Non-intrusive and Unconstrained Keystroke Inference in VR Platforms via Infrared Side Channel

Tao Ni, Yuefeng Du, Qingchuan Zhao*, Cong Wang*
Department of Computer Science, City University of Hong Kong
{taoni2, yf.du, cs.qczhao, congwang}@cityu.edu.hk

***Abstract*—Virtual Reality (VR) technologies are increasingly employed in numerous applications across various areas. Therefore, it is essential to ensure the security of interactions between users and VR devices. In this paper, we disclose a new side-channel leakage in the constellation tracking system of mainstream VR platforms, where the infrared (IR) signals emitted from the VR controllers for controller-headset interactions can be maliciously exploited to reconstruct unconstrained input keystrokes on the virtual keyboard non-intrusively. We propose a novel keystroke inference attack named `VRecKey` to demonstrate the feasibility and practicality of this novel infrared side channel. Specifically, `VRecKey` leverages a customized 2D IR sensor array to intercept ambient IR signals emitted from VR controllers and subsequently infers *(i)* character-level key presses on the virtual keyboard and *(ii)* word-level keystrokes along with their typing trajectories. We extensively evaluate the effectiveness of `VRecKey` with two commercial VR devices, and the results indicate that it can achieve over 94.2% and 90.5% top-3 accuracy in inferring character-level and word-level keystrokes with varying lengths, respectively. In addition, empirical results show that `VRecKey` is resilient to several practical impact factors and presents effectiveness in various real-world scenarios, which provides a complementary and orthogonal attack surface for the exploration of keystroke inference attacks in VR platforms.**

I. Introduction

Virtual Reality (VR) technology has provided a novel human-computer interaction (HCI) paradigm that revolutionizes the way people communicate with digital content by creating immersive and interactive virtual environments. In particular, a VR system usually consists of a headset running particular operating systems and rendering digital content on the head-mounted display (HMD) as well as two handheld VR controllers to facilitate interactions. This new mobile platform transcends traditional screen-based displays with spatial interactions alongside multi-sensory feedback, which has led to its immense popularity in recent years.

Despite its innovative and immersive interactions, people still have to type keystrokes on a virtual keyboard for certain functionalities, *i.e.*, entering passwords to login accounts, which exposes VR users to a legacy vulnerability existing in common mobile platforms, that is, privacy leakage from keystroke inference. Recent studies (*e.g.*, [1]–[12]) have confirmed and demonstrated the feasibility of keystroke inference attacks in VR devices. In particular, these attacks target on the VR headset, either by installing malware to obtain motion sensor data [1]–[4], [11], recording videos of head movements and hand gestures [6], [9], [12], or utilizing side-channel information (*e.g.*, Wi-Fi CSI [5], acoustic signals in pressing controller buttons [10], unencrypted packets in network traffic [7], or gaze information of virtual avatars [8]). In general, these attacks depend on building inference models on various types of data traces to classify keys on a virtual keyboard for keystroke inference. However, the aforementioned keystroke inference attacks are constrained to specific scenarios and present limited scalability. Specifically, previous studies (*e.g.*, [1]–[3], [5]–[8], [10]) are restricted in closed-world classification for inferring keystrokes and inherently difficult to scale due to their reliance of developing multiple trace-based deep neural networks (DNN). In addition, several attacks that exploit side-channel information can only be effective under controlled conditions, *i.e.*, a fixed position without movement [5], a sufficient light condition or silent environment to record videos or audios of hand gestures [6], [9], [10], [12], or a shared virtual space or specific VR apps to monitor avatars' features [7], [8], which also limits their scalability and practicality.

Surprisingly, we disclose a novel side-channel attack from the neglected infrared (IR) leakage emitted from VR controllers, which presents high scalability with fewer

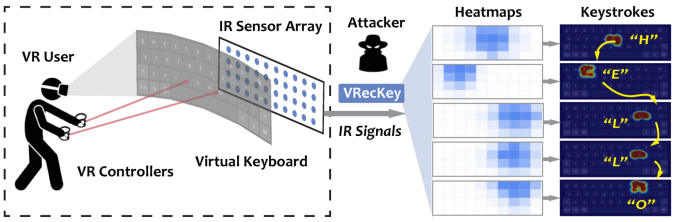


Fig. 1: Illustration of `VRecKey` attack: The victim types the virtual keyboard to input the keystroke "HELLO". Meanwhile, the attacker leverages a 2D IR sensor array to capture the IR signals emitted to the ambient environment and reconstruct the heatmaps to infer the virtual keystroke and its trajectory.

* Corresponding authors

Network and Distributed System Security (NDSS) Symposium 2025
24-28 February 2025, San Diego, CA, USA
ISBN 979-8-9894372-8-3
https://dx.doi.org/10.14722/ndss.2025.240096
www.ndss-symposium.org

Related VR Attacks	Attack Surface	Side Channel	NI	NPC	WMI	UKI	Distance	Character Level	Word Level
TyPose [1]	Motion sensors in VR headset	Malware	○	◐	○	○	–	○	◐ (82.0% T-5)
Zhang *et al.* [2]	Motion sensors in VR headset	Malware	○	◐	○	○	–	◐ (93.8% T-1)	○
Wu *et al.* [3]	Motion sensors in VR headset	Malware	○	◐	○	○	–	● (89.7% T-1)	● (84.9% T-3)
HoloLogger [4]	Motion sensors in VR headset	Malware	○	◐	●	●	–	● (73.0% T-1)	● (89.0% T-3)
VR-Spy [5]	Wi-Fi channel state data	Wi-Fi CSI data	●	○	○	○	1.3m	● (69.8% T-1)	○
Meteriz-Yıldıran *et al.* [6]	Users' hand gestures	Hand tracker/Camera	●	◐	○	○	0.6–0.8m	● (99.0% T-1)	● (87.0% T-5)
Su *et al.* [7]	Unencrypted Photon protocol	Network traffic	◐	●	○	●	–	● (97.6% T-1)	● (98.1% T-3)
GAZEploit [8]	Video of users' virtual avatars	Gaze information	◐	●	○	●	–	● (38.7% T-1)	● (85.9% T-5)
Gopal *et al.* [9]	Video of VR users' gestures	Camera	●	◐	●	○	3.0–6.0m	● (82.3% T-1)	● (57.0% T-3)
Heimdall [10]	Sound from VR controllers	Acoustic signal	●	◐	○	○	1.0–2.2m	● (96.5% T-1)	● (91.2% T-5)
VRecKey	IR signals from VR controllers	IR signal	●	◐	●	●	2.0–4.0m	● (85.8% T-1)	● (90.5% T-3)

TABLE I: Comparative analysis with related keystroke inference attacks in VR devices. "●": Yes, "○": No, "◐": Partly Yes, NI: Non-intrusive, NPC: No physical constraints, WMI: Without model inference, and UKI: Unconstrained keystroke inference.

physical constraints than previous keystroke inference attacks. The causality of IR leakage stems from the fact that most commercial VR devices rely on the *constellation tracking systems* [13] to track hand movements in typing virtual keystrokes. In this system, cameras on the headset continuously scan the infrared signals emitted by LEDs embedded in the VR controllers, allowing them to accurately determine the controllers' positions and orientations, as well as the virtual stick pointing to the virtual keyboard. As such, due to the line-of-sight (LoS) interactions inherent in VR platforms, this infrared side channel can be freely captured and extended to other constellation tracking-based VR systems, which currently dominate the market of commercial VR devices. On the other hand, it is less constrained since it does not require training models to infer keystrokes indirectly. Instead, it can directly recognize typed keys from the IR signal trajectory, enabling the reconstruction of the consecutive keystrokes.

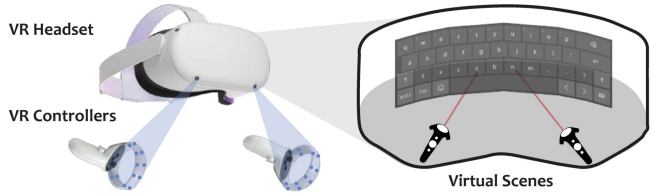
Figure 1 depicts a typical scenario of our proposed novel side-channel attack. The victim wears the VR headset and types the word "HELLO" on the virtual keyboard, while the attacker exploits the IR signals captured by an IR sensor array consisting of multiple IR sensors to capture IR emissions from the VR controllers. The captured signals will then be leveraged to reconstruct corresponding keystrokes for uncovering sensitive user privacy, such as user account passwords. Though it appears to be intuitive and straightforward, this new infrared side-channel attack in the VR platform is non-trivial and contains several overlooked challenges as follows.

- **Challenge ❶: Multiple IR Sources.** Most previous studies have overlooked identifying the active typing source, focusing instead on a simplified single-controller typing scenario. However, typing on both controllers can significantly affect signal traces (*e.g.*, motion sensor data [1]–[4], acoustic signals [10], and network traffic [7]). Therefore, identifying which IR source is typing on the virtual keyboard and reducing non-target interference is crucial.
- **Challenge ❷: Coordinates Calibration.** In practice, the VR user could type the keyboard floating at different orientations in the virtual scenes, which induces variations in the captured IR signals. Therefore, it is crucial to calibrate the coordinates between the virtual keyboard and the IR receivers by leveraging barely the leaked IR signals.
- **Challenge ❸: Scalability Enhancement.** Instead of relying on trace-based DNN models to classify each key for inferring virtual keystrokes [1]–[3], [5]–[10], it is challenging to enhance the scalability to achieve identifying inputs on the virtual keyboard in a straight and model-free perception manner by monitoring the trajectory of IR leakages.

Given that recent research [14] has revealed that most mainstream VR devices (*e.g.*, Meta Oculus Quest, HTC VIVE, and Sony PlayStation VR) have adopted infrared-based constellation tracking systems, the threat posed by this infrared side channel becomes increasingly significant and concerning. Therefore, in this paper, we propose `VRecKey`, a novel keystroke inference attack to validate the feasibility of the newly identified infrared side channel in VR controllers and understand the leakage of keystrokes on virtual keyboards systematically and comprehensively. Specifically, we leverage the captured IR signals to generate heatmaps to monitor and mitigate the image retention to address the issues of multiple IR sources, utilize the IR fluctuation intervals to realize coordinate calibration, and generate the keystroke trajectory on the keyboard to enhance the scalability and practicality.

In the evaluation, we have assessed the effectiveness of `VRecKey` with a customized IR sensor array on the virtual keyboards of two commodity VR devices, Meta Oculus Quest 2 and PICO 4 All-in-One, while typing character-level and word-level keystrokes with different lengths inside the virtual environments. Our evaluation results show that `VRecKey` achieves promising effectiveness in character-level key recognition (T-1 accuracy: 85.8%, T-3 accuracy: 94.2%) and unconstrained word-level keystroke inference (T-1 accuracy 81.7%, T-3 accuracy: 90.5%). Furthermore, `VRecKey` also presents high resilience and transferability when considering several practical impact factors, including different VR devices, orientation angles, cell widths of the customized IR sensor array, attacking distances, typing speeds on the virtual keyboard, and the slight movements of the VR user. We also investigate the realistic influence of user movements, omnidirectional LED distributions, and input with single or both controllers. In addition, we further evaluate `VRecKey` under three real-world scenarios with varying conditions (*e.g.*, low-visibility environment) to show the practicality of `VRecKey`, propose effective countermeasures to defend against this novel side-channel attack, and discuss the ramifications when it comes to the newly-

(a) Infrared LEDs (blue points) on commodity VR controllers.



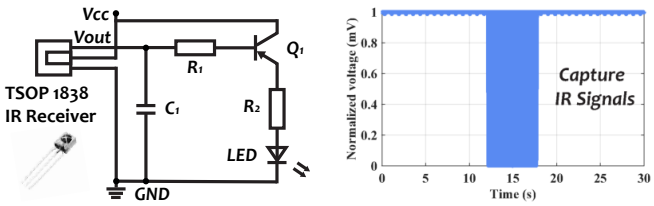
(b) Constellation tracking system.

Fig. 2: Illustration of infrared LEDs embedded in commodity VR controllers (denoted as blue dots on the ring) and the constellation tracking system in VR devices (§ II-A).

released controller-less VR device, the Apple Vision Pro, as well as illustrate the limitations and future works of VRecKey.

Table I illustrates the properties of VRecKey while comparing with ten state-of-the-art keystroke inference attacks in VR devices (*e.g.*, [1]–[10]) qualitatively and quantitatively, where VRecKey presents distinctive advantages of being non-intrusive, model-free, and unconstrained in inferring virtual keystrokes. We further summarize our contributions as follows:

- **Novel Side-channel Attack Vector.** We introduce a new side-channel attack that leverages the IR signals leaked from infrared LED lights embedded in VR controllers to infer unconstrained keystrokes on the virtual keyboard non-intrusively, demonstrating a new attack vector to understand side-channel vulnerabilities in the constellation tracking system of most commercial VR platforms (§ III).
- **Customized Attack Design.** We design and implement VRecKey, a novel attack to demonstrate the feasibility and scalability of the new infrared side channel, which exploits a customized 2D IR sensor array to capture the leaked IR signals from the two VR controllers, utilizes the response interval to calibrate the coordinates and reconstruct both character-level and word-level keyboard input inside the virtual environment, as well as the keystroke trajectory without training trace-based DNN models (§ IV).
- **Comprehensive Evaluations.** We conducted a series of evaluations on two commercial devices, including character-level key recognition and continuous word-level keystroke inference (§ V). Then, we evaluated VRecKey with a set of practical impact factors with consideration of user movements and omnidirectional features of VR controllers in several real-world scenarios (§ V-E). The empirical results show that VRecKey can effectively reconstruct virtual keystrokes with varying lengths non-intrusively.



(a) Circuit of IR sensor. (b) Recorded voltage signal.

Fig. 3: Preliminary of IR sensors, including the circuit to capture IR signals and the recorded voltage signal (§ II-B).

II. Background

A. Infrared LED in VR Controllers

Unlike traditional VR devices such as the HTC VIVE Pro [15], which rely on external base stations for its outside-in tracking approach, many newly-released VR devices, *i.e.*, the Meta Oculus Quest 2 [16], have utilized the constellation tracking system [16], which only consists of a VR headset and two controllers. Each controller incorporates a set of infrared LEDs discretely positioned on the controller's rings. Simultaneously, the VR headset's cameras continuously capture images of these LEDs' emitted IR signals. Then, the constellation tracking system [17] leverages these images to measure the controllers' spatial positions and further infer the user's hand gestures and body movements. Figure 2a shows an illustration of four VR controllers of four commercial VR devices: Meta Oculus Quest 2 [16], PICO 4 All-in-One [18], HP Reverb G2 [19], and PlayStation VR [20], where we observe that the infrared LEDs are evenly distributed around the controller's ring to track the user's interactions accurately. Figure 2b shows the constellation tracking system in these VR devices, and such an IR-based constellation tracking system significantly reduces implementation costs as it eliminates the need for the purchase and setup of external base stations [13]. However, it's important to note that when people utilize these VR controllers for interaction in virtual scenes, the infrared LEDs on the controllers inevitably emit IR signals into the surrounding environment. These signals may potentially contain sensitive information that could be intercepted and analyzed, posing a potential privacy risk, such as snooping input passwords in virtual scenes.

B. Principle of Infrared (IR) Sensors

As discussed above (§ II-A), the IR signals emitted from VR controllers contain sensitive information, which can be captured by IR sensors (*e.g.*, TSOP 1838 Distance Sensor Receiver [21]). In particular, the embedded photodiode in the IR sensor captures the ambient IR signals emitted from VR controllers, and the built-in circuit demodulates the captured signals to extract the baseband signal from the modulated carrier wave. Then, the extracted signals are amplified and filtered with a band-pass filter centered around the modulation frequency (*e.g.*, 38 kHz) to eliminate extraneous noises and enhance signal quality [22]. Then, the IR sensor generates a digital output signal to the connected microcontrollers

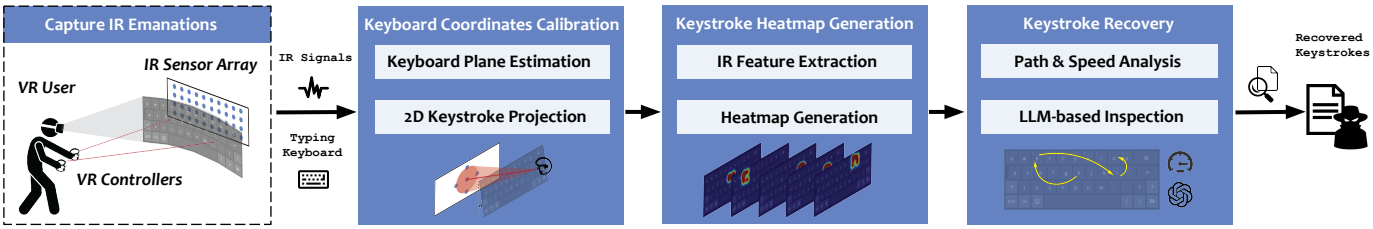


Fig. 4: Overview of VRecKey (§ IV-A).

(MCU), which activate transitions of the output pin to a low state (0 V) when receiving an IR signal and reverts to a high state (V_{cc}) in the absence of IR signals.

Figure 3a presents a typical IR sensor circuit based on the TSOP 1838 IR receiver, which includes the following components: a resistor R_1 to pull up the output of TSOP 1838 to V_{cc} when no signal is received, a capacitor C_1 to filter out noise on the power supply line, a transistor Q_1 to act as a switch for turning on the LED when an IR signal is received, a resistor R_2 to limits the current flowing into the base of Q_1, and an LED light to depict the reception of IR signals. Assuming the base current flowing into the transistor Q_1 is I_B and the current gain factor is h_{FE}, the output voltage of the IR sensor (V_{out}) can be expressed in Equation 1:

$$V_{out} = V_{cc} - (I_C \cdot R_2) = Vcc - (h_{FE} \cdot I_B \cdot R_2) \quad (1)$$

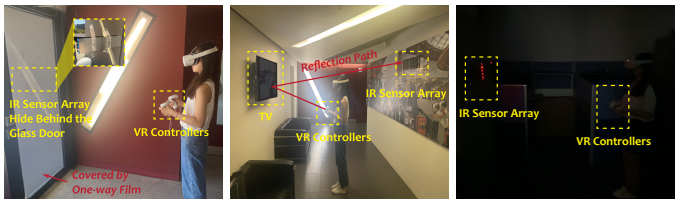
When no IR signal is present, the base of the transistor Q_1 is at a high state, and there are no base current flows ($I_B = 0$), which makes the output voltage close to V_{cc}.

$$V_{out} = V_{cc}, I_B = I_C = 0 \quad (2)$$

On the contrary, when an IR signal is received, the output of the TSOP 1838 goes low, which then allows a base current to flow into transistor Q_1 through the resistor R_2, which also turns on transistor Q_1 to allow current to flow from the collector to the emitter side to the LED. As a results, as shown in Equation 3, the V_{out} is equal to the saturation voltage ($V_{Q(sat)}$) of the transistor Q_1, which is typically a small value close to 0 V as Q_1 is fully on.

$$V_{out} = V_{Q(sat)} \approx 0, V_{cc} \approx I_C \cdot R_2 \quad (3)$$

Therefore, it is feasible to exploit the output voltage (V_{out}) of an IR sensor to monitor the presence of IR signals radiated from the VR controllers. For instance, Figure 3b shows the recorded V_{out} of the IR sensor in capturing IR signals, where we find the corresponding voltage fluctuations when detecting the presence of IR signals emitted from a VR controller of Meta Oculus Quest 2. Specifically, it records both the spatial (*e.g.*, position) and temporal (*e.g.*, time) information of the VR controller. Hence, if multiple IR sensors are at a 2D plane (*e.g.*, IR sensor array), it is feasible to reconstruct the whole trajectory of the VR controller and further infer users' private keystrokes inside the virtual environment, such as entering passwords in the virtual scenes.



(a) Concealed. (b) Reflection-based. (c) Low-visibility.

Fig. 5: Three real-world attack scenarios (§ III), including a concealed attack with one-way film, a reflection-based attack, and an attack in a low-visibility environment.

Note that IR signals emitted from VR controllers operate at a specific modulation frequency (*e.g.*, 38 kHz) that can be captured by the headset's cameras and external IR sensors (*e.g.*, TSOP 1838 IR receivers). As a result, thermal IR radiations from other objects in the environment (*e.g.*, the human body) cannot interfere with the IR receivers because these IR signals transmit at an unmodulated frequency and present relatively low compared to the strong, focused, and modulated IR signals from the embedded LEDs on VR controllers.

III. Threat Model

Attack Scenario. We consider a common scenario when the victim wears the VR headset and holds the two controllers to type virtual keystrokes inside a virtual environment. Following the research line of deploying attacking devices near VR users (*e.g.*, [5], [6], [9], [10], [12]), we assume that the attacker can place an IR sensor array (*e.g.*, 4×10) consisting of multiple IR sensors near the victim at a long distance (*e.g.*, 2.0 m–4.0 m), and remotely analyze the captured IR signals emitted from VR controllers to uncover sensitive keystrokes (*e.g.*, passwords) on the virtual keyboard in an unconstrained manner. Specifically, the IR sensor array can be stealthily placed either in front of the victim, hidden by a one-way film [23] for concealment, or behind the victim to capture reflected IR signals for keystroke inference, which requires no LoS view to infer virtual keystrokes. Such a scenario is prevalent in daily life in various indoor spaces, such as homes and offices, and is plausible for three reasons: *(i)* most controllers of commercial VR devices use infrared-based inside-out tracking systems to realize user interactions with virtual scenes, which inevitably emits IR signals to the surrounding space, *(ii)* a small IR sensor array positioned or concealed at a considerable distance in front rather than on the VR device's side, is less likely to draw the attention of the victim immersed in the virtual environment, especially in low-visibility environments

like a dark room, and *(iii)* a common room has TVs or glass walls, may reflect IR signals, enabling an IR sensor array positioned behind the user to capture these reflections and potentially leak keystroke without LoS view, as demonstrated in relevant studies [24]. Figure 5 illustrates three real-world attack scenarios: a concealed attack, a reflection-based attack, and an attack in a low-visibility environment, respectively. Note that our attack scope primarily targets controller-based VR devices, which constitute the majority of commodity VR products, excluding controller-less VR devices, *i.e.*, Apple Vision Pro [25], which is unavailable as of this writing.

Attacker's Capability. Unlike previous VR-related attacks for snooping virtual keyboard input (*e.g.*, [1]–[4], [11]), we do not assume the attacker can compromise the VR headset to install malicious software or apps for accessing motion sensor data, nor directly obtain the input keystrokes. We also do not assume the victim would join unauthorized online meetings to share hands and gaze movements through the virtual avatar [7], [8]. Moreover, we assume the attacker cannot directly monitor the victim's head movements or hand gestures (*e.g.*, placing multiple cameras to record videos [9], [12]) as this would easily arouse the victim's suspicion. In addition, the victim can type the virtual keyboard displayed in the VR headset while sitting at a stationary boundary or standing within the playing zone with natural and casual movements, instead of requiring the victim to sit at constrained positions between the transceivers in other side-channel attacks [5], [10].

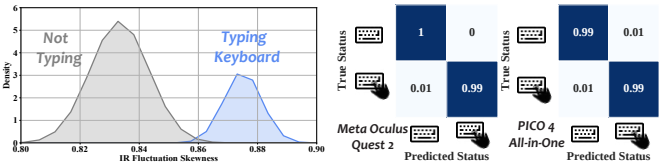
IV. Attack Design

A. Overview of `VRecKey`

Figure 4 shows the overview of `VRecKey`. An attacker first places a 2D IR sensor array near the victim, who wears the VR headset and holds the controllers to type on the virtual keyboard. When the victim types different keystrokes, the VR controllers emit IR signals to interact with the virtual scenes, which also leaked to be captured by the multiple IR sensors on the array (§ IV-B). First, `VRecKey` leverages the IR signals from the IR sensors to estimate the orientation between the virtual keyboard and the 2D IR sensor array to calibrate the keyboard coordinates and project the 2D keystrokes (§ IV-C). Then, `VRecKey` extracts time-domain and frequency-domain IR features from the IR signals and generates heatmap overlay onto the virtual keyboard (§ IV-D). Finally, the attacker can infer unconstrained keystrokes by reconstructing the trajectory of the typed keystrokes, estimating the typing speed, and applying an LLM-based keystroke autocorrection tool to enhance the semantics and grammar of the inferred keystrokes (§ IV-E).

B. Capture IR Emanations and Identify Typing Event

Customized IR Sensor Array. To capture the IR signals emitted from the infrared LEDs on the VR controllers, we have designed and implemented a customized 2D IR sensor array (Figure 25 in Appendix). It consists of three main components: *(i)* 40 IR sensors to capture the emitted IR signals and convert them to measurable voltage signals, *(ii)* five Arduino Nano microcontrollers (MCUs) to control the IR sensors, and



(a) IR Skewness distribution. (b) Identifying Results.

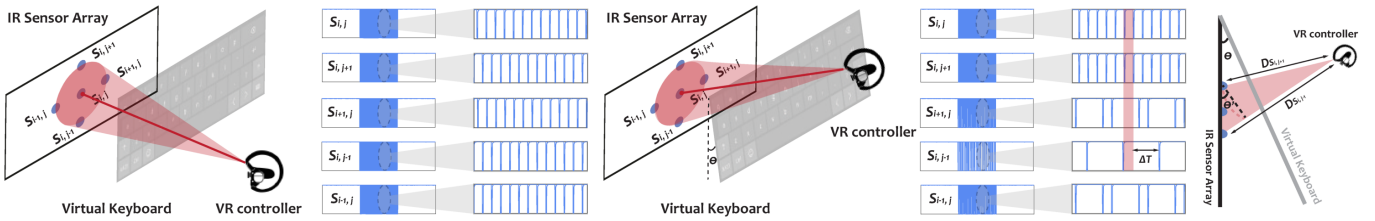
Fig. 6: Identify typing event by IR signal skewness (§ IV-B).

(iii) five microSD card adapters to record the measurable voltage data from the IR sensors. We have integrated these components together on a custom-built PCB board with a size of 23.5 in × 6.3 in (59.7 cm×16.0 cm). Specifically, we utilize 40 KEYES 1838T infrared sensor receiver module boards [26] as the IR sensors, and each sensor presents a small size of 1.1 in × 0.9 in× 0.3 in (2.8 cm× 2.3 cm× 0.8 cm). We chose this IR sensor because it can receive the 38 kHz remote IR signals (*i.e.*, typically 15 m claimed in the datasheet [27]) through the control of Arduino Nano MCUs and then decode the captured IR signals to be voltage output, and it also shows a promising capability of resisting electromagnetic interference and light. Furthermore, the default distance between two adjacent IR sensors (*a.k.a.*, cell width) is set to 5 cm to prevent interference from each other. Finally, the five Arduino Nano MCUs record all the 40 voltage output from the 40 IR sensors at a sampling frequency of 20 kHz, and then store the data into the five 32 GB microSD cards embedded in the card adapters. Note that the total cost of building this 2D IR sensor array prototype is approximately 120 dollars.

Identify Typing Event. After capturing the leakage of IR emanations, we analyze the signals to detect typing events on the virtual keyboard by focusing on the fluctuation patterns within the IR data. Specifically, when the captured IR signals contain no typing information, the fluctuations are evenly distributed. On the contrary, when typing events occur (*e.g.*, button presses on the controller), the IR signals exhibit irregular fluctuations due to the modulation of addresses and commands. As demonstrated in Figure 6a, we calculate the density distribution of skewness in the fluctuating IR signals to distinguish between typing events and static states. Figure 6b further shows that this method achieves 99.3% accuracy in identifying typing events from non-typing statuses on two commercial VR headsets (*i.e.*, Meta Oculus Quest 2 and PICO 4 All-in-One).

C. Keyboard Coordinates Calibration

Unlike physical or soft keyboards on other mobile devices (*e.g.*, smartphones, tablets), virtual keyboards are typically displayed in front of the VR user within the virtual environment during a typing process. Hence, due to the unparalleled orientation between the keyboard plane and the 2D IR sensor array, the captured IR signals could be distorted. Therefore, it becomes imperative to conduct keyboard coordinates calibration to calculate the orientation angles and recalibrate the coordinates to align with the keyboard plane. Below, we design and implement a two-step process for keyboard



(a) Virtual keyboard is parallel to 2D IR sensor array. (b) Virtual keyboard is not parallel to 2D IR sensor array. (c) Illustration.

Fig. 7: Keyboard Coordinates Calibration. We consider whether the virtual keyboard is parallel to the 2D IR sensor array (a) or not (b), and present an illustration of how to measure the orientation angle between the two planes (c) (§ IV-C).

coordinates calibration in `VRecKey`, which encompasses keyboard plane estimation and 2D keystroke projection.

Keyboard Plane Estimation. As demonstrated in [6], virtual keystrokes are aligned on the same plane as virtual keyboards, with the study utilizing a regression model to obtain the keyboard plane based on 3D keystroke data. Drawing inspiration from this work, we propose a model-free method for keyboard plane estimation, leveraging the variations in response times of IR signals captured by multiple sensors to estimate the fitting keyboard plane. Assuming the orientation angle between the keyboard plane and the 2D IR sensor array as θ and the grid width is l, we select a specific IR sensor $s_{i,j}$ with four of the adjacent IR sensors (*i.e.*, top: $s_{i,j+1}$, bottom: $s_{i,j-1}$, left: $s_{i-1,j}$, and right: $s_{i+1,j}$) and analyze their captured IR signals to estimate the keyboard plane as shown in Figure 7a. Specifically, if the two planes are parallel ($\theta = 0°$), the four adjacent IR sensors present similar patterns and fluctuations when capturing IR signals as they are evenly distributed on the circumference of the IR emanation frustum. Meanwhile, due to the closer distance, IR signals captured by $s_{i,j}$ present tight intervals and more drastic fluctuation than the other four sensors. On the contrary, Figure 7b shows the condition when the two planes are not parallel ($\theta \neq 0°$), and the captured IR signals from $s_{i,j+1}$, $s_{i,j}$, and $s_{i,j-1}$ presenting different intervals. Specifically, we denote the interval differences between $s_{i,j-1}$ and $s_{i,j+1}$ as ΔT and the path length are $D_{s_{i,j-1}}$ and $D_{s_{i,j+1}}$, and obtain the length difference ΔD as follows:

$$\Delta D = n_{IR}(D_{s_{i,j-1}} - D_{s_{i,j+1}}) \propto 1/\Delta T, \tag{4}$$

where we can measure the distances $D_{s_{i,j-1}}$ and $D_{s_{i,j+1}}$ by moving the VR controller from the IR sensor array's top to the bottom [28] to obtain the ΔD in a time interval of the sensor response frequency f_{IR} (*e.g.*, 38 kHz), and it contains n_{IR} times of fluctuations in an IR transmission. Since the distance between the VR controller and the 2D IR sensor array (*e.g.*, 3.0 m) is usually much larger than the grid width l (*e.g.*, 5.0 cm), the angle $\theta^{'}$ is close to the orientation angle θ, and Figure 7c shows an illustration of deriving the value of θ as:

$$\theta \approx \theta^{'} = arcsin(\frac{\Delta D}{2 \cdot l \cdot n_{IR}}), n_{IR} \propto f_{IR}. \tag{5}$$

After we obtain the orientation angle θ, we estimate the coordinates of the keyboard plane relative to the 2D IR sensor array plane, and then project the input keystrokes onto the keyboard plane for coordinate calibration.

Fig. 8: Keystroke heatmap generation when the VR user types the word "HELLO" on the virtual keyboard. Upper part: confusion matrices of extracted IR features. Lower part: generated heatmaps on the virtual keyboard (§ IV-D).

2D Keystroke Projection. Based on the obtained orientation angle θ between the virtual keyboard and the 2D IR sensor array, we then project the keystrokes detected from the IR sensor array to the virtual keyboard to achieve the calibration of keyboard coordinates. In practice, we assume the normal vector on the virtual keyboard plane is $N = (A, B, C)$ and the point $P_0 = (x_0, y_0, z_0)$ is on the virtual keyboard, and the virtual keyboard plane can be written as:

$$A(x - x_0) + B(y - y_0) + C(z - z_0) = 0 \tag{6}$$

Furthermore, let $N_{IR} = (A_{IR}, B_{IR}, C_{IR})$ as the normal vector and $P_{IR} = (x_{IR}, y_{IR}, z_{IR})$ be the point of the IR sensor array, the IR sensor array plane is shown Equation 7:

$$\begin{cases} A_{IR}(x - x_{IR}) + B_{IR}(y - y_{IR}) + C_{IR}(z - z_{IR}) = 0 \\ cos(\theta) = \frac{N \cdot N_{IR}}{\|N\| \|N_{IR}\|}, \ \theta \text{ is the orientation angle.} \end{cases} \tag{7}$$

Therefore, the projection of point P_{IR} on the virtual keyboard plane is the point $P_p = (x_p, y_p, z_p)$, where the vector $v = \overrightarrow{P_{IR}P_0}$ is the sum of a vector parallel to the virtual keyboard plane and a vector perpendicular to this plane, where v can be expressed in Equation 8:

$$v = \overrightarrow{P_{IR}P_0} = (x_0 - x_{IR}, y_0 - y_{IR}, z_0 - z_{IR}) \tag{8}$$

Finally, we can project the coordinates of the obtained keystrokes from the IR sensor array plane to the virtual keyboard plane by the following Equation 9:

$$\overrightarrow{P_{IR}P_0} = \overrightarrow{P_pP_{IR}} + G(A, B, C) \tag{9}$$

where G is the scalar from the IR sensor array plane to the virtual keyboard plan. We could substitute these coordinates into the Equation 7 and solve the specific value of G and

further acquire the projected coordinates (x_p, y_p, z_p). Note that we only consider orientation angles in common conditions ($\theta < 90°$) while assuming the pointing vector of VR controllers cannot be paralleled to the keyboard plane.

D. Keystroke Heatmap Generation

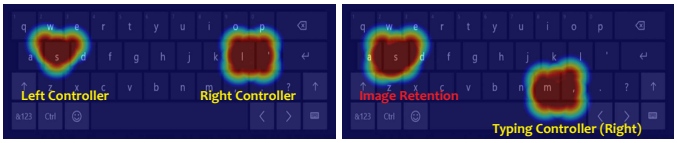
After the calibration of the virtual keyboard coordinates, `VRecKey` then records the captured IR signals from every IR sensor node on the sensor array simultaneously and generates the heatmap for inferring keystrokes. Specifically, `VRecKey` first creates IR feature maps by extracting time-domain and frequency-domain features from the captured IR signals, and then generates corresponding heatmaps to demonstrate the trajectory of the VR controller, which can be further exploited for inferring the user's keystrokes inside the virtual environment.

IR Feature Extraction. As mentioned above, the converted voltage signal of a commercial IR sensor fluctuates between $0\,\mathrm{V}$ and V_{cc} (*e.g.*, $5\,\mathrm{V}$) when capturing the emitted IR signals, which also reflects temporal (*e.g.*, lasting time, fluctuation interval) and spatial (*e.g.*, position) IR-related features. Specifically, the voltage variance remains below 0.05 when no IR signals are captured and it exceeds 0.90 when capturing IR signals. Hence, in `VRecKey`, we first apply a moving-variance window with a threshold of 0.1 to select the informative signal segment from the raw IR signal, as well as the timestamps indicating the starting time and the end time of the IR emanations. Then, we normalize the amplitude of IR signals to $[0, 1]$ to mitigate the impact of varying strengths of different VR controllers. In particular, we extract six time-domain features, including the starting timestamp, duration time, peaks, troughs, mean, and variance. Furthermore, we then utilize the *Fast Fourier Transform* (FFT) to the IR signal segment and extract the mean of frequency components, the power spectral density, and the entropy. These features describe *(i)* position on the virtual keyboard the VR controller points at, *(ii)* duration time when pointing the specific key to represent the potential continuous same keys in many words, *i.e.*, the word "HELLO" needs to press the key "L" twice successively, and *(iii)* the trajectory of the VR controller across the virtual keyboard at varying timestamps.

Heatmap Generation. Subsequently, to visualize the IR features and exhibit the keystroke, `VRecKey` leverages the IR features to generate a heatmap on the IR sensor array to determine the keystrokes. From the above discussion, we know that each feature map corresponds to a specific key-typing event on the virtual keyboard. That is, we can map the extracted IR features to the coordinates (*e.g.*, (x_i, y_i)) of a specific IR sensor (*e.g.*, $s_{i,j}$) on the IR sensor array. Let $\mathcal{M}$ represent the mapping function between the IR features and the corresponding keystroke-related heatmap, where $\mathcal{M}$ can be expressed as follows (Equation 10):

$$(x_i, y_i) = \mathcal{M}(\sum \alpha_i f_{t_i} + \sum \beta_j f_{f_j}) \tag{10}$$

where f_{t_i}, f_{f_j} represent the ith time-domain feature and the jth frequency-domain feature, respectively. The parameters



(a) Heatmap at time t. (b) Heatmap at time $t + \Delta t$.

Fig. 9: Image retention remove in a time interval Δt (§ IV-D).

α_i and β_j are the coefficients of the features to balance the weights across diverse dimensions. Specifically, we leverage Multiple Linear Regression (MLR) [29] to find coefficients with the closest distance to the IR sensor position (x_i, y_i). Next, since the IR signals emitted from a VR controller can be captured by other nearby IR sensors, `VRecKey` exploits the output coefficients to ascertain the weights of each IR sensor on the array. These weights normalize them to a range from 0 to 1 to facilitate the generation of a heatmap on a 4×10 colored matrix. In the final stage, `VRecKey` combines the colored matrices generated in the one-second time interval of a keyboard-typing (*e.g.*, typically $0.1\,\mathrm{s}$–$1.5\,\mathrm{s}$ [30]), and utilizes the *applyColorMap* method in Python OpenCV package [31] to transition the colored matrix into a corresponding heatmap. Such a heatmap is overlaid on the virtual keyboard and provides a visual representation of keyboard typing events, effectively revealing the trajectory of the keystroke input from the VR controllers.

Figure 8 shows an example of confusion matrices of IR features and the generated heatmap from `VRecKey` when the VR user types the word "HELLO" on the virtual keyboard in the commercial VR headset, Meta Oculus Quest 2. Specifically, we set the threshold at 0.8 to filter out low-value noise while maintaining the spot with the largest probability, and generate the heatmap with clear spots to reflect the typed keys on the virtual keyboard. By exploiting these heatmaps, `VRecKey` achieves unconstrained keystroke inference while not relying on training machine-learning-based models for classification, which is widely used in other keystroke inference works (*i.e.*, [1], [2], [4], [5], [32], [33]). Note that the threshold is determined by the IR signal's strength captured at specific key positions and the key sizes on the generated heatmap. It can be transferred to different VR devices by mapping the heatmap to the layout of the virtual keyboard.

Image Retention Remove. Despite most people prefer to use only one controller to type on the virtual keyboard, the VR user could enter keystrokes through VR controllers on both left and right hands in a realistic scenario while the IR sensor array may capture IR leakage from multiple IR sources. As a consequence, the generated heatmaps could contain image retention from the VR controller that does not type on the virtual keyboard. Therefore, we select the two key positions with the highest strength values on the heatmap and monitor their movements in three typing intervals, and then remove the image retentions from the heatmap once we detect the dynamic source, *i.e.*, the typing VR controller. Figure 9a and Figure 9b individually show the heatmaps of detecting the image retention in a typing interval when the typing VR

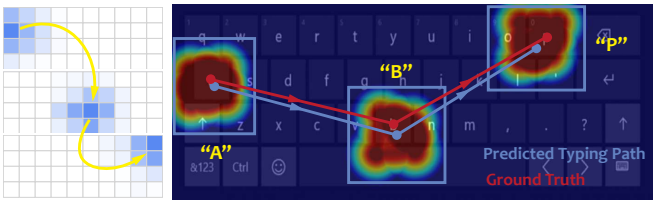


Fig. 10: Example of typing path analysis when typing keys "A→B→P" continuously on the virtual keyboard (§ IV-E).

controller (right hand) moves from key "L" to key "M", and we can determine the left spot area is the image retention from another untapped VR controller (left hand).

E. Unconstrained Keystroke Inference

By harnessing the generated heatmaps from the IR signals, VRecKey has the ability to identify each individual typing key on the virtual keyboard accurately. To achieve unconstrained keystroke inference, it is crucial to reconstruct the typing trajectory between consecutive keys. Therefore, VRecKey utilizes the bounding box center method to estimate the potential typing trajectory on the virtual keyboard, and estimates the VR user's typing speed on the virtual keyboard while distinguishing consecutive identical characters. Subsequently, it leverages the capabilities of state-of-the-art large language models (LLMs), *i.e.*, ChatGPT, to automatically check the format of reconstructed keystrokes and address semantic and grammatical errors.

Typing Path Analysis. So far, VRecKey has exhibited the ability to recover individual keystrokes by utilizing captured IR signals. Typically, a VR user holds the controllers to input keystrokes in a continuous sequence. This sequence encapsulates not only singular key-pressing events but also the trajectory of the typed keys, revealing the order of the keys in a typed word. In practice, while typing continuously on the virtual keyboard, the VR controller inevitably sweeps across keys, which triggers an immediate response from the IR sensors along the path that could be falsely recognized as a key-typing event. Therefore, to mitigate the interference of IR sensors and determine the typing path, VRecKey selects the spots on the heatmaps with values higher than the threshold 0.8 while filtering out interference from low-response (< 0.8) IR signals (*e.g.*, green and blue zones), and then exploits the bounding box center method [34] in MATLAB to determine the center from each irregular spot. Upon acquiring the centers of each bounding box, we connect them together to infer the approximate typing path on the virtual keyboard. For example, Figure 10 illustrates both the ground truth and VRecKey's predictions concerning the typing path as the VR user inputs "A→B→P" sequentially, which depicts similar keystroke trajectories with little deviations.

Typing Speed Estimation. In practice, when it comes to typing words that contain consecutive identical characters, the reconstructed keystroke trajectory cannot reflect the whole words, *i.e.*, VRecKey outputs the same trajectories when typing words like "BEE" and "BE", "OFF" and "OF" on the virtual keyboard. Meanwhile, a recent keystroke study has demonstrated that the typing speed could impact the performance of side-channel keystroke inference [30]. Therefore, it is necessary to estimate the typing speed to segment these double-typed characters and reduce its impact on VRecKey's performance in unconstrained keystroke inference. Figure 11 shows the interval time distribution of five different VR users when typing the required keys on the virtual keyboard of Meta Oculus Quest 2 for 100 times, where we find that the time interval of typing a key ranges approximately from $0.3\,\mathrm{s}$ to $2.8\,\mathrm{s}$ while the response time of the virtual keyboard ranges from $0.1\,\mathrm{s}$ to $0.3\,\mathrm{s}$. In practice, to estimate the user's typing speed, we define the typing speed at three levels of typing speed: fast (typing interval $< 0.5\,\mathrm{s}$), medium (typing interval between $0.5\,\mathrm{s}$ and $2.0\,\mathrm{s}$), and slow (typing interval $> 2.0\,\mathrm{s}$) , and we collect data samples from all 25 participants to alleviate user variations. As we have set the time interval of heatmap generation as one second (§ IV-D), if the estimated typing speed falls into the slow level ($> 2.0\,\mathrm{s}$), it is possible that the two subsequently generated heatmaps depict the same key on the virtual keyboard. Once detected, we concatenate the two same individual keys together and regenerate the output (*e.g.*, "BE" to "BEE") to prevent the double-typing cases that present similar keystroke trajectories, and recover the reasonable keystrokes.

LLM-based Keystroke Inspection. Upon obtaining the recovered keys and keystroke trajectory, VRecKey generates the outputs of the keystroke typed on the virtual keyboard. Given that adjacent keys on the virtual keyboard are in close proximity, misclassified cases exist in recognizing adjacent keys, potentially increasing the incidence of false recognition and impacting VRecKey's accuracy. To mitigate this, we integrate a large language model (LLM) for two purposes: *(i)* check whether the recovered keystrokes follow a password format, and *(ii)* perform semantic and grammatical refinement of the output word. Specifically, we design a zero-shot LLM-enhanced keystroke inspection tool based on ChatGPT [35] to check keystroke format and correct the predicted words automatically, which is implemented by entering the prompt "Check if this keystroke <*recovered keystrokes*> follows a typical password format, otherwise, perform spelling and grammatical check, and then generate top-3 candidates." which aims to generate the three most probable alternatives for the keystrokes identified. Figure 12 shows the results of an ablation study in VRecKey's performance in word-level keystroke inference when applying the LLM inspection tool or not, where it increases the T-1 and T-3 accuracy by 5.4% and 3.3%, respectively. Additionally, other online LLMs can also be exploited, *i.e.*, GPT-4 [36] and Google Gemini [37].

V. Evaluation

A. Experimental Methodology

Experiment Setup. We conduct experiments for the evaluation of VRecKey's performance using two commodity VR devices: Meta Oculus Quest 2 and PICO 4 All-in-One, which both adopt the infrared tracking mechanism between the

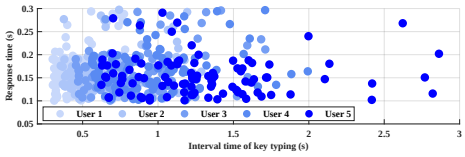


Fig. 11: Interval time distribution of five VR users in typing the virtual keyboard.

Fig. 12: Ablation study results of applying the LLM-based keystroke inspection.

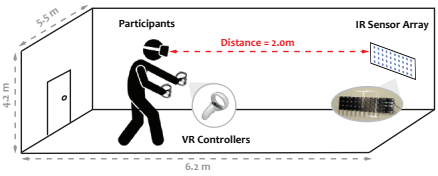


Fig. 13: Default experiment setup.

VR headset and the controllers. In practice, Figure 13 shows the default experiment setup, where we ask the participants to wear the headset and use the controllers to type the required keys. Specifically, we place the customized IR sensor array (§ IV-B) in front of the VR headset at a primary distance of 2.0 m and collect the captured IR signals from all IR sensors simultaneously. In § V-D, we conduct further experiments in the three real-world scenarios (Figure 5) with different settings to further demonstrate VRecKey's practicality. The sampling rate of the five Arduino Nano MCUs is set to 200 kHz, and the recorded IR sensors' voltage signals are stored in five 32 GB SD cards. Finally, the collected data samples are processed on a desktop remotely.

Participants. We recruited 25 university students and staffs (15 males and 10 females, with ages ranging from 18 to 35) for the data collection in this study[1]. Only 11 participants have prior VR experience, and the other 14 participants have no knowledge of using VR devices. We ask the participants to type on the virtual keyboards in VR devices for 30 minutes to get familiar with VR typing before the official data collection. All participants were informed that the infrared signals from the VR controllers would be recorded to infer their keystrokes. During the experiments, participants could move slightly when standing before the IR sensor array while playing the VR device naturally, which aligns with the experimental settings with most prior studies (*e.g.*, [1]–[6], [9]–[11]). According to our institution's IRB approval, each participant must sign a written consent form that allows us to collect data from human behaviors for evaluation.

Evaluation Metrics. To evaluate VRecKey's effectiveness, we select *accuracy* as the metric for character-level keystroke recognition, which is defined as the ratio of keystrokes correctly identified as k to the total occurrences of k. In respect of evaluating VRecKey's performance in word-level keystroke inference, we select the *top-3 (T-3) accuracy* as the metric since the proposed LLM-based keystroke autocorrection algorithm generates a number of potential candidates after checking the spelling and grammar. This metric reflects the likelihood of the correct keystroke being present within the first three candidates containing the keystrokes input by the VR user.

[1]**Ethical Considerations and Open Science Compliance:** We take ethical considerations seriously. This study has obtained the IRB approval from our institution for data collection, and we only use our accounts of Meta Oculus Quest and PICO platforms to type their default virtual keyboards. VRecKey and our customized IR sensor array have never been released to any other parties. More details (*e.g.*, code, dataset, demo), updates, and appendices will be released on the project website: https://vreckey.github.io/.

B. Effectiveness of Character-level Inference

Data Collection. To evaluate the performance of VRecKey in inferring individual typing keys, we collect data samples of the IR signals from the 2D sensor array while typing 31 keys (*i.e.*, 26 alphabets, space, shift, comma, dot, and enter) through the VR controllers repeatedly. In practice, each participant presses each key for 100 times on the default virtual keyboards in the two commercial VR devices, Meta Oculus Quest 2 and PICO 4. In total, we collect 124,000 IR signals from the 40 IR sensors on the 2D IR sensor array when we perform 3,100 single key-typing actions. Furthermore, the collected IR signals can be utilized to generate 3,100 IR feature maps and corresponding heatmaps overlay onto the virtual keyboard. Finally, the inferred keys are compared with the ground truth labels to evaluate VRecKey in character-level key inference.

Character-level Key Inference Results. Figure 14 shows the effectiveness of VRecKey in recognizing the 31 single keys on the virtual keyboard, where it achieves averagely 85.8% T-1 accuracy and 94.2% T-3 accuracy. In particular, we found that VRecKey demonstrates the highest level of accuracy when identifying keys positioned along the border of the virtual keyboard, *i.e.*, alphabetic keys like "Q" (T-1: 95%, T-3: 100%), "P" (T-1: 96%, T-3: 100%), "A" (T-1: 96%, T-3: 100%), and special keys including Enter (T-1: 93%, T-3: 100%) and Shift (T-1: 92%, T-3: 100%). Furthermore, since the Space key occupies a larger layout than normal alphabetic keys, VRecKey can recognize it with a higher performance. On the contrary, the internal keys of the virtual keyboard are more susceptible to being misidentified due to the simultaneous capture of IR signals by adjacent IR sensors. That is, a biased typing event on the virtual keyboard could cause these misidentified cases, making VRecKey present relatively lower performance, *i.e.*, "G" (T-1: 85%, T-3: 97%), "H" (T-1: 83%, T-3: 96%), and "J" (T-1: 80%, T-3: 93%). Despite this, VRecKey still achieves high T-3 accuracy in both border and internal keys, which depicts its promising performance in the character-level key inference on the virtual keyboard inside the VR headset.

C. Effectiveness of Word-level Inference

Data Collection. We further evaluate the proposed attack framework, VRecKey, in recovering the word-level keystrokes entered by the VR user under more practical attack scenarios (*e.g.*, accounts, passwords). In practice, we generate alphabetic sequences of high-frequency words ranging in length from one to fifteen, and for each length, we randomly select 100 words from the Cambridge English vocabulary list [38], respectively. Then, we ask each participant to type each sequence on the virtual keyboard of the VR headset

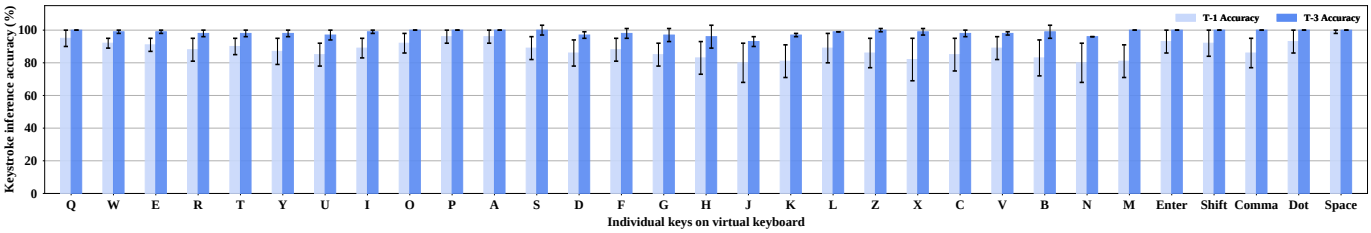


Fig. 14: Evaluation results of VRecKey in character-level keystroke inference, including 26 alphabetic and 5 special keys.

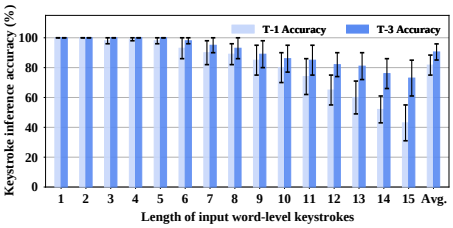


Fig. 15: Evaluation results of VRecKey in word-level keystroke inference.

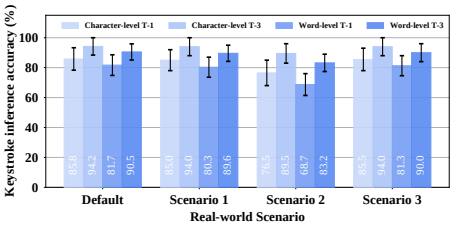


Fig. 16: Evaluation results in the default setting and three real-world scenarios.

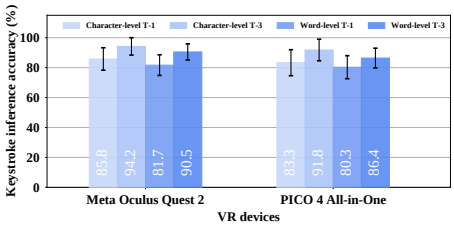


Fig. 17: Evaluation results of two different commercial controller-based VR devices.

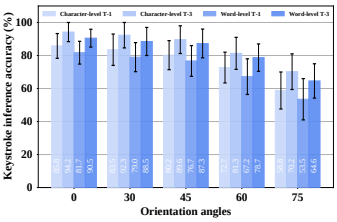


Fig. 18: Evaluation results of three orientation angles.

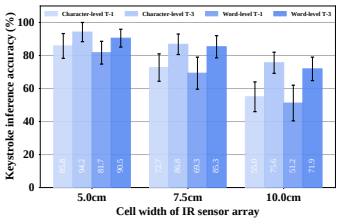


Fig. 19: Evaluation results of three different cell widths.

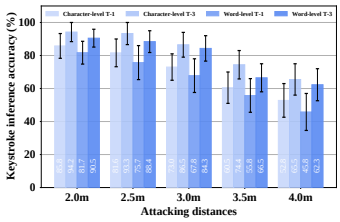


Fig. 20: Evaluation results of five different distances.

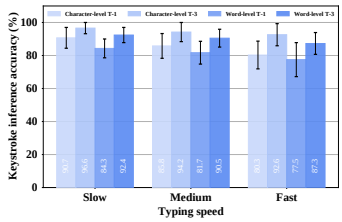


Fig. 21: Evaluation results of three typing speeds.

Meta Oculus Quest 2 and collect corresponding 40 IR signals from the 2D IR sensor array, where this process is repeated ten times. In total, for each user, we collect 600,000 IR signals from the 2D sensor array, extract 15,000 IR feature maps, generate 15,000 heatmaps for word-level keystroke inference, and obtain 1,500 predicted typing path for analysis. Finally, the recovered keystroke candidates are compared with the ground truth labels to obtain the T-1 and T-3 accuracy to evaluate VRecKey in word-level keystroke inference.

Word-level Keystroke Inference Results. Figure 15 shows the effectiveness of VRecKey in recovering word-level keystrokes on the virtual keyboard with lengths ranging from one to 15, where it achieves an overall T-1 accuracy of 81.7% and T-3 accuracy of 90.5%. Specifically, for keystrokes with lengths less than five, VRecKey achieves 99% T-1 accuracy and 100% T-3 accuracy in inferring these words, whereas its performance degrades drastically when the length of testing keystrokes exceeds ten. For instance, VRecKey achieves only 43% T-1 accuracy and 73% T-3 accuracy when recovering keystrokes with the length of 15. To gain a comprehensive understanding of these misidentified cases, our investigation unveiled two primary factors contributing to these occurrences: *(i)* increasing the length of testing keystrokes also leads to more complicated typing paths, which amplifies the possibility of misidentified keys within the keystrokes, and *(ii)* the LLM-based inspection module occasionally generates incorrect words that closely resemble the intended input, especially when the input keystroke deviates from the correct word. Nevertheless, it is important to note that VRecKey continues to demonstrate competitive performance in word-level keystroke inference under unconstrained conditions when comparing to prior research works (*e.g.*, [1]–[10]).

D. Real-world Attack Scenarios

To validate the stealthiness and practicality of our virtual keystroke inference attack in real-world scenarios, we follow the same procedure and collect data samples from the three settings (§ III) to evaluate VRecKey's end-to-end performance, including a concealed attack scenario, a reflection-based attack scenario, and an attack in a low-visibility scenario. Specifically, we leverage the Meta Oculus Quest 2 and require all participants to use controllers to type on the default full-size keyboards to enter both character-level and word-level keystrokes in the virtual scene.

Scenario ❶: Concealed Attack. We first conduct the experiments by leveraging commercial one-way film [23] to conceal the IR sensor array behind a 1.1 m×2.8 m floor-to-ceiling window (Figure 5a) and ask the participant typing the virtual keyboard at a distance of 2.0 m. This film allows IR signals to pass through while significantly reducing the visibility of the sensor array, which helps to mitigate the victim's suspicion. Figure 16 demonstrates that the application of the one-way film has minimal impact on VRecKey's keystroke inference performance, which maintains competitive

accuracy rates for both character-level (T-1: 85.0%, T-3: 94.0%) and word-level (T-1: 80.3%, T-3: 89.6%) inference. These results highlight the effectiveness of VRecKey in executing stealthy attacks while concealing the IR sensor array, confirming its potential for unobtrusive surveillance.

Scenario ❷: Reflection-based Attack. We then evaluated VRecKey's performance in a reflection-based attack scenario, where the IR sensor array is positioned 1.5 m behind the VR user standing in front of a reflective surface like a TV or glass wall with proximity of 1.5 m (Figure 5b), and we receive the reflected IR signals to infer keystrokes. This reflection-based setting is reasonable, which aligns with a previous study [24] and requires only flipping the generated heatmaps vertically to retrieve the correct keystroke trajectory. The results shown in Figure 16 indicate a degradation of approximately 4.7%–9.3% and 7.3%–13.0% in character-level (T-1: 76.5%, T-3: 89.5%) and word-level (T-1: 68.7%, T-3: 83.2%) keystroke inference, respectively. This decrease is because of the signal attenuation during reflection, which leads to an increase in recognition errors from the heatmaps. Nonetheless, the reflection-based attack demonstrates VRecKey's capability in a non-line-of-sight (NLoS) scenario, significantly enhancing the stealthiness of the attack.

Scenario ❸: Low-visibility Attack. In addition, we also evaluated VRecKey in a real-world low-visibility scenario (typically light intensity < 0.1 lux), such as when users interact with virtual keyboards in a dark room or at midnight (Figure 5c) with the default distance settings (2.0 m). Under these conditions, traditional camera-based VR keystroke inference attacks (*e.g.*, [6], [9], [12]) would fail, and enabling the see-through mode in the VR headset would not be feasible. In contrast, our IR sensor array can still capture leaked IR signals to accurately reconstruct virtual keystrokes, achieving high accuracy in both character-level (T-1: 85.5%, T-3: 94.0%) and word-level (T-1: 81.3%, T-3: 90.0%) inference, even in very low-visibility conditions. This newly identified infrared side-channel attack significantly extends the threat model of existing keystroke inference attacks on VR platforms, proving highly effective across various scenarios.

E. Practical Impact Factors

Different VR Devices. Since different VR devices support various user interfaces, the default virtual keyboards in different commercial VR devices present alternative layouts. On the other hand, most of these virtual keyboards adopt a layout similar to full-size QWERTY keyboards that are widely used in other mobile devices (*e.g.*, smartphones and tablets). These commonalities enhance the potential for extending the applicability of VRecKey to target various VR devices. Thus, to evaluate whether VRecKey can launch keystroke inference attacks on different VR devices, we conducted further experiments by separately collecting data samples of IR signals for evaluation from another commercial VR device, the PICO 4 All-in-One headset. Figure 17 shows the evaluation results of recovering character-level and word-level keystrokes on the virtual keyboards inside the two VR headsets, where we find VRecKey achieves 83.3% T-1 accuracy and 91.8% T-3 accuracy in character-level inference, and 80.3% T-1 accuracy and 86.4% T-3 accuracy in word-level inference on the virtual keyboard of PICO 4 All-in-One, respectively. In particular, the performance of our proposed attack on the two commercial VR devices remains consistent, as their default virtual keyboards feature nearly identical layouts.

Different Orientations between Virtual Keyboard and IR Sensor Array. In § IV-C, we have proposed methods for estimating orientation angles between the virtual keyboard and the IR sensor array. To further explore the impact of different orientation angles, we individually collect data with an orientation angle of 0° (default settings), 30°, 45°, 60°, and 75°. Figure 18 shows the evaluation results of VRecKey in keystroke inference at the three mentioned orientation angles. We observe that VRecKey decreases to 83.5% T-1 accuracy and 92.3% T-3 accuracy in character-level inference, as well as 79.0% T-1 accuracy and 88.5% T-3 accuracy in word-level inference at the orientation angle of 30°, which shows minimal performance degradation. Nevertheless, when adjusting the orientation angle to 75°, character-level inference accuracy decreases to 58.8% T1 and 70.2% T-3 (approximately 25.5% drop), and word-level inference accuracy decreases to 53.5% T-1 accuracy and 64.6% T-3 accuracy (approximately 27.1% drop). Hence, the results show the effectiveness of the keystroke coordinates calibration method in VRecKey (§ IV-C), which maintains a promising keystroke inference performance within common orientation changes (*e.g.*, less than 60°).

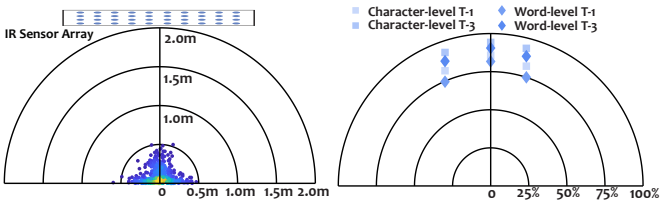
Different IR Sensor Arrays with Varying Cell Widths. In the current prototype of our customized IR sensor array (§ IV-B), we set the cell width between adjacent IR sensors as 5 cm. As is discussed in § IV-C, multiple adjacent IR sensors could capture the IR signals simultaneously, which also reflect on the generated matrices and heatmaps, which could affect the performance of the proposed attack. To investigate the impact of different cell widths between adjacent IR sensors on VRecKey's performance, we have designed and implemented two other IR sensor array boards with cell widths as 7.5 cm and 10.0 cm, respectively. Figure 19 shows the evaluation results when placing different IR sensor array boards in front of the VR user. In particular, VRecKey achieves 72.7% T-1 accuracy and 86.8% T-3 accuracy in character-level inference, as well as 69.3% T-1 accuracy and 85.3% T-3 accuracy in word-level inference when applying the IR sensor array with 7.5 cm cell width. In addition, when we select the IR sensor array with a cell width of 10.0 cm, the performance of VRecKey decreases to 55.0% T-1 accuracy and 51.2% T-3 accuracy in recognizing character-level keys, and 75.6% T-1 accuracy and 71.9% T-3 accuracy in recovering unconstrained word-level keystrokes. The findings illustrate that enlarging the cell width between adjacent IR sensors can result in a larger positional bias in capturing IR signals, which significantly degrades the performance and stealthiness of VRecKey. Therefore, we selected the fine-tuned configuration

of 5 cm cell width in designing the 2D IR sensor array.

Different Attacking Distances between VR Controllers and IR Sensor Array. In our primary experiments in Figure 13, we set the distance between the VR controllers and the 2D IR sensor array as 2.0 m. Nevertheless, the varying attacking distances could affect the captured IR sensors because of the signal attenuation and interference from the surrounding environment [24]. Hence, to investigate the impact of different attacking distances on the performance of VRecKey, we place the 2D IR sensor array at different attacking distances: 2.0 m, 2.5 m, 3.0 m, 3.5 m, and 4.0 m, and collect data samples of IR signals when typing on the virtual keyboard at each attacking distance to evaluate VRecKey, respectively. Figure 20 shows the evaluation results at different attacking distances. When the attacking distance is set to 2.5 m, VRecKey achieves 81.6% T-1 and 93.3% T-3 accuracy in character-level inference, and 75.7% T-1 and 88.4% T-3 accuracy in word-level inference. Furthermore, when the attacking distance is 4.0 m, VRecKey exhibits the performance of 52.8% T-1 and 45.8% T-3 accuracy in character-level inference, and 65.5% T-1 and 62.3% T-3 accuracy in word-level inference, respectively.

Hence, we notice that VRecKey's performance decreases with the increasing of the attacking distance between the VR controllers and the IR sensor array because of the attenuation of emitted IR signals from the infrared LED and the interference from surrounding environments. Moreover, we found that the IR sensor array is unable to capture IR signals when the attacking distance is over 5.0 m, which is much shorter than the typical receiving range of the KEYES 1838T infrared sensor receiver module boards (*i.e.*, typically 15 m [26]). A reasonable explanation could be the limited strength of IR signals emitted from the infrared LEDs on the VR controllers, which are designed for short-range communications between the cameras on the VR headset and the infrared LEDs on the controllers. For instance, most commercial-off-the-shelf (COTS) infrared LEDs used by commodity VR controllers present radiant intensity ranging from 4 mW to 125 mW (*e.g.*, OSRAM SFH 4055 [39]: 4–12.5 mW, Vishay TSAL6400 [40]: 25–125 mW, and Everlight IR333-A [41]: 7.8–20 mW), while resulting the insensitivity of being captured by the IR sensors. Nevertheless, our experiments still show that VRecKey realizes unconstrained keystroke inference with acceptable accuracy at practical attacking distances.

Different Typing Speeds. Previous studies have demonstrated that the typing speed on soft keyboards could influence the performance of keystroke inference attacks [30], [32], [42]. In § IV-E, we have demonstrated that a VR user usually types the virtual keyboard at a time interval ranging from 0.3 s to 2.8 s , and different typing speed leads to different response times of the captured IR signals, which may impact the VRecKey's performance in keystroke inference. Therefore, to understand the impact of VR controllers' typing speed on the virtual keyboard, we separately collect data samples of IR signals by typing the keyboards at three levels of typing speed: fast (typing interval < 0.5 s), medium (typing interval between 0.5 s and 2.0 s), and slow (typing interval > 2.0 s). Figure 21 shows the evaluation results when we individually type the virtual keyboard inside the Meta Oculus Quest 2 at a faster speed and a slower speed than the primary experiments, where we type virtual keystrokes at medium speed. The results demonstrate that VRecKey achieves 90.7% T-1 accuracy and 96.6% T-3 accuracy in character-level inference, and 84.3% T-1 accuracy and 92.4% T-3 accuracy in word-level inference when typing on the virtual keyboard at the slow speed. By contrast, when typing on the virtual keyboard at a faster speed level, VRecKey only presents the performance of 80.3% T-1 accuracy and 92.6% T-3 accuracy in character-level inference, and 77.5% T-1 accuracy and 87.3% T-3 accuracy in word-level inference. We have determined that there is a notable performance degradation of approximately 10.4% and 6.8% T-1 accuracy rates in character-level and word-level inference, respectively, when comparing the slow speed level to the fast speed level. This is attributed to the fact that a rapid typing speed results in shorter time intervals focused on the keys of the virtual keyboard, leading to less effective duration features for accurate keystroke inference.

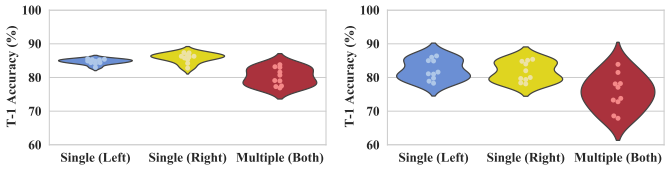


(a) User movement distribution. (b) Keystroke inference results.

Fig. 22: User movement analysis, including VR users' movements and keystroke inference results (§ V-F).

F. User Movement Analysis

In § V-A, we have illustrated that we allow participants to move casually and naturally when standing before the IR sensor array as they used the VR controllers to type on the virtual keyboard. Nevertheless, VR users' movements during the process of typing on the virtual keyboard could impact VRecKey's performance in keystroke inference. To further understand the user's movements and the potential influence, we record the VR user's standing point distribution in front of the IR sensor array during the typing process, as shown in Figure 22a. It shows that the VR user's movements are concentrated in the range between approximately 0 m and 0.6 m in both horizontal and vertical axes. We evaluated the keystroke inference performance of VRecKey when the VR user moves to the leftmost (-0.60 m) and rightmost (0.47 m) points. Figure 22b shows character-level and word-level keystroke inference results when launching VRecKey under the user's different moving statuses. In particular, VRecKey presents the highest performance when standing exactly in front of the IR sensor array. When the user moves to the leftmost point, the performance decreases by 4.5% T-1 and 1.6% T-3 accuracy in character-level inference, and 5.0% T-1 and 3.2% T-3 accuracy in word-level inference. Meanwhile, when



(a) Character-level inference. (b) Word-level inference.

Fig. 23: Performance in different IR source settings (§ V-G).

moving to the rightmost, the character-level inference accuracy decreases by 3.8% T-1 and 0.9% T-3, and word-level inference accuracy decreases by 6.7% T-1 and 2.2% T-3. Therefore, the results of user movement analysis have demonstrated that the VR user's movements during the typing process lead to limited impact on the keystroke inference from VRecKey, especially presenting a neglectable impact on T-3 accuracy.

G. Single v.s. Multiple IR Source Analysis

In § IV-D, we discussed the method of removing image retention in considering typing both left and right controllers. To further understand the impact of using VR controllers from a single hand or both hands, we collect data when typing at the virtual keyboard at three conditions: only left hand, only right hand, both left and right hand, and then evaluate VRecKey's performance in both character-level and word-level keystroke inference. Figure 23 shows the empirical results under the settings with a single IR source (left hand or right hand) and multiple IR sources (both hands). Specifically, we observe that VRecKey performs similarly in a single IR source but decreases approximately 5.1% T-1 accuracy in character-level and 6.5% T-1 accuracy in word-level keystroke inference because image retention exists when the typing interval is larger than a normal typing speed (*e.g.*, > 2.0 s). Overall, VRecKey maintains a promising accuracy with minor variations in recognizing virtual keystrokes from the infrared side channel under multiple input IR sources.

VI. Discussion

A. Countermeasures

Encrypted IR Transmission. With its reliance on LoS view and limited transmission of insensitive information, the current infrared (IR) transmission mechanism lacks essential security measures, such as encryption. However, these limitations are now being challenged due to the emergence of new deployment cases in VR devices and our proposed attack. Therefore, one potential countermeasure to defend against VRecKey is redesigning the IR communication protocol to incorporate encryption, thereby modifying patterns and preventing eavesdropping on the transmitted IR signals in such scenarios [43]. Similar to other protocols like Bluetooth [44], the foundation of IR encryption lies in establishing a shared key, achieved through methods such as Diffie-Hellman key exchange schemes [45], involving exchanging messages between the controllers and the VR headset to negotiate a shared secret.

To justify the effectiveness of IR encryption in defending against VRecKey, we build upon prior work [43] and propose two encryption schemes, illustrated in Figure 24a and Figure 24b. In both schemes, an 8-bit key is generated by a pseudo-random number generator (pRNG) [46]. In particular, scheme 2 includes an 8-bit counter to encrypt the IR signals each time a VR user types on the virtual keyboard. Specifically, Equation 11 presents the two encryption schemes as:

(a) IR encryption scheme 1.

(b) IR encryption scheme 2.

(c) Empirical results in applying IR encryption schemes.

Fig. 24: IR encryption schemes and defense results (§ VI-A).

$$\begin{cases} \text{Enc}_{S1}(\text{Tap}_8, \text{Key}_8) = \text{Tap}_8 \oplus \text{Key}_8 \\ \text{Enc}_{S2}(\text{Tap}_8, \text{Key}_8) = \text{Tap}_8 \oplus ((\text{Key}_8 + \text{Ctr}_8) \bmod 256) \end{cases} \tag{11}$$

To implement these encryption schemes, we tore down Meta Oculus Quest 2 controllers and integrated them with an extra Arduino Nano MCU to encrypt the transmitted IR signals. We then collected IR samples from the 2D sensor array in three different statutes and evaluated VRecKey's performance.

Figure 24c presents the empirical results. After applying the two encryption schemes, VRecKey's character-level inference performance decreased by approximately 7.1%–8.3% (T-1) and 5.6%–6.3% (T-3). Notably, word-level inference performance saw a more significant drop of about 23.2% (T-1) and 18.5% (T-3) with scheme 2 because of the multiple encryption actions triggered by the counting process while typing long-length words. Note that the experiments are conducted under default conditions for empirical comparison, and the countermeasures could be even more effective when combined with environmental interference in real-world scenarios. However, the integration of extra hardware into VR controllers could impact usability and does not guarantee security against brute-forcing attacks [47]–[49] or extended attacks from VRecKey if the transmitted key is extracted from the IR signals [50].

Shuffling Virtual Keyboards. Since VRecKey leverages captured IR signals from VR controllers to infer virtual keyboard input, another approach to mitigate our reported side-channel attack is to implement signal masking [51], [52]. That is, we can apply direct interference IR signal from other modulated light sources, *i.e.*, fluorescent lamps [53] and TV displays [48], with a similar carrier frequency (*e.g.*, 38 kHz [24]) of IR signals to obfuscate the captured signals of VRecKey. In addition, to protect VR users from keystroke inference attacks, it has been demonstrated that shuffling soft keyboards [24], [32], [54]–[58] could be an effective approach, as the attacker is unable to know the randomized layout of the keyboard and cannot further infer sensitive keystrokes. Hence, VRecKey

cannot effectively recover the specific keystrokes from the virtual input of a shuffled keyboard. Nevertheless, applying signal obfuscation may interfere with the communication between the controllers and the VR headset, which impacts the link quality of the communication [59], [60] and further lowers functionality and usability. Likewise, as demonstrated by [6], shuffling virtual keyboards for each interface or after every key tap could increase the time spent typing on the keyboard and affect usability, especially for long keystroke input.

B. Limitations and Future Works

Despite the promising evaluation results regarding the effectiveness of `VRecKey`, there are still several limitations in our current research. In particular, `VRecKey` requires to place the IR sensor array near the VR user to capture the leaked IR signals from the VR controllers. However, it is worth noting that `VRecKey` can be effective in different real-world scenarios, which outperforms other state-of-the-art non-intrusive VR keystroke inference attacks that mandate placing cameras in the LoS view with sufficient light intensity and close proximity. Moreover, leveraging high-resolution IR cameras (*e.g.*, CoolEYE 2D module [61]) to capture thermal radiations from the ambient environment like prior studies [62] is impractical because the weak strength of IR signals emitted from VR controllers could be overwhelmed in the captured thermal images.

Additionally, our newly disclosed infrared side channel provides an orthogonal and complementary solution to other side-channel explorations in VR privacy leakages, such as unencrypted network traffic in multi-user apps or gaze information from eye-tracking sensors [8], [63]. The current prototype of `VRecKey` performs optimally when the IR sensor array is placed within a $2.0\,\mathrm{m}$–$4.0\,\mathrm{m}$ range of the target VR user. This limitation stems from the inherent constraints of the IR signals emitted by commercial VR controllers, including their limited strength and transmission attenuation. While we believe the $2.0\,\mathrm{m}$–$4.0\,\mathrm{m}$ range sufficiently demonstrates the feasibility of this novel side-channel attack, enhancing the attacking distance will be a key focus of our future research efforts.

VII. Related Works

A. Keystroke Inference Attacks in VR

With the rise of Metaverse, recent studies have investigated VR attacks for stealing private information, *e.g.*, virtual keyboard input. Most of them exploit pre-installed malware to obtain data from built-in motion sensors (*e.g.*, accelerometer, gyroscope) of the VR headset and train multiple models to infer keystrokes [1]–[4], [10], [11]. Moreover, Meteriz-Yıldıran *et al.* [6] presents the first keylogging attack on the virtual keyboard by placing a camera or hand tracker to monitor the VR user's hand gestures. Similarly, Gopal *et al.* proposed the *Hidden Reality* attack [9] that utilizes cameras to record the hand gestures of the VR user for recognizing the typing keys on the virtual keyboard. In addition, *VR-Spy* [5] also shows the Wi-Fi CSI data can be hacked to monitor hand gestures and further infer keystrokes, whereas it imposes physical constraints that the user needs to sit between the transceivers.

Furthermore, *Heimdall* [10] shows the feasibility of leveraging the controllers' button-pressing sounds to infer virtual keystrokes, which also requires the placement of a recording device at close proximity (*e.g.*, $1.0\,\mathrm{m}$–$2.2\,\mathrm{m}$) and employs pre-trained models for inference, but its effectiveness decreases in noisy settings. Su *et al.* [7] shows that some multi-user VR apps, Rec Room, which adopts the unencrypted Photon protocol, leak the avatar's hand movement data in network traffic that results in keystroke inference attacks. One recent work, *GAZEploit* [8], exploits the avatar's gaze information recorded in online meetings to infer keystrokes typed with eye-tracking functionalities. Compared with these works, `VRecKey` presents the following three advantages: *(i)* it requires no malware installation and launches attacks non-intrusively at a relatively longer distance, *(ii)* it infers unconstrained virtual keystrokes without training specific machine learning models for classification, and *(iii)* it exploits a novel infrared side channel, which discloses an orthogonal solution with promising resilience and real-world practicality than prior attacks from other side channels.

B. Other Keystroke Inference Attacks

There are many efforts to exploit different side channels existing on mobile devices such as smartphones and tablets to infer people's keystrokes [64], [65]. For instance, an attacker can leverage the readings of built-in motion sensors like accelerometers, gyroscopes, and magnetometers [66], [67], system loads [30], [68], acoustic signals from microphones and speakers [69]–[72], and electromagnetic (EM) emanations from GPU [30] or induced by human-touchscreen coupling effects [32] to recognize input keystrokes. Furthermore, it is feasible to monitor changes in the channel state information (CSI) of wireless signals (*e.g.*, Wi-Fi) to steal the typing password [42], [73]. Besides, recent studies have demonstrated the power traces in smartphone charging processes, *i.e.*, USB charging [55], [74] or wireless charging [54], [75], exposing new attack surface for keystroke inference. In addition, *HomeSpy* [24] reveals that IR signals emitted from the remote control of a smart TV can be sniffed to infer input passwords and PIN codes. Likewise, `VRecKey` leverages the IR signals leaked from VR hand controllers to infer virtual keystrokes in an unconstrained and non-intrusive manner from the perspective of the infrared side channel that exists across devices (*e.g.*, smart TV [24]) in smart home scenarios [76].

VIII. Conclusion

In this paper, we present a novel side-channel attack for unconstrained keystroke inference on virtual keyboards in VR platforms by capturing the IR signals leaked from VR controllers designed for its constellation tracking system. To validate its feasibility, we design and implement `VRecKey`, an end-to-end attack framework that leverages a customized IR sensor array to non-intrusively capture IR signals emitted from infrared LEDs embedded in VR controllers and then recognize character-level keys and analyze typing paths to infer the consecutive keystrokes within virtual scenes. Our extensive

evaluation depicts that VRecKey achieves high accuracy in recognizing keystrokes with different lengths and presents promising resilience and practicality in real-world scenarios with varying conditions. We hope our findings can raise public awareness of the privacy leakage from the communication characteristics between the VR headset and the VR controllers and spur research on detecting forthcoming side-channel attacks and developing new defense approaches.

ACKNOWLEDGMENT

We sincerely appreciate our shepherd and all anonymous reviewers for their constructive feedback and invaluable comments. This work was fully supported by the Research Grants Council of Hong Kong (RGC) under Grants CityU 21219223, 11218521, 11218322, R6021-20F, R1012-21, RFS2122-1S04, C2004-21G, C1029-22G, C6015-23G, N_CityU139/21, and in part by the Innovation and Technology Commission of Hong Kong (ITC) under Mainland-Hong Kong Joint Funding Scheme (MHKJFS) MHP/135/23. This work was also substantially supported by InnoHK initiative, The Government of the HKSAR, and Laboratory for AI-Powered Financial Technologies (AIFT). Any opinions, findings, and conclusions in this paper are those of the authors and are not necessarily of the supported organizations.

REFERENCES

[1] C. Slocum, Y. Zhang, N. Abu-Ghazaleh, and J. Chen, "Going through the motions: AR/VR keylogging from user head motions," in *Proceedings of the 32nd USENIX Security Symposium*, 2023.

[2] Y. Zhang, C. Slocum, J. Chen, and N. Abu-Ghazaleh, "It's all in your head (set): Side-channel attacks on AR/VR systems," in *Proceedings of the 32nd USENIX Security Symposium*, 2023.

[3] Y. Wu, C. Shi, T. Zhang, P. Walker, J. Liu, N. Saxena, and Y. Chen, "Privacy leakage via unrestricted motion-position sensors in the age of virtual reality: A study of snooping typed input on virtual keyboards," in *Proceedings of the IEEE Symposium on Security and Privacy (SP)*, 2023.

[4] S. Luo, X. Hu, and Z. Yan, "Holologger: Keystroke inference on mixed reality head-mounted displays," in *Proceedings of the IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, 2022.

[5] A. Al Arafat, Z. Guo, and A. Awad, "VR-Spy: A side-channel attack on virtual key-logging in VR headsets," in *Proceedings of the IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, 2021.

[6] Ü. Meteriz-Yıldıran, N. F. Yıldıran, A. Awad, and D. Mohaisen, "A keylogging inference attack on air-tapping keyboards in virtual environments," in *Proceedings of the IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, 2022.

[7] Z. Su, K. Cai, R. Beeler, L. Dresel, A. Garcia, I. Grishchenko, Y. Tian, C. Kruegel, and G. Vigna, "Remote keylogging attacks in multi-user vr applications," *arXiv preprint arXiv:2405.14036*, 2024.

[8] H. Wang, Z. Zhan, H. Shan, S. Dai, M. Panoff, and S. Wang, "GAZEploit: Remote keystroke inference attack by gaze estimation from avatar views in VR/MR devices," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2024.

[9] S. R. K. Gopal, D. Shukla, J. D. Wheelock, and N. Saxena, "Hidden Reality: Caution, your hand gesture inputs in the immersive virtual world are visible to all!" in *Proceedings of the 32nd USENIX Security Symposium*, 2023.

[10] S. Luo, A. Nguyen, H. Farooq, K. Sun, and Z. Yan, "Eavesdropping on controller acoustic emanation for keystroke inference attack in virtual reality," in *Proceedings of the Network and Distributed System Security (NDSS) Symposium*, 2024.

[11] Z. Ling, Z. Li, C. Chen, J. Luo, W. Yu, and X. Fu, "I know what you enter on gear VR," in *Proceedings of the IEEE Conference on Communications and Network Security (CNS)*, 2019.

[12] H. Khalili, A. Chen, T. Papaiakovou, T. Jacques, H.-J. Chien, C. Liu, A. Ding, A. Hass, S. Zonouz, and N. Sehatbakhsh, "Virtual keymysteries unveiled: Detecting keystrokes in VR with external side-channels," in *Proceedings of the IEEE Security and Privacy Workshops (SPW)*, 2024.

[13] Pimax, "Pose tracking methods: Outside-in vs inside-out tracking in VR," https://pimax.com/pose-tracking-methods-outside-in-vs-inside-out-tracking-in-vr, 2023.

[14] D. Gajsek, "Vr controllers: The way of interacting with the virtual worlds," https://circuitstream.com/blog/vr-controllers-the-way-of-interacting-with-the-virtual-worlds, 2022.

[15] HTC VIVE, "Vive pro," https://www.vive.com/eu/product/vive-pro, 2023.

[16] Meta, "Meta Quest 2: Immersive all-in-one VR headset," https://www.meta.com/quest/products/quest-2, 2023.

[17] I. G. Gerloni, V. Carchiolo, F. R. Vitello, E. Sciacca, U. Becciani, A. Costa, S. Riggi, F. L. Bonali, E. Russo, L. Fallati *et al.*, "Immersive virtual reality for earth sciences," in *Proceedings of the Federated Conference on Computer Science and Information Systems (FedCSIS)*, 2018.

[18] PICO, "Live the game with PICO 4 all-in-one VR headset," https://www.picoxr.com/global/products/pico4, 2023.

[19] W. Greenwald, "HP Reverb G2 review," https://www.pcmag.com/reviews/hp-reverb-g2, 2023.

[20] PlayStation Blog, "Next-gen VR on PS5: the new controller," https://blog.playstation.com/2021/03/18/next-gen-vr-on-ps5-the-new-controller/, 2021.

[21] Twins Chip, "TSOP1838 infrared sensor," https://www.twinschip.com/TSOP1838_Infrared_Sensor, 2023.

[22] S. Daud, S. M. Sobani, M. Ramiee, N. Mahmood, P. Leow, and F. C. Harun, "Application of infrared sensor for shape detection," in *Proceedings of the IEEE International Conference on Photonics (ICP)*, 2013.

[23] Coavas Store, "Coavas One Way Privacy Window Film," 2024, https://www.amazon.com/Coavas-Privacy-Window-Film-Tools/dp/B0CLLWX5MS.

[24] K. Huang, Y. Zhou, K. Zhang, J. Xu, J. Chen, D. Tang, and K. Zhang, "HOMESPY: The invisible sniffer of infrared remote control of smart TVs," in *Proceedings of the 32nd USENIX Security Symposium*, 2023.

[25] Apple Inc., "Vision Pro," https://www.apple.com/apple-vision-pro, 2023.

[26] Elecbee, "1838T infrared sensor receiver module board remote controller IR sensor with cable for arduino," https://www.elecbee.com/en-26475-10pcs-1838T-Infrared-Sensor-Receiver-Module-Board-Remote-Controller-IR-Sensor-with-Cable-for-Arduino-products-that-work-with-official-Arduino-boards, 2023.

[27] Open Impulse, "Tl1838 infrared receiver datasheet," http://eeshop.unl.edu/pdf/VS1838-Infrared-Receiver-datasheet.pdf, 2023.

[28] G. Benet, F. Blanes, J. E. Simó, and P. Pérez, "Using infrared sensors for distance measurement in mobile robots," *Robotics and autonomous systems*, 2002.

[29] G. K. Uyanık and N. Güler, "A study on multiple linear regression analysis," *Procedia-Social and Behavioral Sciences*, vol. 106, pp. 234–240, 2013.

[30] B. Yang, R. Chen, K. Huang, J. Yang, and W. Gao, "Eavesdropping user credentials via gpu side channels on smartphones," in *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2022.

[31] C. Zhao and A. B. Chan, "Odam: Gradient-based instance-specific visual explanations for object detection," in *Proceedings of the 11th International Conference on Learning Representations (ICLR)*, 2023.

[32] W. Jin, S. Murali, H. Zhu, and M. Li, "Periscope: A keystroke inference attack using human coupled electromagnetic emanations," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2021.

[33] P. Cronin, X. Gao, C. Yang, and H. Wang, "Charger-Surfing: Exploiting a power line side-channel for smartphone information leakage," in *Proceedings of the 30th USENIX Security Symposium*, 2021.

[34] MathWorks, "Bounding box of polyshape," https://www.mathworks.com/help/matlab/ref/polyshape.boundingbox.html, 2023.

[35] H. Wu, W. Wang, Y. Wan, W. Jiao, and M. Lyu, "ChatGPT or grammarly? evaluating ChatGPT on grammatical error correction benchmark," *arXiv preprint arXiv:2303.13648*, 2023.

[36] M. C. Penteado and F. Perez, "Evaluating GPT-3.5 and GPT-4 on grammatical error correction for brazilian portuguese," *arXiv preprint arXiv:2306.15788*, 2023.

[37] H. R. Saeidnia, "Welcome to the gemini era: Google deepmind and the information industry," *Library Hi Tech News*, no. ahead-of-print, 2023.

[38] The University of Cambridge, "Cambridge English Vocabulary List," 2012, https://www.cambridgeenglish.org/images/84669-pet-vocabulary-list.pdf.

[39] OSRAM Opto Semiconductors, "Sfh 4055," https://look.ams-osram.com/m/7723eb0dcefc7b33/original/SFH-4055.pdf, 2019.

[40] Vishay Semiconductors, "High power infrared emitting diode," https://www.vishay.com/docs/81011/tsal6400.pdf, 2014.

[41] Everlight Europe, "IR333-A," https://everlighteurope.com/ir-emitters/159/IR333A.html, 2023.

[42] J. Hu, H. Wang, T. Zheng, J. Hu, Z. Chen, H. Jiang, and J. Luo, "Password-stealing without hacking: Wi-Fi enabled practical keystroke eavesdropping," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2023.

[43] M. Kim and T. Suh, "Eavesdropping vulnerability and countermeasure in infrared communication for iot devices," *Sensors*, 2021.

[44] J. Padgette, K. Scarfone, and L. Chen, "Guide to bluetooth security," *NIST special publication*, 2017.

[45] S. Kallam, "Diffie-hellman: key exchange and public key cryptosystems," *Master degree of Science, Math and Computer Science, Department of India State University, USA*, 2015.

[46] Leonardo Miliani, "pRNG.h: pretty Random Number Generator," 2016, https://github.com/leomil72/pRNG.

[47] T. Yao, K. Fukui, J. Nakashima, and T. Nakai, "Initial common secret key sharing using random plaintexts for short-range wireless communications," *IEEE Transactions on Consumer Electronics*, 2009.

[48] Y. Zhang, S. Ma, T. Chen, J. Li, R. H. Deng, and E. Bertino, "Evilscreen attack: Smart TV hijacking via multi-channel remote control mimicry," *IEEE Transactions on Dependable and Secure Computing (TDSC)*, 2023.

[49] S. Picek, G. Perin, L. Mariot, L. Wu, and L. Batina, "Sok: Deep learning-based physical side-channel analysis," *ACM Computing Surveys*, 2023.

[50] S. N. Premnath, S. Jana, J. Croft, P. L. Gowda, M. Clark, S. K. Kasera, N. Patwari, and S. V. Krishnamurthy, "Secret key extraction from wireless signal strength in real environments," *IEEE Transactions on Mobile Computing (TMC)*, 2012.

[51] L. Batina, S. Bhasin, D. Jap, and S. Picek, "CSI NN: Reverse engineering of neural network architectures through electromagnetic side channel," in *Proceedings of the USENIX Security Symposium*, 2019.

[52] T. Ni, X. Zhang, and Q. Zhao, "Recovering fingerprints from in-display fingerprint sensors via electromagnetic side channel," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2023.

[53] R. Narasimhan, M. D. Audeh, and J. M. Kahn, "Effect of electronic-ballast fluorescent lighting on wireless infrared links," *IEE Proceedings-Optoelectronics*, 1996.

[54] T. Ni, X. Zhang, C. Zuo, J. Li, Z. Yan, W. Wang, W. Xu, X. Luo, and Q. Zhao, "Uncovering user interactions on smartphones via contactless wireless charging side channels," in *Proceedings of the IEEE Symposium on Security and Privacy (SP)*, 2023.

[55] T. Ni, Y. Chen, W. Xu, L. Xue, and Q. Zhao, "Xporter: A study of the multi-port charger security on privacy leakage and voice injection," in *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2023.

[56] A. Alghamdi, A. Alkinoon, A. Alghuried, and D. Mohaisen, "xr-droid: A benchmark dataset for ar/vr and security applications," *IEEE Transactions on Dependable and Secure Computing (TDSC)*, 2024.

[57] N. F. Yıldıran, Ü. Meteriz-Yildiran, and D. Mohaisen, "Airtype: an air-tapping keyboard for augmented reality environments," in *Proceedings of the IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, 2022.

[58] H. Althebeiti, R. Gedawy, A. Alghuried, D. Nyang, and D. Mohaisen, "Defending airtype against inference attacks using 3d in-air keyboard layouts: Design and evaluation," in *Proceedings of the International Conference on Information Security Applications (ICISA)*, 2023.

[59] T. Ni, G. Lan, J. Wang, Q. Zhao, and W. Xu, "Eavesdropping mobile app activity via radio-frequency energy harvesting," in *Proceedings of the 32nd USENIX Security Symposium*, 2023.

[60] T. Ni, Z. Sun, M. Han, Y. Xie, G. Lan, Z. Li, T. Gu, and W. Xu, "Rehsense: Towards battery-free wireless sensing via radio frequency energy harvesting," in *Proceedings of the 25th International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing (MobiHoc)*, 2024.

[61] E. Technologies, "CoolEYE IR 2D Modules," 2020, https://www.exceli tas.com/product-category/cooleye-ir-2d-modules.

[62] Z. Yu, Z. Li, Y. Chang, S. Fong, J. Liu, and N. Zhang, "Heatdecam: detecting hidden spy cameras via thermal emissions," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2022.

[63] T. Ni, "Sensor security in virtual reality: Exploration and mitigation," in *Proceedings of the 22nd Annual International Conference on Mobile Systems, Applications and Services (MobiSys)*, 2024.

[64] G. Wang, C. Zhou, Y. Wang, B. Chen, H. Guo, and Q. Yan, "Beyond boundaries: A comprehensive survey of transferable attacks on ai systems," *arXiv preprint arXiv:2311.11796*, 2023.

[65] R. Spreitzer, V. Moonsamy, T. Korak, and S. Mangard, "Systematic classification of side-channel attacks: A case study for mobile devices," *IEEE Communications Surveys and Tutorials*, 2017.

[66] L. Cai and H. Chen, "Touchlogger: Inferring keystrokes on touch screen from smartphone motion," in *Proceedings of the 6th USENIX Workshop on Hot Topics in Security (HotSec)*, 2011.

[67] X. Liu, Z. Zhou, W. Diao, Z. Li, and K. Zhang, "When good becomes evil: Keystroke inference with smartwatch," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2015.

[68] M. Schwarz, M. Lipp, D. Gruss, S. Weiser, C. L. N. Maurice, R. Spreitzer, and S. Mangard, "Keydrown: Eliminating software-based keystroke timing side-channel attacks," in *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2018.

[69] L. Lu, J. Yu, Y. Chen, Y. Zhu, X. Xu, G. Xue, and M. Li, "Keylistener: Inferring keystrokes on QWERTY keyboard of touch screen through acoustic signals," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, 2019.

[70] Ü. Meteriz Yỳldỳran, N. F. Yỳldỳran, and D. Mohaisen, "Acoustictype: Smartwatch-enabled cross-device text entry method using keyboard acoustics," in *Proceedings of the CHI Conference on Human Factors in Computing Systems Extended Abstracts*, 2022.

[71] Ü. Meteriz-Yıldıran, N. F. Yildiran, and D. Mohaisen, "Sia: Smartwatch-enabled inference attacks on physical keyboards using acoustic signals," in *Proceedings of the 20th Workshop on Workshop on Privacy in the Electronic Society*, 2021.

[72] Y. Chen, T. Ni, W. Xu, and T. Gu, "Swipepass: Acoustic-based second-factor user authentication for smartphones," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, 2022.

[73] E. Yang, Q. He, and S. Fang, "WINK: Wireless inference of numerical keystrokes via zero-training spatiotemporal analysis," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2022.

[74] Y. Su, D. Genkin, D. Ranasinghe, and Y. Yarom, "USB snooping made easy: crosstalk leakage attacks on USB hubs," in *Proceedings of the 26th USENIX Security Symposium*, 2017.

[75] T. Ni, J. Li, X. Zhang, C. Zuo, W. Wang, W. Xu, X. Luo, and Q. Zhao, "Exploiting contactless side channels in wireless charging power banks for user privacy inference via few-shot learning," in *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2023.

[76] H. Chi, Q. Zeng, and X. Du, "Detecting and handling IoT interaction threats in multi-platform multi-control-channel smart homes," in *Proceedings of the USENIX Security Symposium*, 2023.

[77] S. Neamoniti and V. Kasapakis, "Hand tracking vs motion controllers: The effects on immersive virtual reality game experience," in *Proceedings of the IEEE International Symposium on Multimedia (ISM)*, 2022.

[78] H. Zhu, W. Jin, M. Xiao, S. Murali, and M. Li, "Blinkey: A two-factor user authentication method for virtual reality devices," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, 2020.

[79] H. Zhu, M. Xiao, D. Sherman, and M. Li, "Soundlock: A novel user authentication scheme for vr devices using auditory-pupillary response." in *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2023.

APPENDIX

A. Outlook of IR Sensor Array Prototype

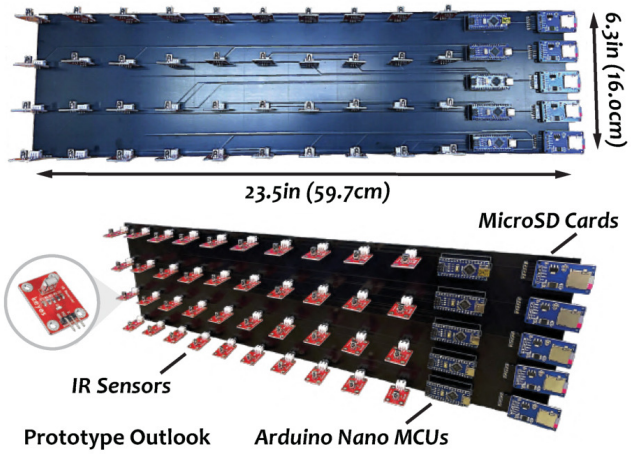


Fig. 25: Prototype of the customized 2D IR sensor array (§ IV-B), which consists of 40 1838T IR sensors [26], five Arduino Nano MCUs, and five MicroSD card adapters.

B. Controller Omnidirectional Analysis

As discussed in § II-A, multiple infrared LEDs are embedded around the ring of the VR controller, resulting in an omnidirectional IR radiation pattern. This means that IR signals emitted from the VR controllers disperse in all directions when the user unconsciously rotates their arm, potentially causing additional interference with the IR sensor array. To investigate the effects of omnidirectional radiation, we instructed participants to type the same key on a virtual keyboard at various wrist orientation angles. Figure 26a and Figure 26b illustrate the heatmaps generated when the user holds the VR controller to type the key "G" at up-to-down orientation angles ranging from $-45°$ to $45°$ and left-to-right orientation angles ranging from $-90°$ to $90°$, respectively. We observe that, although there are slight variations in the generated IR heatmaps, the densest color regions consistently focus on the same key on the virtual keyboard. This indicates that the omnidirectional emissions of IR signals from the VR controller have a limited impact on `VRecKey`'s keystroke inference performance. Given the limited power of the embedded LEDs and the substantial distance between the VR controllers and the IR sensor array, multiple LEDs on a VR controller can be effectively summarized as a single IR source. This is because the majority of the IR signals' power from these LEDs is directed towards the front, pointing towards the target virtual key, thereby leading to keystroke leakage from the infrared side channel. Note that we considered the up-to-down and left-to-right orientation angles for the participants' hands under typical keyboard typing conditions.

C. Controller-based v.s. Controller-less VR

Recently, Apple released its highly-anticipated VR headset, the Apple Vision Pro [25] in February 2024. Unlike traditional

(a) Up-to-down arm rotation. (b) Left-to-right arm rotation.

Fig. 26: Controller omnidirectional analysis (§ B).

VR devices, the Apple Vision Pro selects a controller-less design, revolutionizing user interaction by incorporating hand-tracking technology. On the other hand, other VR devices like the Meta Oculus Quest 2 and PICO 4 All-in-One have already supported native hand tracking [77]. Consequently, when VR users rely on hand-tracking mode, there is no emission of IR signals, rendering `VRecKey` ineffective in such scenarios. Nevertheless, we have investigated 10 popular VR devices from different metrics, and Table II show that most VR devices still adopt hand controllers to enhance user interactions within virtual environments for several reasons:

- **Lack Haptic Feedback:** Compared to tactile feedback from pressing physical buttons on VR controllers, using hand-tracking mode to interact with the VR headset often lacks haptic feedback, resulting in a higher false positive rate.
- **High Response Latency in Hand-tracking Mode:** The response latency experienced in the VR user's interactions with hand-tracking cannot match the high responsiveness of other controller-based VR interactions [14].
- **Limited Availability of Hand-tracking Scenarios:** There are a limited number of commodity VR apps that currently support the hand-tracking mode, which restricts VR apps' generalization across different VR platforms.
- **Threats of Potential Replay Attacks:** Using embedded cameras to track hand gestures is susceptible to imitation replay attacks [78], [79] because human gestures are easy to be monitored, simulated and replicated by adversaries.

As such, we can observe that the majority of mainstream VR devices, including Meta Oculus Quest, PICO, and HTC VIVE, continue to incorporate hand controllers in their products and over 50% of them adopt the IR-based constellation tracking systems, making `VRecKey` an applicable attack.

VR Devices	Hand Controllers?	Constellation?	Hand-tracking?
Meta Oculus Quest 2	●	●	●
PICO 4 All-in-One	●	●	●
HTC Vive Pro 2	●	●	●
Sony PlayStation VR 2	●	○	○
Meta Oculus Quest Pro	●	●	●
Meta Oculus Quest 3	●	●	●
Valve Index	●	●	○
HP WMR Headset	●	●	○
Dell Visor	●	○	○
Apple Vision Pro	○	○	●

TABLE II: Investigation results of 10 popular VR devices.