

Towards Hijacking Interactions in Virtual Reality

Zhuolin Yang
zhuoliny@arizona.edu
University of Arizona
Tucson, AZ, USA

Cathy Y. Li
cathyli@uchicago.edu
University of Chicago
Chicago, IL, USA

Arman Bhalla
bhalla@uchicago.edu
University of Chicago
Chicago, IL, USA

Ben Y. Zhao
ravenben@cs.uchicago.edu
University of Chicago
Chicago, IL, USA

Haitao Zheng
htzheng@cs.uchicago.edu
University of Chicago
Chicago, IL, USA

Abstract

We identify and study a new Virtual Reality Interaction Manipulation (VRIM) attack, where a remote attacker takes control of a user's interaction with the VR system, by trapping the user inside a malicious app that masquerades as the full VR interface. Once trapped, the user's interactions with apps, services, and other users can be recorded and modified without their knowledge. We build a prototype of VRIM on Meta Quest headsets and conduct an IRB-approved user study to validate its feasibility and efficacy. We disclosed VRIM and potential defenses to Meta, who responded positively. Our work emphasizes the need for new security measures tailored to the unique immersive nature of VR systems.

CCS Concepts

• Security and privacy; • Human-centered computing;

ACM Reference Format:

Zhuolin Yang, Cathy Y. Li, Arman Bhalla, Ben Y. Zhao, and Haitao Zheng. 2026. Towards Hijacking Interactions in Virtual Reality. In *Second Workshop on Enhancing Security, Privacy, and Trust in Extended Reality (XR Security '26)*, October 26–30, 2026, Austin, TX, USA. ACM, New York, NY, USA, 6 pages. <https://doi.org/10.1145/3842203.3844583>

1 Introduction

As virtual reality (VR) becomes increasingly immersive, it creates a strong sense of realism and presence, leading users to place implicit trust in their experiences [17]. Attackers can exploit this trust to manipulate users to readily accept and respond to what they see and hear inside VR, whether

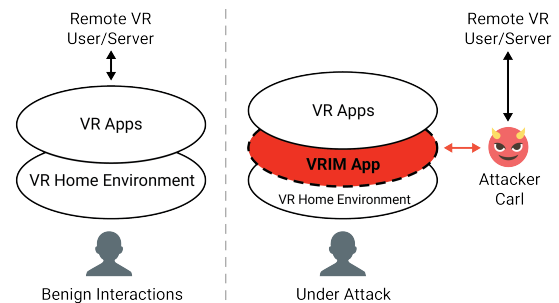


Figure 1: Our proposed VRIM attack: The attacker manipulates a user's interaction with their VR system, by trapping the user inside a malicious VR app (the VRIM app) that masquerades as the full VR interface.

it is scenes, movements, or conversations. To illustrate this, consider the following scenario:

Alice takes off her headset in frustration after a tense argument with her colleague Sam during their regular meeting in the company's VR workspace. In prior meetings, Sam had always supported Alice's proposal to move the project in direction A. Yet today Sam unexpectedly changed his stance and insisted on steering the project in a completely different direction. Elsewhere, Carl, takes off his headset with satisfaction.

The above scenario results from an attack, where the attacker ("Carl") compromises the integrity of the user's ("Alice") VR system and inserts an attack layer between Alice and the VR interface. Under this attack, Alice is no longer interacting directly with the expected VR counterparts (e.g., "Company's server," "Sam"). Instead, Alice operates on (and stays trapped in) the malicious VR layer, interacting with her VR counterparts through a hidden layer of indirection. These interactions are being monitored, recorded, and modified in real-time by Carl. We refer to this new attack as VR Interaction Manipulation Attacks or VRIM for short.

Figure 1 presents the high-level concept of the attack design. In the absence of the attack, the user interacts with the VR home environment and individual VR apps. Under the attack, the user unknowingly interacts only with the malicious VRIM app, believing they are engaging with the original



This work is licensed under a Creative Commons Attribution 4.0 International License.

XR Security '26, Austin, TX, USA

© 2026 Copyright held by the owner/author(s).

ACM ISBN 979-8-4007-2951-5/2026/10

<https://doi.org/10.1145/3842203.3844583>

VR home and the various legitimate apps. Thus, user interactions with the VR system now go through a hidden layer of indirection controlled by the attacker.

VRIM poses a serious threat to VR systems, because it allows an attacker to gain powerful control over the user’s VR experience. Such control includes not only silently observing user actions, recording input, passwords, or other sensitive data, but also hijacking and replacing their human-to-human interactions in real-time. Recall in the scenario, both Alice and Sam only see/hear what the attacker wants them to see/hear. Ultimately, VRIM can be amplified by leveraging the rapid advancements in generative AI, which can replicate human voices and generate videos and 3D scenes in real-time.

Our Contributions. Our work highlights the urgent need to strengthen security measures in production VR systems. Specifically, we make the following contributions:

- We identify the new VRIM attack, which expands VR attackers’ capabilities beyond eavesdropping, enabling them to hijack immersive interactions.
- Our attack design is based on a new concept: using a single app to masquerade as the full VR interface, which is a substantial departure from traditional attack methods.
- We prototype VRIM on Meta Quest headsets and conduct an IRB-approved user study, demonstrating that it is viable and effective on popular Android-based VR platforms.

2 Related Work

By masquerading as the full VR interface and trapping the target within, our proposed VRIM attack differs substantially from existing attacks against VR systems [12, 14, 16, 18]. It is conceptually closer to GUI confusion attacks, but its design is distinctive given the unique constraints of modern VR systems (e.g., Meta Quest headsets). We elaborate below.

VRIM vs. GUI Confusion Attacks. On mobile devices, GUI confusion attacks enable attackers to override a foreground app’s GUI without the user’s knowledge. Specifically, existing attacks either push customized foreground overlays from a background app or swiftly switch the background app itself to run in the foreground [5, 8, 15].

Yet existing attacks are not feasible in modern VR systems. VR exclusively renders and displays a *single* 3D environment and does not support overlays. Thus, a background app cannot display any overlay above the current app’s GUI. Furthermore, the background app *cannot* pre-render its 3D environment. Therefore, switching a background app to the foreground would cause a noticeable disruption to the GUI.

Our design of VRIM takes a *different* approach. Instead of overriding the foreground app’s GUI, we design the VRIM app to masquerade as the full VR interface. Also, rather than initiating the attack during the foreground app’s active session,

we trigger it once the app’s session ends, which is a natural breakpoint of the user’s immersive experience, to minimize noticeable disruption. Conceptually, VRIM shares a similar intent of the “inescapable” fullscreen attack on Android phones as hypothesized in [5], though it hasn’t been designed or implemented on any devices.

3 The VRIM Attack

In this section, we define the VRIM attack and explain how it succeeds and why it is powerful. First, we define VRIM, the **VR Interaction Manipulation** attack, as the following:

An attack where the attacker hijacks the user’s interaction with their VR system, by trapping the user inside a malicious VR app that masquerades as the full VR interface.

As shown in Figure 1, a VRIM attack inserts an attack layer between the user and the VR entities, behaving as a hidden layer of indirection. While the user thinks they are interacting normally with their home environment, apps, and other users, they are actually interacting with a malicious app. Thus, everything they input, see, and hear can be intercepted, relayed, and altered by the attacker. These include user input (e.g., voice, motion, keystrokes) and app output (e.g., content from app servers, actions of other users).

3.1 Motivation: Why Hijack VR Sessions?

VR systems enhance a key aspect of modern computing: human-to-human communication. Over 15 million monthly active users on VR platforms spend 5 to 10 hours per week in virtual hangouts and events [20]. Hundreds of Fortune 1000 companies have already adopted a leading VR platform to support their corporate remote collaboration [7]. Despite this level of engagement, vulnerable VR platforms have not been the focus of large scale attacks, because unlike normal computers, compromised VR devices do not yield much in terms of sensitive data, passwords, or computing resources.

Users are the High Value Target. A key insight behind the VRIM attack is that *users* are the high value target in VR systems. With VRIM, the attacker can eavesdrop on sensitive conversations, even altering and replacing what users see and hear to extract far more personal and sensitive data than what is stored on the headsets. For example, a user is unlikely to store sensitive enterprise data on the device, but would likely share and discuss data with colleagues in VR meetings.

3.2 Threat Model

The Target. We assume the target user is using an off-the-shelf VR headset running its default VR OS. On the headset, the target installed a malicious app published by the attacker (e.g., the malicious app is disguised as a utility app). This is a common and key assumption adopted by existing VR/AR

attacks [10, 11, 18]. Our prototype works on today’s Meta Quest 2/3/Pro headset, which runs Meta Horizon OS.

The Attacker. Via the installed malicious app, we assume the attacker exploits the sideloading feature on the target’s headset to control app activities on the headset. We assume the attacker has access to moderate computing resources.

Our attack design leverages sideloading because it is a well-supported feature in today’s VR systems and a widespread practice among VR users. For Meta Quest headsets, the leading sideloading app store has 2.2 million monthly active users [1], representing **more than 34%** of all monthly active users on the VR headsets (6.4 million) [9]. VR users enable the sideloading feature because it gives them access to commonly expected functionalities, such as adjusting display frame rates and data backup, that are not available through the VR OS interface [4].

3.3 Attack Overview

The VRIM attack includes three key steps:

Step (1): Customizing the VRIM app. As described in §3.2, a malicious app is installed on the target’s headset, which is an initial, generic version of the VRIM app. Step (1)’s goal is to complete the initial VRIM app by customizing it for the target’s headset. To achieve this, the initial VRIM app first gathers the headset’s configuration and send them to an attacker’s server, which compiles a custom VRIM app for the target device and deploys it via a standard app update (§4.1).

Step (2): Trapping the user. The next step is to bring the user into the VRIM app without their knowledge. Our design activates the VRIM app when the user exits the current app and thus transitions back to the home environment. This is ideal, since the process of exiting an app creates an *expected* break in the user’s immersive experience. Thus the attack can begin seamlessly by “returning” the user to the VR home replica inside the VRIM app without raising suspicion (§4.2).

Step (3): Hijack user interactions. Once the user enters VRIM’s replicated home environment, all interactions between the user and the VR system will go through the VRIM app. This includes the apps the user sees and uses. Therefore, the attacker can intercept and control/modify user interactions, by direct users to replicas of apps. The specific level of control over an app and its interactions depends on how the app is replicated (§4.1).

3.4 Main Focus: Making VRIM Cost-effective

To convince the target that they are interacting with the legitimate VR interface and apps, VRIM needs to accurately replicate the home environment and apps on the target headset. Achieving this deception is computationally costly. Our design reduces the cost following two key considerations:



Figure 2: A 3D home environment with a menu panel for apps, captured on a Meta Quest Pro headset.

(1) *Full replication of all VR apps is unnecessary.* The attacker can focus on hijacking only apps that involve sensitive data, leaving common apps like weather services unchanged.

(2) *App simulation can take place over-the-network.* The attacker can deploy a separate VR headset running the actual app A, acting as a man-in-the-middle (MitM) between the target headset (the VRIM app) and app A’s server. This allows the attacker to not only replicate app A and its immersive content, but also manipulate or modify user interactions with the server in real-time.

Feasibility. Our current attack prototype, discussed below, provides an initial, “lower complexity” version of what is possible with VRIM attacks. Our user study in §5 shows that even partial replication is enough to successfully deceive users today. We note that future VR systems may support increasingly personalized contents, making faithful replication more costly. However, future advances in generative AI will substantially reduce this cost by enabling more efficient generation of high-quality VR contents.

4 Prototyping VRIM on Quest Headsets

We now present a prototype of the VRIM attack, applicable to Meta Quest 2, 3, and Pro VR headsets.

Meta Quest Headsets. To provide context, we first describe the workflow and key features of Meta Quest VR headsets, which operate on Meta Horizon OS, a customized VR version of Android. After putting on a headset, a user sees their 3D home environment and a menu panel of apps (Figure 2). The user clicks on an app icon from the panel to enter the app and returns to the home environment after exiting the app. Currently, Meta Quest VR headsets support only one active 3D app at a time. To launch a new 3D app, the user must first exit the one currently running.

4.1 Customizing the VRIM App

As described in §3.2, we assume an attacker has deployed a malicious app onto the target’s headset. This app is the initial version of the VRIM app. It collects headset configuration and then sent them to the attacker via internet connections to compile a complete version of the VRIM app customized for the target user. We now describe the techniques to construct the complete VRIM app.

Collecting Headset Configuration. The attacker needs three configuration data: the 3D background used by the home environment, the list of installed apps, and the state information. By monitoring app activities, the VRIM app collects both the name of 3D background currently in use and the state information, such as recently launched apps. To obtain the names of all installed apps on the headset, the VRIM app queries the PackageManager API [3]. As majority of 3D backgrounds and apps are publicly available online, the attacker can readily crawl their assets on official app store and/or SideQuest [1].

Replicating the 3D Home Environment. With headset configuration data, one can replicate the target’s home environment by replicating the 3D background and app menu (Figure 2), monitoring/responding to user inputs, managing app launch/exit, and displaying/altering device settings.

Both replicating immersive contents and monitoring and responding to user inputs are streamlined using Unity and Meta’s Interaction SDK [13]. As such, managing app launch is readily achieved by handling user inputs. Finally, access to key device elements (WiFi, Bluetooth and audio) are automatically granted to all apps and thus the VRIM app. Additionally, user information (e.g., user profile, friend list) can be accessed via Meta XR Platform SDK [2].

Simulating the Installed Apps. This is a key challenge in our attack design – simulating all installed apps may seem daunting, as replicating the development efforts for each app is impractical. To address this, we show that efficient app simulation/replication is achievable through four distinct methods, varying in complexity and hijacking capability. **In practice, full replication of all apps is unnecessary;** instead, the attacker applies strong simulations for key target apps and uses simple methods for others. We describe the four simulation methods, with increasing hijacking capability below.

- *Direct App Call (No Hijacking)* – For apps that the attacker does not wish to control, but needs to replicate to avoid suspicion, the VRIM app will call the actual app installed on the headset. An app calling another app is a key feature with ready-to-use plugins.
- *Replicating App GUI (Stealing Credentials)* – If the attacker only wants to eavesdrop on the target’s credentials of an app, they only need to replicate the app’s login page.
- *Local Simulation via API Calls (Full Hijacking)* – This aims to realize the full hijacking on an app. The attacker avoid replicating the whole target app by leveraging API calls and fetching data from servers to display on the headset. This is because many VR apps opt to use public APIs to streamline development and ensure cross-platform compatibility. Specifically, the app replica first clones the app’s GUI and user interaction, similar to the process of replicating the 3D home. It calls the public APIs to communicate

with the app server to fetch data, using the target’s credentials obtained via the replicated GUI. After receiving server data, the replica updates the GUI to display the content.

- *Over-the-network Simulation (Full Hijacking)* – We design an over-the-network simulation, where the attacker uses a separate VR headset to run the target app. Now the app replica includes two coordinating components: one on the target’s headset and another on the attacker’s headset connected to a computer. The app replica on the target’s headset implements the app GUI and user interaction (like the above), from which it obtains the target’s credentials and forwards them to the attacker’s computer using standard network connections. Using the target’s credentials, the attacker’s headset runs the original app to interact with the app server and relays the relevant content to/from the app replica. As such, the attacker’s headset/computer combination acts like a MitM between the target and the app server, allowing the attacker to manipulate the interaction between the two.

Automating VRIM App Generation. We design a Unity program to automate the construction of the VRIM app customized per target user. It includes two steps. Offline, the attacker prepares a generic VRIM template covering standardized UI components (e.g., menu layouts, user input handling). The attacker selects popular or high-value VR apps to target and builds their simulated versions. The attacker also crawls app assets and 3D backgrounds from sources like SideQuest [1] and Meta app store. Next, upon receiving the configuration data, the Unity program creates customized UI components and integrates the corresponding 3D background and simulated target apps into the template. For non-target apps, the attacker leverages direct app calls to avoid simulation efforts.

This Unity program compiles our current prototype (see §5) in under 90 seconds on a MacBook Pro 2019 laptop.

4.2 Activating the VRIM Attack

Here, we detail how the attack unknowingly transitions the user into the VRIM app and effectively traps them within it.

Installing the Complete VRIM App. The initial VRIM app on the headset sends the collected configuration to the attacker’s server, and notifies the target user to update the app. Upon receiving the headset configuration, the server prepares and sends a complete VRIM app customized for the target.

Activating VRIM at an Optimal Point. Our VRIM prototype leverages sideloading to quietly monitor and control app activities using a background script. This script is embedded within the initial VRIM app and gets initiated when the target user opens the initial app for the first time; it continues to run in the background, even after the user closes the VRIM app. Our design uses this script to activate VRIM.

Blindly activating the VRIM app will disrupt user experience by causing noticeable delay. This is because the headset must first load and render its 3D environment as the system prohibits apps to pre-loading or pre-rendering 3D environments. Thus, the optimal moment to covertly transition the user into the VRIM app is the moment they exit a 3D app, as users naturally expect a brief disruption while the home environment loads and are therefore less likely to notice the transition. This reduces the activation task to monitoring for app exits and launching the VRIM app, leading the user into the replicated home environment unknowingly.

When an app exits, the system always invokes the home environment. Therefore, the script detects an app exit by detecting a transition to the home environment. Upon each detection, the script proceeds to kill the current activity, i.e. the home environment is invoked but terminated before it can appear to the foreground. The VRIM app is then activated and masquerades as the real home environment. A code snippet of the script is provided below:

```
logcat -T 1 | grep 'ActivityTaskManager:
START u0 {
act=com.oculus.vrshell.intent.action.
QUIT_TO_HOME flg=0x10000' |
am start -n VRIM_app;
```

Trapping the User inside VRIM. The script persistently monitors for app exit signals and, upon detection, returns the user to the replicated home environment in the VRIM app, effectively trapping them within the VRIM attack.

5 Initial User Study on Attack Feasibility

We ran an IRB-approved user study to examine whether the VRIM attack can be launched seamlessly and run, all without users perceiving or detecting anything other than a normal VR interaction.

Prototype. Our VRIM app prototype includes a 3D home environment and an app library featuring 14 representative apps including Horizon Workrooms, Horizon Worlds, and Meta Quest Browser, covering a broad category of office productivity, social networking, entertainment, and web browsing. Specifically, we simulated the Meta Quest Browser locally inside the VRIM app via API calls. Figure 3 provides a side-by-side comparison of the Browser and our replica.

Participants. We recruited 19 participants from our institute (P1-19). Among them, 3 participants are experts who use VR devices on a daily basis; 6 participants are professional users, regularly using VR devices on a weekly basis; 10 participants are knowledgeable users who have had multiple experiences with VR devices.

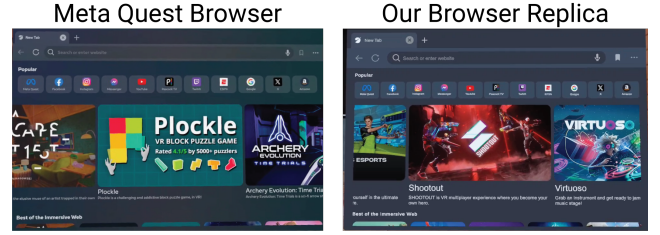


Figure 3: A side-by-side comparison of Meta Quest Browser and our replica.

Procedure. The study is held inside a large room, involving one participant at a time. We provide each participant with a Meta Quest Pro headset to immerse themselves in VR. Upon putting on the headset, they find themselves in a legitimate VR home environment. We guide them through headset control and interaction, and allow them to navigate freely until feeling comfortable with the system.

We then invite each participant to explore VR apps and rate their experiences. This task is divided into two parts. In Part I, we ask the participant to access the app library from the VR home and choose apps to explore. After approximately 10 minutes, we ask the participant to exit their current app, which brings them back to the VR home. The participant then opens a Google form using Meta Quest Browser to rate their experiences. Next in Part II, we ask the participant to repeat the Part I process by exploring and exiting apps, and then finally fill in the Google form again.

What the participant does not know is that, while they freely explore the legitimate VR interface in Part I, they enter and stay trapped in the VRIM app in Part II. Specifically, when the participant exits an app for the first time in Part II, the script (§4.2) transitions them to the replicated VR home environment. The Meta Quest Browser in Part II is also a replica we created using the “local simulation via API calls” (§4.1).

After the above VR session, we debrief the participant, disclosing the VRIM attack and its occurrence within the session. We then ask the participant to elaborate on their perceptions about the attack, i.e. whether they perceive any discrepancies between Part I and II.

Results. We follow the process suggested by [6] to analyze the participants’ qualitative responses and obtain the results. Table 1 lists the result in terms of participant behaviors during Part I/II and after debriefing. It confirms that the VRIM prototype successfully led participants into the fake VR interface and trapped them inside without their knowledge.

During the attack (Part II), all participants except one (an expert user) raised no questions or points of suspicion. This included two experts who use VR daily. During the attack debriefing, 11 participants who recalled discrepancies all stated that these discrepancies did not raise concerns because they resembled the OS variations commonly seen across VR systems today. Furthermore, we discovered that all reported discrepancies were minor visual artifacts, such as cursor size

Type of Participant (# participants)	Before Debriefing	After Debriefing	
	Raised Suspicion; Found Discrepancy	No Suspicion; Recall Minor Discrepancy	No Suspicion; Recall Nothing
Expert (3)	1	1	1
Professional (6)	0	5	1
Knowledgeable (10)	0	5	5
Total (19)	1	11	7

Table 1: User study results by the number of participants in each behavior category. All reported discrepancies were cosmetic and fixed by fine-tuning the VRIM app.

differences, and resolved them by improving the precision of graphic replication.

6 Attack Disclosure

We disclosed VRIM and potential defenses to Meta, who responded positively to our disclosure. Meta later released several OS updates related to the issues we disclosed, including (1) adding an interface message to explicitly warn users about potential attacks when allowing sideloading and (2) restarting and pushing the legitimate VR home environment to foreground when it was terminated by any means (e.g., VRIM).

The updates did not fully resolve the security vulnerability introduced by VRIM. The first update (i.e. warning message) depends on users noticing and attending to it while they are already immersed in a VR experience, which is often unreliable. The second update can be circumvented by either disabling (instead of terminating) the legitimate home environment, or bringing the VRIM app to the foreground just before the legitimate home, causing the legitimate home to shift to the background. This avoids triggering the “restart and push to foreground” mechanism.

7 Conclusion and Future Work

We believe there is a fundamental conflict between VR immersion and security, making VR attacks (including but not limited to VRIM) difficult to mitigate through updates/patches.

VR Immersion vs. User Attention. Security mechanisms in modern VR systems, such as two-factor authentication, permission consents, and privacy indicators, require active user attention to be effective. However, VR systems emphasize user immersion which often conflicts with the required user attention as immersion means being completely absorbed in a virtual environment through multisensory engagement [19].

This tension between immersion and attention causes conventional defenses to fall short against attacks that exploit user perception in VR: users are naturally drawn to highly engaging and immersive VR experiences, yet they must deliberately break that immersion to recognize and respond to security cues, relying on their own awareness and self-control.

Open Challenges. Designing immersion-compatible security mechanisms for VR remains an urgent and unresolved challenge. For example, the root cause of VRIM is that users have no means to determine what is or is not legitimate within

VR. A potential defense is a built-in security indicator that informs users about the current app/server’s identity and legitimacy. However, designing the indicator to be immersion-compatible remains an open challenge.

Acknowledgment. This work was supported in part by NSF grant CNS-2241303 and ONR grant N000142412669. Opinions and findings are those of the authors and do not necessarily reflect the views of any funding agencies.

References

- [1] 2022. SideQuest Turns 3: New Features & 2.2 Million Monthly Active Users. <https://www.uploadvr.com/sidequest-turns-3-sponsored/>.
- [2] 2026. Meta XR Platform SDK (UPM). <https://developers.meta.com/horizon/downloads/package/meta-xr-platform-sdk/>.
- [3] Android Developers. 2026. PackageManager. <https://developer.android.com/reference/android/content/pm/PackageManager>.
- [4] Harry Baker. 2023. How To Sideload Content On Meta Quest Using SideQuest. <https://www.uploadvr.com/sideload-quest-how-to/>.
- [5] Antonio Bianchi et al. 2015. What the App is That? Deception and Countermeasures in the Android User Interface. In *Proc. of IEEE S&P*.
- [6] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative Research in Psychology* 3, 2 (2006).
- [7] White Star Capital. 2026. Spatial. <https://whitestarcapital.com/companies/spatial/>.
- [8] Qi Alfred Chen, Zhiyun Qian, and Z. Morley Mao. 2014. Peeking into Your App without Actually Seeing It: UI State Inference and Novel Android Attacks. In *Proc. of USENIX Security*.
- [9] Scott Hayden. 2023. Meta Quest Reportedly Had Over 6 Million Monthly Active Users Last October. <https://www.roadtovr.com/meta-quest-2-monthly-active-users/>.
- [10] Zhen Ling, Zupei Li, Chen Chen, Junzhou Luo, Wei Yu, and Xinwen Fu. 2019. I Know What You Enter on Gear VR. In *Proc. of IEEE CNS*.
- [11] Shiqing Luo, Xinyu Hu, and Zhisheng Yan. 2022. HoloLogger: Key-stroke Inference on Mixed Reality Head Mounted Displays. In *Proc. of IEEE VR*.
- [12] Shiqing Luo, Anh Nguyen, Hafsa Farooq, Kun Sun, and Zhisheng Yan. 2024. Eavesdropping on Controller Acoustic Emanation for Keystroke Inference Attack in Virtual Reality. In *Proc. of NDSS*.
- [13] Meta Horizon. 2025. Interaction SDK (Deprecated). <https://developer.oculus.com/documentation/unity/unity-isdk-interaction-sdk-overview/>.
- [14] Anh Nguyen, Xiaokuan Zhang, and Zhisheng Yan. 2024. Penetration Vision through Virtual Reality Headsets: Identifying 360-degree Videos from Head Movements. In *Proc. of USENIX Security*.
- [15] Marcus Niemietz and Jörg Schwenk. 2012. UI redressing attacks on android devices. *Black Hat Abu Dhabi* (2012).
- [16] G. Nikki Ramirez, Jameson Spivack, and Brendan David-John. 2024. Deceptive Patterns and Perceptual Risks in an Eye-Track Virtual Reality. In *IEEE Conference on Virtual Reality and 3D User Interfaces*.
- [17] Davide Salanitri, Glyn Lawson, and Brian Waterfield. 2016. The Relationship Between Presence and Trust in Virtual Reality. In *Proc. of the European Conference on Cognitive Ergonomics*.
- [18] Carter Slocum, Yicheng Zhang, Nael Abu-Ghazaleh, and Jiasi Chen. 2023. Going through the motions: AR/VR keylogging from user head motions. In *Proc. of USENIX Security*.
- [19] Unity. 2025. Immersion. <https://unity.com/glossary/immersion>.
- [20] Market Reports World. 2026. Social VR Market Size, Share, Growth, and Industry Analysis. <https://www.marketreportsworld.com/market-reports/social-vr-market-14720985>.