

Characterizing and Enforcing Interaction Integrity on the Immersive Web

Chandrika Mukherjee
Purdue University
West Lafayette, IN, USA
cmukherj@purdue.edu

Abstract

The immersive web (WebXR) brings extended reality (XR) to the browser on head-mounted displays. UI properties such as transparency and same-space object overlap enhance immersion, but an adversary can manipulate these same properties to subvert user intent. The resulting UI-based attacks lead to sensitive data theft, malware exposure, and financial loss while users remain immersed and unaware. This dissertation develops *interaction integrity* for the immersive web, the guarantee that the interaction a user perceives is the interaction that actually occurs. We pursue this by characterizing the exploitable attack surface, understanding the developers who build immersive scenes as well as the users who experience them, and enforcing integrity at runtime. Our preliminary work demonstrates feasibility. We developed a taxonomy of WebXR UI-based attacks and showed in a 100-participant study that these deceptive patterns overwhelmingly evaded detection, and we built eXRPOSE, a runtime policy-enforcement system that detects every evaluated attack with negligible overhead. Building on these results, we outline directions that broaden this characterization as interaction modalities and model-generated content evolve, ground defenses in evidence from developers and users, and extend enforcement to keep pace with the surface as it grows.

CCS Concepts

• Security and privacy → Web application security; Usability in security and privacy.

Keywords

WebXR security, UI-based attacks, runtime monitoring

ACM Reference Format:

Chandrika Mukherjee. 2026. Characterizing and Enforcing Interaction Integrity on the Immersive Web. In *Proceedings of the 2026 ACM SIGSAC Conference on Computer and Communications Security (CCS '26)*, November 15–19, 2026, The Hague, Netherlands. ACM, New York, NY, USA, 3 pages. <https://doi.org/10.1145/3830454.3846936>

1 Introduction

WebXR is a W3C standard [20] that enables users to interact with 3D environments directly in the browser on head-mounted displays (HMDs) such as the Meta Quest and Apple Vision Pro. Users navigate to the WebXR scene by entering its URL in the HMD's browser

and then clicking the “Enter AR/VR” button to experience six degrees of freedom (6DoF). WebXR supports multiple input modalities, including controller raycasts, gaze input, and hand tracking. It also exposes UI properties such as transparency, object overlap, and synthetic (programmable) input for building immersive applications.

To enhance immersion or to monetize their scenes, developers integrate third-party content, e.g., advertisements, analytics services, and AI-driven virtual agents. Unlike the standard web, where this content loads inside isolating elements such as iframes, 3D scenes provide no equivalent containment. Moreover, XR properties such as the 360° scene extending beyond the user's field of view, unrestricted 3D object placement, and transparent objects can be exploited to trick users into unintended actions. These attacks lead to security and privacy harms, e.g., sensitive data theft and financial loss, for both end-users and other stakeholders.

This dissertation develops *interaction integrity* for the immersive web, the guarantee that the interaction a user perceives is the interaction that actually occurs. To pursue this goal, we characterize what makes immersive interaction exploitable, study how developers and users perceive these risks, and enforce integrity through formal policies that the platform evaluates at runtime. In this prospectus, we define the problem and our goals, outline the threat models we consider, present our preliminary results, and describe the ongoing efforts that build on them.

2 Related Work on Immersive UI Security

UI Attacks in XR. Attacks such as clickjacking and tapjacking have long manipulated user perception and interaction on the 2D web and mobile platforms [2, 7]. Recent work shows that these threats take a new form in immersive scenes. UI properties, such as same-space placement, opacity manipulation, and synthetic input, enable click and impression fraud, functionality disruption, and data extraction [5, 9]. Dark patterns exploit immersion itself, since presence and embodiment leave users less able to distance themselves from manipulation [8, 19]. These attacks arise from XR-specific interface properties and interaction modalities, not from direct adaptations of their 2D counterparts. Prior work, however, largely establishes feasibility without systematically analyzing the exploitable UI properties or their impact on users.

Defenses against UI Attacks in XR. 2D defenses confine embedded content to frame boundaries, and mobile platforms harden trusted UI against overlay abuse [3]. Both presuppose flat, bounded interfaces and miss the spatial composition, input modalities, and visibility manipulation of immersive scenes. XR defenses remain limited to sandboxing third-party content [9], post-hoc provenance analysis [18], and runtime monitoring of only a subset of insecure properties [10]. None enforces interaction integrity as it unfolds.



This work is licensed under a Creative Commons Attribution 4.0 International License. CCS '26, The Hague, Netherlands

© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2871-6/2026/11
<https://doi.org/10.1145/3830454.3846936>

Developer and User Studies in XR Security. Task-based developer studies have reshaped security guidance in other domains, on topics from password storage to the information sources developers consult while writing code [1, 17]. In immersive settings, a recent interview study charts how XR developers perceive security and privacy threats and finds them often unaware of the risks their design decisions introduce [4], while expert co-design studies surface emerging XR dark patterns [8]. How developers reason while building immersive scenes, and how users perceive model-generated manipulation within them, remain open questions.

3 Problem & Motivation

The immersive web renders an entire 3D experience inside a single HTML `<canvas>` element. Unlike a 2D page, where an `<iframe>` confines third-party content to a flat, bounded region, a WebXR scene has no isolation boundary. Third-party objects share the same scene graph, rendering context, and JavaScript execution environment as first-party content. Any party with code in the scene can create, move, resize, or restyle objects anywhere in the user’s surroundings, register listeners on any interaction, and dispatch events programmatically.

However, isolation alone would not resolve the problem, because the properties that produce immersion are precisely those that mediate interaction. Transparency enables glass- and water-like effects while transparent surfaces remain interactive; overlapping objects compose scenes in layers, with depth and occlusion deciding which layer is seen and clicked; the wraparound scene places content beyond the user’s field of view; and programmatic input lets objects act on one another without contact. Each of these properties is dual-use. A transparent object in front of an interactive target is benign as an invisible collision layer, malicious when it silently intercepts the clicks meant for that target. A synthetic click is benign as animation scripting, malicious when dispatched on an advertisement the user never sees. We call such properties *security-sensitive*. Because legitimate design and manipulation use the same vocabulary, manipulation is invisible by construction. The user’s attention is confined to the immersive view, so out-of-view interactions and layered deceptions in plain sight pass without notice.

Nothing in the medium marks these manipulations as wrong, nor is malice required for them to arise. Developers receive no guidance on which properties are security-sensitive, so well-intentioned integration of third-party content can reproduce the very patterns attacks exploit. Existing defenses [9, 10, 18] rely on trustworthy developer configurations, support only post-hoc forensics, or cover few exploitable properties. Correct configuration requires identifying which properties carry risk, yet whether developers can do so remains unstudied. Static analysis is no substitute, since violations depend on properties and interactions that exist only at runtime.

These manipulations harm every stakeholder in the ecosystem. Intercepted and rerouted clicks lead to cross-origin redirection, malware downloads, sensitive data theft, and financial loss, all while users remain immersed, unaware, and unable to attribute the failure to an attack. In the advertising ecosystem, honest advertisers pay for engagement no user produced, publishers lose their audience and absorb the reputational damage when an embedded party hijacks interactions, and ad-service providers risk losing the

publishers whose applications are degraded through their SDKs. In the aggregate, the medium itself suffers. The immersive web is still taking shape, and an ecosystem whose interactions cannot be trusted will calcify that distrust into its standards, its frameworks, and its users’ habits.

These manipulations all produce *unintended interactions*, in which what the user perceives diverges from what actually occurs. This divergence is observable at runtime as a pattern of scene-graph events, spatial conditions, and timing. Preserving the correspondence between perceived and actual interaction is precisely the interaction integrity this dissertation develops. This requires systematically investigating the UI properties that enable these manipulations, so the exploitable surface is known rather than anecdotal. It also requires understanding the stakeholders who shape and encounter this surface. Users may not perceive a loss of autonomy while immersed, and developers, in turn, shape this surface with little support. Finally, these manipulations must be expressed as formal policies that a trusted platform can enforce at runtime. Because the surface will keep growing, through shared multi-user scenes, emerging interaction channels, and generative models that produce scene content, enforcement must extend to new patterns rather than remain tied to a fixed list of attacks. Ultimately, we aim to move from detecting violations to preventing them.

4 Threat Model

We consider an adversary that manipulates the JavaScript-accessible properties of objects in a WebXR scene. The adversary may be a malicious developer controlling the application code and scene layout, or a malicious third party with write access through an included library, SDK, or script. In either case, the adversary can position or overlap objects, make them transparent, dispatch synthetic input, register listeners, or trigger redirections, all to induce unintended interactions. We assume a trusted platform provider, such as the browser, framework or operating system, integrates these defenses.

Our ongoing efforts also consider the *benign-but-fallible developer*, not an attacker but an actor whose mistakes produce exploitable patterns. With *model-mediated* interaction, a language-only or vision-language model that generates scene content or acts on the environment can synthesize and personalize deceptive patterns at runtime, while the physical surroundings become an input channel for instructing the model [21]. Model inference can further expose sensitive attributes that were never explicitly collected, extending harms to non-consenting bystanders [6]. Finally, in *multi-user* scenes, a malicious user manipulates avatars, objects, and social cues to deceive or surveil others [12].

5 Preliminary Work

We summarize our prior work [13–16] toward these goals.

Characterizing the Attack Surface. In *Shadowed Realities* [16], we analyzed the UI properties WebXR exposes, proposed five novel attack techniques exploiting them, and systematized existing and new attacks into a four-category taxonomy. We ground these categories in the advertising ecosystem, showing how each stakeholder can exploit a different set of UI properties to their own benefit. To measure their impact on users, we conducted a 100-participant between-subjects study. The deceptive patterns overwhelmingly

evaded detection while achieving their goals, establishing empirically that users cannot serve as their own defense. Further, a companion demonstration showed these attacks operating live on a commodity HMD [13].

Enforcing Policy at Runtime. We first outlined the vision of security-focused logging of user interactions as a defense against UI manipulation in WebXR [15]. Realizing this vision, EXRPOSE [14] is, to our knowledge, the first formally grounded runtime policy-enforcement system against UI-based attacks in WebXR. Informed by the UI-based attacks [5, 9, 16], it formalizes security policies in signal temporal logic [11] and evaluates them over runtime scene-graph events and spatial conditions to flag violations and alert the user. EXRPOSE requires no application cooperation; it observes interactions solely through platform-level instrumentation of the scene graph and input pipeline. Policies are therefore enforceable even when the application developer or an embedded third party is the adversary. On a Meta Quest 3 testbed spanning 15 distinct attacks across all four categories [16] in four WebXR applications, EXRPOSE detects every ground-truth violation with per-event overhead under 5 ms and amortized overhead below 1%. A 20-participant within-subjects study showed that its real-time alerts helped users recognize attacks they otherwise missed, with no statistically significant impact on presence or task load.

6 Ongoing Efforts and Directions

Here, we present our ongoing efforts toward our goals.

Understanding Developer Practices. Grounded in our attack taxonomy [16], our ongoing study examines how WebXR developers reason about and act on security-sensitive UI properties when integrating third-party content. Unlike prior interview-based work [4], we study what developers actually build. Its findings will inform where protections belong while WebXR frameworks and standards are still being written.

User Perception of Model-Mediated Deception. Our user studies showed that participants overwhelmingly fail to notice manipulation while immersed [16], and the model-mediated adversary compounds this risk by inferring far more during the interaction than users anticipate. Always-on scene interpretation and zero-shot cross-modal reasoning let a model infer far more during the interaction itself, where the inference can immediately be leveraged for harm. Through a scenario-based survey, we will study users' awareness of these harms and how it shapes trust and disclosure, informing what runtime warnings [14] must communicate.

Broadening the Attack-Surface Characterization. We plan to extend our taxonomy [16] to multi-user scenes [12], where avatars, objects, and social cues become manipulation vectors, and to emerging channels such as hand tracking and haptics. Most consequentially, language-model-driven scene generation lets an adversary synthesize and personalize deceptive patterns at runtime [21]. As the attack surface grows, we will automate policy synthesis so newly identified patterns yield formal policies without manual derivation.

Across these directions, we aim to advance enforcement from detecting unintended interactions to provably preventing them, informed by established trusted-path research, making interaction integrity a guarantee the immersive web provides.

7 Acknowledgments

I am grateful to my advisor, Professor Z. Berkay Celik, for his guidance and unwavering support throughout. I also thank my collaborators and the anonymous reviewers for their valuable feedback and insights. This research was partially funded by the National Science Foundation (NSF) through grants CNS-2144645 and IIS-2229876.

References

- [1] Yasemin Acar, Michael Backes, Sascha Fahl, Doowon Kim, Michelle L. Mazurek, and Christian Stransky. 2016. You Get Where You're Looking for: The Impact of Information Sources on Code Security. In *IEEE Symposium on Security and Privacy (S&P)*.
- [2] Philipp Beer, Marco Squarcina, Sebastian Roth, and Martina Lindorfer. 2025. TapTrap: Animation-Driven Tapjacking on Android. In *USENIX Security*.
- [3] Davide Bove. 2022. SoK: The Evolution of Trusted UI on Mobile. In *ACM on Asia Conference on Computer and Communications Security (ASIA CCS)*.
- [4] Kunlin Cai, Jinghui Zhang, Ying Li, Zhiyuan Wang, Xun Chen, Tianshi Li, and Yuan Tian. 2026. From Perception to Protection: A Developer-Centered Study of Security and Privacy Threats in Extended Reality (XR). In *NDSS*.
- [5] Kaiming Cheng, Arkaprabha Bhattacharya, Michelle Lin, Jaewook Lee, Aroosh Kumar, Jeffery F. Tian, Tadayoshi Kohno, and Franziska Roesner. 2024. When the User Is Inside the User Interface: An Empirical Study of UI Security Properties in Augmented Reality. In *USENIX Security*.
- [6] Gonzalo Munilla Garrido, Vivek Nair, and Dawn Song. 2024. SoK: Data Privacy in Virtual Reality. *Proceedings on Privacy Enhancing Technologies (PoPETs)*.
- [7] Lin-Shung Huang, Alex Moshchuk, Helen J. Wang, Stuart Schecter, and Collin Jackson. 2012. Clickjacking: Attacks and Defenses. In *USENIX Security*.
- [8] Veronika Krauß, Pejman Saeghe, Alexander Boden, Mohamed Khamis, Mark McGill, Jan Gugenheimer, and Michael Nebeling. 2024. What Makes XR Dark? Examining Emerging Dark Patterns in Augmented and Virtual Reality through Expert Co-Design. *ACM Trans. Comput.-Hum. Interact.*
- [9] Hyunjo Lee, Jiyeon Lee, Daejun Kim, Suman Jana, Insik Shin, and Soeul Son. 2021. AdCube: WebVR Ad Fraud and Practical Confinement of Third-Party Ads. In *USENIX Security*.
- [10] Bertram Liu, Vamsi Shankar Simhadri, and Xiaokuan Zhang. 2025. Hunting Insecure UI Properties in Extended Reality. In *ACM MobiHoc XR-Security Workshop*.
- [11] Oded Maler and Dejan Nickovic. 2004. Monitoring Temporal Properties of Continuous Signals. In *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*.
- [12] Andrea Mengascini, Ryan Aurelio, and Giancarlo Pellegrino. 2024. The Big Brother's New Playground: Unmasking the Illusion of Privacy in Web Metaverses from a Malicious User's Perspective. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*.
- [13] Chandrika Mukherjee, Reham Mohamed Aburas, Arjun Arunasalam, Habiba Farrukh, and Z. Berkay Celik. 2025. Demo: UI Based Attacks in WebXR. In *ACM International Conference on Mobile Systems, Applications and Services (MobiSys)*.
- [14] Chandrika Mukherjee, Aishwarya Devi Akila Pandian, Arjun Arunasalam, Jiasi Chen, Z. Berkay Celik, and Habiba Farrukh. 2026. EXRPOSE: Runtime UI-based Attack Detection in WebXR. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*.
- [15] Chandrika Mukherjee, Arjun Arunasalam, Habiba Farrukh, Reham Mohamed Aburas, and Z. Berkay Celik. 2025. Towards Secure User Interaction in WebXR. In *International Workshop on Human-Centered Sensing, Modeling, and Intelligent Systems*.
- [16] Chandrika Mukherjee, Reham Mohamed, Arjun Arunasalam, Habiba Farrukh, and Z. Berkay Celik. 2025. Shadowed Realities: An Investigation of UI Attacks in WebXR. In *USENIX Security*.
- [17] Alena Naiakshina, Anastasia Danilova, Christian Tiefenau, Marco Herzog, Sergej Dechand, and Matthew Smith. 2017. Why Do Developers Get Password Storage Wrong? A Qualitative Usability Study. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*.
- [18] Muhammad Shoaib, Alex Suh, and Wajih Ul Hassan. 2025. Principled and Automated Approach for Investigating AR/VR Attacks. In *USENIX Security*.
- [19] Wen-Jie Tseng, Elise Bonnail, Mark McGill, Mohamed Khamis, Eric Lecolinet, Samuel Huron, and Jan Gugenheimer. 2022. The Dark Side of Perceptual Manipulations in Virtual Reality. In *CHI Conference on Human Factors in Computing Systems*.
- [20] W3C WebXR Device API. <https://www.w3.org/TR/webxr/>. Accessed: 11-Sep-2026.
- [21] Yicheng Zhang, Zijian Huang, Sophie Chen, Erfan Shayegani, Jiasi Chen, and Nael Abu-Ghazaleh. 2025. Evil Vizier: Vulnerabilities of LLM-Integrated XR Systems. <https://arxiv.org/abs/2509.15213>.