal displacement of $dx = 2\,\text{cm}$, while '$\nearrow (3,2)$' denotes $dx = 3\,\text{cm}$ and $dy = 2\,\text{cm}$.

Scenarios	Keyboard	Displacement			Rotation
		→2cm	→4cm	↗(3,2)	5°
Error	#1 keyboard	0.40 cm	0.41 cm	0.44cm	0.61°
	#6 keyboard	0.41 cm	0.47 cm	0.38cm	0.43°
Accuracy	#1 keyboard	98.7%	98.5%	99.0%	98.1%
	#6 keyboard	98.2%	98.4%	98.8%	98.8%

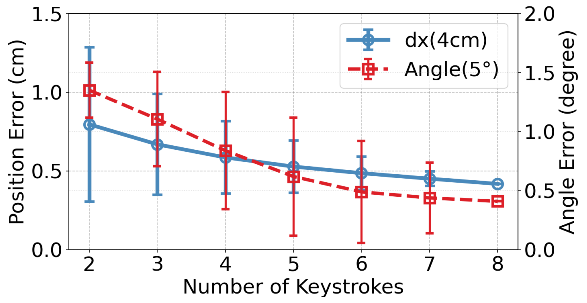


Fig. 23: The accuracy of DualStrike's calibration algorithm under different number of key pairs.

E. End-to-end Attack

We further evaluated DualStrike's ability to execute end-to-end attacks by combining misalignment calibration, eavesdropping, and injection modules as shown in Fig. 4. We emulated two real-world end-to-end attack scenarios:

Ghost Login: The attacker can utilize DualStrike to capture the login password of the victim and subsequently emulate login attempts.

Sudo Hijack: Based on the eavesdropping module, the attacker can use DualStrike to identify the root password after "sudo". Then, the attacker can reuse the captured password to perform two follow-up attacks: 1) `sudo rm -rf / + password` to delete critical files. 2) `sudo mkfs.ext /dev/sda + password` to format the system disk.

In both scenarios, we introduced keyboard misalignment at the beginning to emulate real-world victim interactions. Misalignment settings followed setup in Sec. VI-D. DualStrike first performed calibration to estimate the displacement parameters. We then emulated victim behavior by entering credentials on the displaced keyboard, including a login password (Ghost Login) and a sudo command with a root password (Sudo Hijack). Both passwords were 8-digit strings randomly generated using a cryptographically secure number generator [33]. DualStrike recorded the full keystroke sequence between the start and end of input as the inferred password. In the Sudo Hijack scenario, once a `sudo` command was detected, the subsequent keystrokes were extracted as the root password. Finally, the two follow-up attacks described above were executed.

TABLE V: End-to-end attack result, shown as success rate.

Scenarios	Keyboard	Displacement			Rotation
		→2cm	→4cm	↗(3,2)	5°
Ghost Login	#1 keyboard	100%	100%	100%	100%
	#6 keyboard	100%	100%	100%	100%
Sudo Hijack-1	#1 keyboard	100%	100%	100%	100%
	#6 keyboard	100%	90%	100%	100%
Sudo Hijack-2	#1 keyboard	100%	100%	100%	100%
	#6 keyboard	100%	100%	100%	100%

We tested two representative keyboards (Keyboards #1 and #6), repeating each scenario 10 times. As shown in Table V, `DualStrike` achieves near-100% accuracy across all settings, including displacements up to 4 cm—demonstrating its robustness for real-world end-to-end attacks. By proactively capturing login and sudo credentials, `DualStrike` can enable broader and more impactful attacks. For example, after Ghost Login, an attacker can impersonate the user to access system files or extract additional credentials. Once the sudo password is captured, `DualStrike` can bypass security defenses.

F. Energy Cost

We measured the battery current to assess the energy consumption. Specifically, when $V_{\text{attack}} = 30\,\text{V}$, the two 12 V cells draw 0.12 A and 0.16 A on average, corresponding to 1.44 W and 1.92 W. With a 2800 mAh capacity, each battery can sustain over 16 h of continuous injection, enabling long-term `DualStrike` operation.

VII. Defense

A. Existing Magnetic Shielding Methods

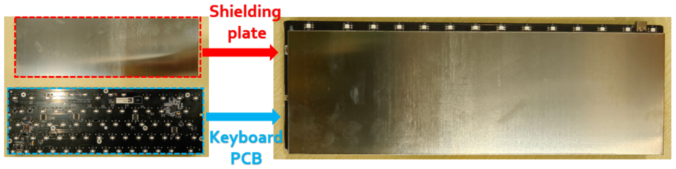
Existing countermeasures against external magnetic interference are widely adopted in devices such as computers and smartphones. These systems often use high-permeability materials (e.g., steel, permalloy) to suppress electromagnetic interference (EMI) [20], [51]. Similarly, some mechanical and membrane keyboards incorporate steel plates beneath the matrix circuit—e.g., Sunread [9] and Cherry Stream [7], which is effective in protecting the keyboard from EMI attacks [41].

We evaluated `DualStrike`'s injection effectiveness with two commonly used magnetic shielding materials: steel and 1J85 permalloy. The latter is more expensive but is known for its superior magnetic shielding properties due to its higher magnetic permeability [51], [42]. We varied the thickness of these materials, testing plates of 0.2mm and 0.5mm thicknesses. We utilized the #4 and #6 keyboards, setting the V_{attack} to 30V, 35V, and 40V. Following the setting of conventional keyboards, we placed the shielding plate beneath the Hall-effect keyboard PCB, as shown in Fig. 24(a). The injection accuracy was measured using the same method as in Sec. VI-A.

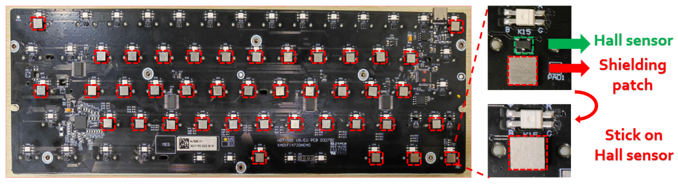
As shown in Table.VI, the results indicate that conventional shielding material, such as steel, has minimal impact on `DualStrike`'s injection. This is because, magnetic shielding only attenuates magnetic fields and can be compromised by a stronger magnetic field with higher voltage.

TABLE VI: Shielding results

Material	Thickness	Keyboard #4			Keyboard #6		
		30V	35V	40V	30V	35V	40V
Steel	0.2mm	99.0%	98.8%	99.1%	98.9%	98.7%	98.7%
	0.5mm	99.2%	99.0%	99.0%	98.6%	99.2%	99.0%
Mu metal	0.2mm	93.4%	99.1%	99.0%	73.1%	86.2%	98.7%
	0.5mm	7.8%	16.2%	41.0%	0%	0%	11.2%
Per-sensor shielding		**0%**	**0%**	**0%**	**0%**	**0%**	**0%**



(a) Existing Magnetic shield with shielding plate



(b) Our proposed per-sensor shielding

Fig. 24: A new hardware defense against `DualStrike`.

B. Defenses against `DualStrike`

`DualStrike` targets the Hall sensors and the corresponding magnets above, each of which has a compact size ($4mm \times 4mm$) compared to the keyboard area ($300mm \times 100mm$). Therefore, we propose a *per-sensor shielding* approach, where small-sized shielding patches (i.e., 1.5 mm thick, $5mm \times 5mm$ square mu-metal patches) are attached only to the back of the Hall-effect sensors on targeted keys, shown in Fig. 24(b).

This method effectively blocks magnetic spoofing from below and attenuates external sensors' ability to detect internal keystroke fields, defending against both eavesdropping and injection. Using only 10% of the material required for full keyboard shielding, it reduces the keystroke-induced field below the eavesdropping detection threshold (Sec. V-A), and—as shown in Table VI—lowers injection accuracy to 0%. We further suggest that manufacturers can selectively shield critical keys (e.g., `Space`, `Enter`, `Shift`, `OS`) to block key-based attacks at minimal cost.

VIII. Related Works

Keyboard security. Existing studies have primarily focused on keystroke inference attacks targeting membrane or mechanical keyboards, leveraging side-channel signals such as visual [56], [58], acoustic [46], [60], and electromagnetic [43] information to infer current keystrokes. However, these works can not achieve keystroke injection attacks. Researchers have also investigated active spoofing attacks [48], which involve the use of reprogrammed USB devices to mimic keyboards injecting malicious fake keystrokes. However, these invasive attacks require physical modification of keyboards and can be easily prevented through authenticated connections [37], [39]. Moreover, mainstream operating systems like Windows and security vendors such as Kaspersky already provide built-in defenses against such USB-based spoofing attacks [22], [31]. GhostType [41] enables contactless keystroke injection by inducing false voltage on keyboard GPIO pins via electromagnetic interference (EMI). However, it only focuses on existing membrane and mechanical keyboards and cannot achieve per-key keystroke injection since the attack vector (EMI) can only perform attack injection per scanning circuits. In contrast, `DualStrike` is the first to achieve both eavesdropping and per-key injection on commodity keyboards. It exploits magnetic sensing vulnerabilities in Hall-effect switches, enabling high-speed (APM), precise attacks. This dual capability allows more practical and robust real-world attack scenarios.

Hall-effect sensor attack. Hall-effect sensors have been widely used in cyber-physical systems (CPS) because of their high accuracy and efficiency [45], [34], [53]. These compact sensors leverage the Hall effect [49] to measure the magnetic field strength, which can produce a voltage proportional to the applied magnetic field and thus reflect the strength of the magnetic field flux density. Hence, an attacker can compromise the measurement by injecting fake external magnetic fields for malicious purposes, e.g., causing spoofing and denial-of-service (DoS) attacks. For example, prior work [36] achieves a non-invasive attack on the grid-tied solar inverters by spoofing the Hall sensor of the inverter with an external magnetic field. [52] utilizes an electromagnetic actuator near the wheel speed sensor to inject the malicious magnetic field signal, thus spoofing the measured wheel speeds of Anti-lock Braking Systems (ABS). In this work, we are the *first* to use magnetic injection to attack the fast-emerging Hall-effect keyboards. To achieve this, we address several design challenges, such as ensuring accurate and high-speed magnetic injection while maintaining adaptability to various Hall-effect keyboard configurations.

IX. Conclusion

We present `DualStrike`, the first viable scheme that can achieve both eavesdropping and injection on commodity keyboards, in particular, Hall-effect keyboards. We perform reverse engineering efforts of the latest Hall-effect models, thus revealing three novel findings of their vulnerabilities. We introduce a novel key injection and eavesdropping hardware design. Specifically, the eavesdropping module can collect sensitive context information and help boost the robustness of `DualStrike`. The new injection module can accurately target arbitrary keys on the latest Hall-effect keyboard models. Finally, we propose a new hardware-based countermeasure. We will disclose these vulnerabilities to manufacturers to help them address any related risks.

Acknowledgements

We thank the anonymous reviewers and shepherd for their suggestions. This work was supported by the National Natural Science Foundation of China under Grant No. 62472283.

References

[1] Honeywell introduced the world's first hall effect keyboard. http://web.archive.org/web/19990422030528/http://www.honeywell.com/sensing/about/history.stm, 1998.
[2] debouncing. https://my.eng.utah.edu/~cs5780/debouncing.pdf, 2008.
[3] Logitech hall effect keyboard. https://me.ign.com/en/logitech-g-pro-x-tkl-rapid/229139/review/logitech-moves-to-magnetic-switches-with-the-pro-x-tkl-rapid-keyboard-the-results-are-great, 2009.
[4] Newest lost price hall effect keyboard. https://detail.tmall.com/item.htm?abbucket=18&id=844075013790&mi_id=0000gnbSy3w5twSUx5pqluOuqjY_ZmzKF9IwIGk31dBdl_Y&ns=1&priceTId=2150440f17574279679651634e121b&skuId=6062957174941&spm=a21n57.1.hoverItem.1&utparam=%7B%22aplus_abtest%22%3A%22c8e448cfcc195092caa6f3f0f1458bea%22%7D&xxc=taobaoSearch, 2009.
[5] brazil disinformation. https://www.dw.com/en/brazil-police-to-probe-allegations-of-election-disinformation-on-whatsapp/a-45965369), 2018.
[6] stenographers type accuracy. https://stenovate.com/how-stenographers-type-faster-than-the-speed-of-speech/, 2020.
[7] Cherry stream. https://www.cherry-world.com/stream-keyboard, 2022.
[8] Drunkdeer a75 pro. https://drunkdeer.com/products/drunkdeer-a75-pro-wired-actuation-distance-adjustable-magnetic-switch-keyboard?variant=48570245939503, 2022.
[9] Sunread skb886s. http://www.sunreed.com.cn/h-pd-48.html, 2022.
[10] Micro switch sw series. https://deskthority.net/wiki/Micro_Switch_SW_Series, 2023.
[11] Steelseries apex pro. https://cn.steelseries.com/gaming-keyboards/apex-pro, 2023.
[12] Understanding rl circuit operation and time constant. https://eepower.com/technical-articles/understanding-rl-circuit-operation-and-time-constant/#, 2023.
[13] Aerospace hall effect speed & position magnetic sensors. https://sensorso.com/aerospace.html, 2024.
[14] Ampere's law. https://byjus.com/physics/amperes-law/, 2024.
[15] Ansi layout standard of keyboard. https://en.wikipedia.org/wiki/Keyboard_layout, 2024.
[16] Ansys. https://www.ansys.com/zh-cn, 2024.
[17] Corsair k70 max rgb magnetic-mechanical gaming keyboard. https://www.corsair.com/us/en/p/keyboards/ch-910961g-na/k70-max-rgb-magnetic-mechanical-gaming-keyboard-adjustable-corsair-mgx-switches-steel-grey-ch-910961g-na, 2024.
[18] Gaming keyboard switches. https://steelseries.com/innovation/keyboard-gaming-switches?srsltid=AfmBOoqzR5brrBsw7xNSuDoytS-LcXx1Aj1DcTN3TCcr92ArNPIFAD1e, 2024.
[19] Keydous nj98 keyboards. https://keydous.store/pages/keydous-nj98, 2024.
[20] Magnetic permeability. https://en.wikipedia.org/wiki/Permeability_(electromagnetism), 2024.
[21] Melexis: Mlx90393 triaxis micropower magnetometer. https://www.melexis.com/-/media/files/documents/datasheets/mlx90393-datasheet-melexis.pdf, 2024.
[22] Microsoft defender. https://techcommunity.microsoft.com/blog/windows-itpro-blog/keylogging-malware-protection-built-into-windows/4256289, 2024.
[23] Military speed & position sensors. https://sensorso.com/military.html, 2024.
[24] Ocean & underwater magnetic sensors. https://sensorso.com/marine.html, 2024.
[25] Pizzagate conspiracy theory. https://en.wikipedia.org/wiki/Pizzagate_conspiracy_theory), 2024.
[26] Qwerty layout standard of keyboard. https://en.wikipedia.org/wiki/QWERTY, 2024.
[27] Redragon m61 wired magnetic switch mechanical keyboard. https://mechkeys.com/products/redragon-m61?variant=44572643328223, 2024.
[28] Wooting 60he. https://wooting.io/zh-CN/wooting-60he, 2024.
[29] Wooting recommand. https://wooting.io/zh-CN/post/the-best-keyboard-switches-for-gaming, 2024.

[30] Anonymous demo site of DualStrike. https://sites.google.com/view/magkey-anonymous?usp=sharing, 2025.

[31] Kaspersky defender. https://support.kaspersky.it/help/NextPro/1.0/en-US/230866.htm, 2025.

[32] Lt8361 datasheet. https://www.analog.com/media/en/technical-documentation/data-sheets/lt8361.pdf, 2025.

[33] Python secret module. https://docs.python.org/3/library/secrets.html, 2025.

[34] A. Ajbl, M. Pastre, and M. Kayal. A fully integrated hall sensor microsystem for contactless current measurement. *IEEE Sensors Journal*, 13(6):2271–2278, 2013.

[35] K. Ali, A. X. Liu, W. Wang, and M. Shahzad. Keystroke recognition using wifi signals. In *Proceedings of the 21st annual international conference on mobile computing and networking*, pages 90–102, 2015.

[36] A. Barua and M. A. A. Faruque. Hall spoofing: A Non-Invasive DoS attack on Grid-Tied solar inverter. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 1273–1290. USENIX Association, Aug. 2020.

[37] P. Cronin, X. Gao, H. Wang, and C. Cotton. Time-print: Authenticating usb flash drives with novel timing fingerprints. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1002–1017, 2022.

[38] A. J. Feldman, J. A. Halderman, and E. W. Felten. Security analysis of the diebold accuvote-ts voting machine. 2006.

[39] F. Griscioli, M. Pizzonia, and M. Sacchetti. Usbcheckin: Preventing badusb attacks by forcing human-device interaction. In *2016 14th Annual Conference on Privacy, Security and Trust (PST)*, pages 493–496, 2016.

[40] A. Jain, R. Bansal, A. Kumar, and K. Singh. A comparative study of visual and auditory reaction times on the basis of gender and physical activity levels of medical first year students. *International journal of applied and basic medical research*, 5(2):124–127, 2015.

[41] Q. Jiang, Y. Ren, Y. Long, C. Yan, Y. Sun, X. Ji, K. Fu, and W. Xu. Ghosttype: The limits of using contactless electromagnetic interference to inject phantom keys into analog circuits of keyboards. 01 2024.

[42] D. Jiles. *Introduction to magnetism and magnetic materials*. CRC press, 2015.

[43] W. Jin, S. Murali, H. Zhu, and M. Li. Periscope: A keystroke inference attack using human coupled electromagnetic emanations. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, CCS '21, page 700–714, New York, NY, USA, 2021. Association for Computing Machinery.

[44] W. Jin, S. Murali, H. Zhu, and M. Li. Periscope: A keystroke inference attack using human coupled electromagnetic emanations. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 700–714, 2021.

[45] E. Kabalci and Y. Kabalci. A wireless metering and monitoring system for solar string inverters. *International Journal of Electrical Power Energy Systems*, 96:282–295, 2018.

[46] 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*, MobiCom '15, page 142–154, New York, NY, USA, 2015. Association for Computing Machinery.

[47] C. Mainka, V. Mladenov, and S. Rohlmann. Shadow attacks: Hiding and replacing content in signed pdfs. In *NDSS*, 2021.

[48] K. Nohl and J. Lell. Badusb-on accessories that turn evil. *Black Hat USA*, 1(9):1–22, 2014.

[49] E. Ramsden. *Hall-effect sensors: theory and application*. Elsevier, 2011.

[50] S. Rohlmann, V. Mladenov, C. Mainka, D. Hirschberger, and J. Schwenk. Every signature is broken: On the insecurity of microsoft {Office's}{OOXML} signatures. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 7411–7428, 2023.

[51] T. Sato, T. Yamada, and M. Kobayashi. Magnetic shielding material, Sept. 3 1991. US Patent 5,045,637.

[52] Y. Shoukry, P. Martin, P. Tabuada, and M. Srivastava. Non-invasive spoofing attacks for anti-lock braking systems. In G. Bertoni and J.-S. Coron, editors, *Cryptographic Hardware and Embedded Systems - CHES 2013*, pages 55–72, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[53] Y.-P. Tsai, K.-L. Chen, and N. Chen. Design of a hall effect current microsensor for power networks. *IEEE Transactions on Smart Grid*, 2(3):421–427, 2011.

[54] J. G. Van Bladel. *Electromagnetic fields*, volume 19. John Wiley & Sons, 2007.

[55] J. Wang, S. Wang, Y. Iravantchi, M. Wang, A. Sample, K. G. Shin, X. Wang, C. Zhou, and D. Chen. Metro: Magnetic road markings for all-weather, smart roads. In *Proceedings of the 21st ACM Conference on Embedded Networked Sensor Systems*, SenSys '23, page 280–293, New York, NY, USA, 2024. Association for Computing Machinery.

[56] Z. Yang, Y. Chen, Z. Sarwar, H. Schwartz, B. Y. Zhao, and H. Zheng. Towards a general video-based keystroke inference attack. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 141–158, Anaheim, CA, Aug. 2023. USENIX Association.

[57] C. Yu, Y. Xiao, J. Lu, Y. Li, Y. Li, L. Li, Y. Dong, J. Wang, J. Shi, D. Bo, et al. File hijacking vulnerability: The elephant in the room. In *Proceedings of the Network and Distributed System Security Symposium. San Diego, CA, USA: Internet Society*, 2024.

[58] Q. Yue, Z. Ling, X. Fu, B. Liu, K. Ren, and W. Zhao. Blind recognition of touched keys on mobile devices. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, page 1403–1414, New York, NY, USA, 2014. Association for Computing Machinery.

[59] Z. Zhan, Y. Yang, H. Shan, H. Wang, Y. Jin, and S. Wang. Voltschemer: Use voltage noise to manipulate your wireless charger. *arXiv preprint arXiv:2402.11423*, 2024.

[60] T. Zhu, Q. Ma, S. Zhang, and Y. Liu. Context-free attacks using keyboard acoustic emanations. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, page 453–464, New York, NY, USA, 2014. Association for Computing Machinery.

Appendix A
Artifact Appendix

A. Description & Requirements

This artifact provides a complete framework for developing `DualStrike`, a new attack module that can perform both real-time keyboard listening and non-invasive, per-key keystroke injection on Hall-effect keyboards, while also incorporating a calibration mechanism to mitigate real world disruptions such as keyboard displacements. The demo video of `DualStrike` can be found in the link: https://sites.google.com/view/magkey-anonymous?usp=sharing.

1) How to access: The source for our `DualStrike` artifact can be accessed by cloning or downloading the repository from the following GitHub link: https://github.com/blankchenxm/DualStrike. Our repository contains all design files, source code, scripts and a minimal dataset needed to reproduce our main experiments. For archival and long-term availability, the artifact is also preserved on Zenodo with the DOI: https://doi.org/10.5281/zenodo.17033644. The directory layout is as follows:

- **Hardware**: Source files for building the `DualStrike` attack device, as described in Fig. 14 and Sec. V-C.
- **Firmware**: Arduino framework code to run all `DualStrike` functions—keystroke injection, eavesdropping, and calibration—in real time and locally on the MCU control module.
- **Software**: Python code for collecting training data of MLP model in the eavesdropping pipeline (Sec. V-A) and for offline data analysis, supporting both eavesdropping and calibration.
- **3D model**: 3D design file for the `DualStrike` base, as described in Fig. 17.
- **demo**: A demo video presenting the full procedure of the main experiments and showing the results.

- **README.md**: Detailed instructions to build the system and use `DualStrike`.

2) Hardware dependencies: Verification of `DualStrike` requires the following specific hardware:

- **Attack device**, including:
  1) **Main PCB**: The MCU control module (MDBT42Q) runs the complete pipeline locally and in real-time, i.e., keystroke injection, inference, and displacement calibration. The magnetometer array (2×4 MLX90393 sensors) provides eavesdropping; and the switching module drives the electromagnets. Note that an additional flashing board (e.g., CP2104 chip) is required to program the PCB. A detailed bill of materials (BOM) for each board is available in the repository.
  2) **Electromagnets**: 51 electromagnets of 16 mm diameter × 25 mm height mounted in the PCB's designated slots.
  3) **Power supply**: Two 12 V batteries and two boost converters, connected to the PCB via Dupont cables.
- **3D-printed base**: Printed in PLA material to support the PCB and its 51 electromagnets, featuring a 1 mm wall thickness at each electromagnet mounting point.
- **Hall-effect keyboard**: Serves as the attack target; connected via USB to any desktop or laptop.
- **Computing**: Note that any common desktop or laptop can be used solely to connect the Hall-effect keyboard and display keystroke injection results. No special GPU is required, as a standard CPU handles all processing.

3) Software dependencies: The following software and development tools are required:

- **Arduino IDE (v2.3.6)**: Used to program the MCU with arduino code.
- **Python Interpreter (Python 3.10)**: Runs the Python implementation of `DualStrike` attack scripts. We also provide the necessary Python library dependencies. Please refer to the "requirements.txt" file in the "3.Software" directory.

B. Artifact Installation & Configuration

To build `DualStrike`, the following steps are required:

1) Fabricate the hardware: Design and fabricate all the components in attack device, and the 3D-printed base.
2) Assemble the attack device: Solder the 51 electromagnets, battery, and DC–DC converter to the PCB, then mount the board on the 3D-printed base.
3) Program the device: Use Arduino to program the `DualStrike` firmware via the flashing board.
4) Set up the experiment environment: As shown in Fig. 17, invert the attack device so that the main PCB rests on the table with the 3D-printed base facing upward. Place the Hall-effect keyboard on the base. To replicate the "Table thicknesses and materials" test (Sec. VI-C), insert a removable layer of the specified thickness and material between the keyboard and the `DualStrike` device.
5) Run the `DualStrike` pipeline: Use the **Python implementation** (an offline version of our sensing algorithm framework) to process and visualize eavesdropping and calibration data. The provided dataset and source code are aligned with the experiments demonstrated in our demo. Full functions can be executed in real time on the MCU with the **Arduino implementation**.

Configuration: If you are using one of the six Hall-effect keyboards characterized in our paper, select its preset in the code. For other models, measure the scanning cycle as described in Sec. IV-B to obtain optimal `DualStrike` performance.

C. Experiment Workflow

After installation, we first collect the eavesdropping training data described in Sec. VI-B. The user streams magnetometer readings over the serial port while pressing each key, then trains the eavesdropping MLP model using **Python Implementation** and update **Arduino Implementation**'s model.

For the Arduino implementation, the firmware offers six operational modes, with results streamed live on the serial console. Full details are provided in "2.Firmware/DualStrike_Arduino/README.md":

- **LISTENER**: User presses the Hall-effect keyboard, `DualStrike` predicts each keystroke and displays the predicted result and confidence.
- **ATTACKER**: The user defines a custom keystroke injection sequence in the code; with the keyboard connected to a computer, `DualStrike` performs injection, making the injected keystrokes visible in any open text editor.
- **CALIBRATION**: After the keyboard is displaced from its aligned position, the user enters the pre-defined calibration sequence, and `DualStrike` calculates and displays the resulting displacement offset.
- **LISTENER_AFTER_CALIBRATION**: Once displacement and calibration are complete, pressing the keyboard prompts `DualStrike` to correct the eavesdropping output, and serial console shows the keys being pressed.
- **ATTACKER_AFTER_CALIBRATION**: Under the same post-calibration conditions, `DualStrike` compensates the user-specified injection sequence so that the attack produces the intended keystrokes.
- **END_TO_END**: Corresponds to end-to-end attack experiment in Sec.VI-E. `DualStrike` calibrates after displacement, eavesdrops (e.g., captures a password), waits 30 seconds, then concatenates the captured data with the attack text and injects the combined sequence.

For the offline Python implementation, we provide an offline version of `DualStrike` that supports all functionalities except for keystroke injection. This implementation allows analysis of both eavesdropping and calibration, producing the keystroke visualization shown in Fig. 11. We also provide a dataset that enables users to replay our experiments offline without requiring the specific hardware setup. For the eavesdropping and calibration experiments, users can reproduce the results demonstrated in our demo video using the offline sensing algorithm framework. For the keystroke injection experiment, since it requires a Hall-effect keyboard as the carrier, we instead recorded the injection results and corresponding timestamps from the demo video experiment for offline demonstration. Full details and procedures are provided in "README.md #Getting Started with DualStrike".

D. Major Claims

- **C1: Real-time eavesdropping accuracy.**
  `DualStrike`'s eavesdropping module identifies each pressed key with more than 99% accuracy.
  Proven by experiment E1: Sec. VI-A, *Performance of Eavesdropping*.
- **C2: High-speed, per-key injection.**
  `DualStrike` achieves non-invasive keystroke injection at up to 12,553 APM (keyboard#1) while maintaining at least 98.9% accuracy across six commercial keyboards.
  Supported by experiment E2: Sec. VI-B, *Injection Accuracy and Speed* (Fig. 18, Fig. 19) and experiment E3: Sec. VI-B, *Real-world Sequence Injection* (Fig. 22).
- **C3: Robust displacement calibration.**
  `DualStrike`'s calibration module compensates for up to 4 cm lateral shift or 5° rotation, restoring injection accuracy to above 98.5%.
  Demonstrated by experiment E4: Sec. VI-D, *Performance of Misalignment Calibration* (Table IV, Fig. 23); and experiment E5: Sec. VI-E, *End-to-end Attack* (Table V).

E. Evaluation

We detail the evaluation workflow for three core experiments that show the repeatability of `DualStrike`: (E1) quantifying eavesdropping performance, (E3) measuring the success rate of real-world sequence injection, and (E5) performing an end-to-end attack after introducing displacement.

1) Eavesdropping performance: We assess the performance of eavesdropping: the entire process, including data collection, model training, and subsequent accuracy testing, takes approximately two hours by human.

Preparation: Train the MLP model and update C file of model parameters for Arduino code.

Execution: Select the *LISTENER* mode in the Arduino code implementation to monitor keystrokes in real time.

Results: During training, the evaluation accuracy reaches 98.86% on keyboards#1; the serial console displays both the predicted keys and their confidence scores in real time.

2) Real-world keystroke sequence injection: We assess the success-rate of real-world sequence injection. The whole process takes about 30 minutes by human.

Execution: Select the *ATTACKER* mode in the Arduino code, specify an attack text longer than 500 characters, and record the injected keystrokes on the host computer or laptop.

Results: The injected keystroke sequence is compared to the original; an exact match indicates success, yielding an overall keystroke-injection success rate of 90%.

3) End-to-end attack: We assess the end-to-end attack. The whole process takes about 1 hour by human.

Preparation: Manually displace the keyboard from its aligned position: e.g., 3 cm to the right and 2 cm upward.

Execution: Choose *END_TO_END* mode. Users first enter the prescribed calibration sequence; the firmware auto-calibrates and eavesdrops to capture user input (e.g., an 8-digit random password). Once input is complete, it waits 30 seconds, concatenates the captured string with the attack text, compensates for the offset, and injects the combined sequence.

Results: The whole end-to-end attack procedure reaches a 90% success injection rate.

F. Customization

None.

G. Notes

`DualStrike` system contains dedicated hardware designs, e.g., the specific PCB, Hall-effect keyboard. Therefore, during the artifact evaluation process, we will enable evaluators to observe the hardware setup for assessment, such as via Zoom.