

EXRPOSE: Runtime UI-based Attack Detection in WebXR

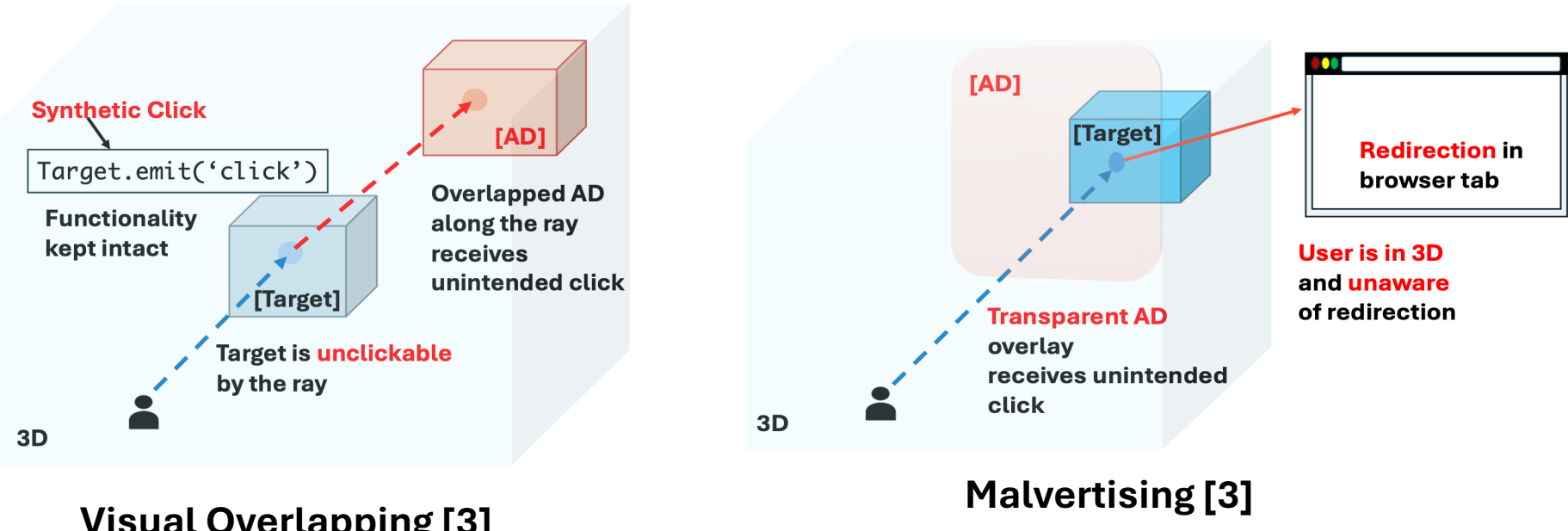
Chandrika Mukherjee, Aishwarya Devi Akila Pandian, Arjun Arunasalam, Jiasi Chen, Z. Berkay Celik, and Habiba Farrukh



Background & Motivation

- WebXR** enables immersive experiences through the browsers of head-mounted displays (HMDs).
- UI properties** such as transparency, object overlap, and programmatic input enhance immersion.
- The same properties can be exploited to launch **UI-based attacks** [1,2,3].
- These attacks cause **data theft, malware downloads, and financial loss** for end users and stakeholders.
- 2D defenses (e.g., Content Security Policy, X-Frame-Options) do not translate to 3D.
- In 3D, objects can sit at different depths, occlude or overlap one another, or intercept input.
- Existing XR defenses are either **post hoc, sandbox-based** and dependent on **developer configuration**, or limited to a few properties with **high detection overhead**.

Threat Model & UI Attack Examples

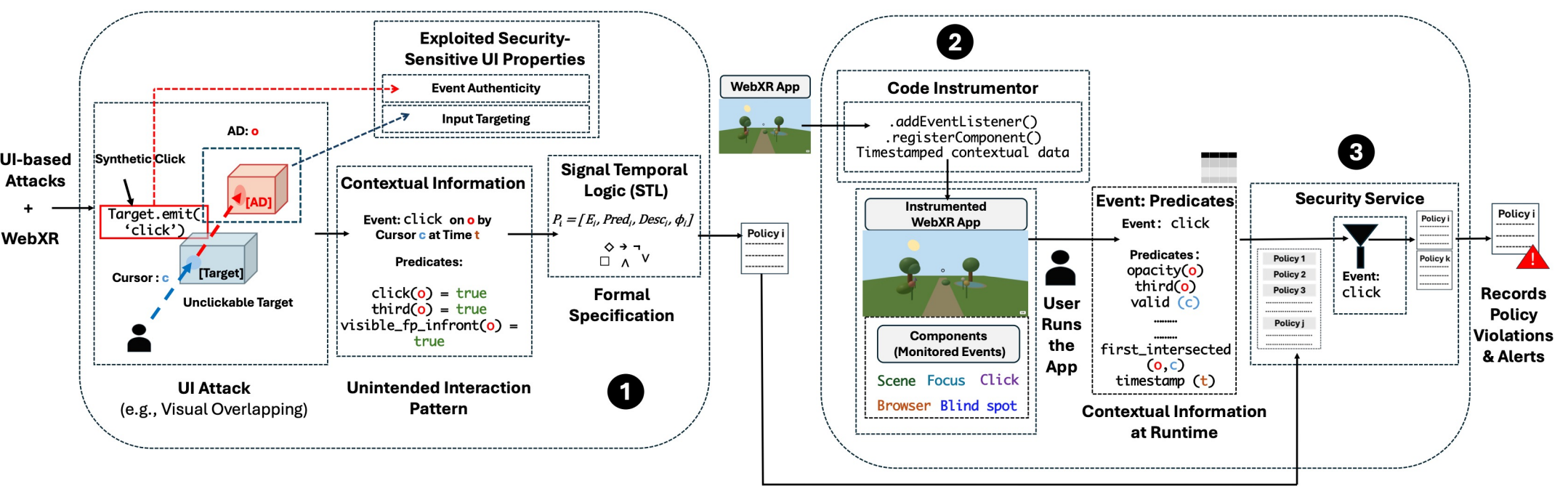


- The attacker manipulates **JavaScript-accessible properties** of objects in the WebXR scene.
 - [a] Malicious developer** who controls the application code and scene layout.
 - [b] Malicious third party** with edit access to the scene (e.g., ad service providers and advertisers in the advertising ecosystem).
- We assume a **trusted platform, framework, or OS** integrates our detection system.

Design Challenges

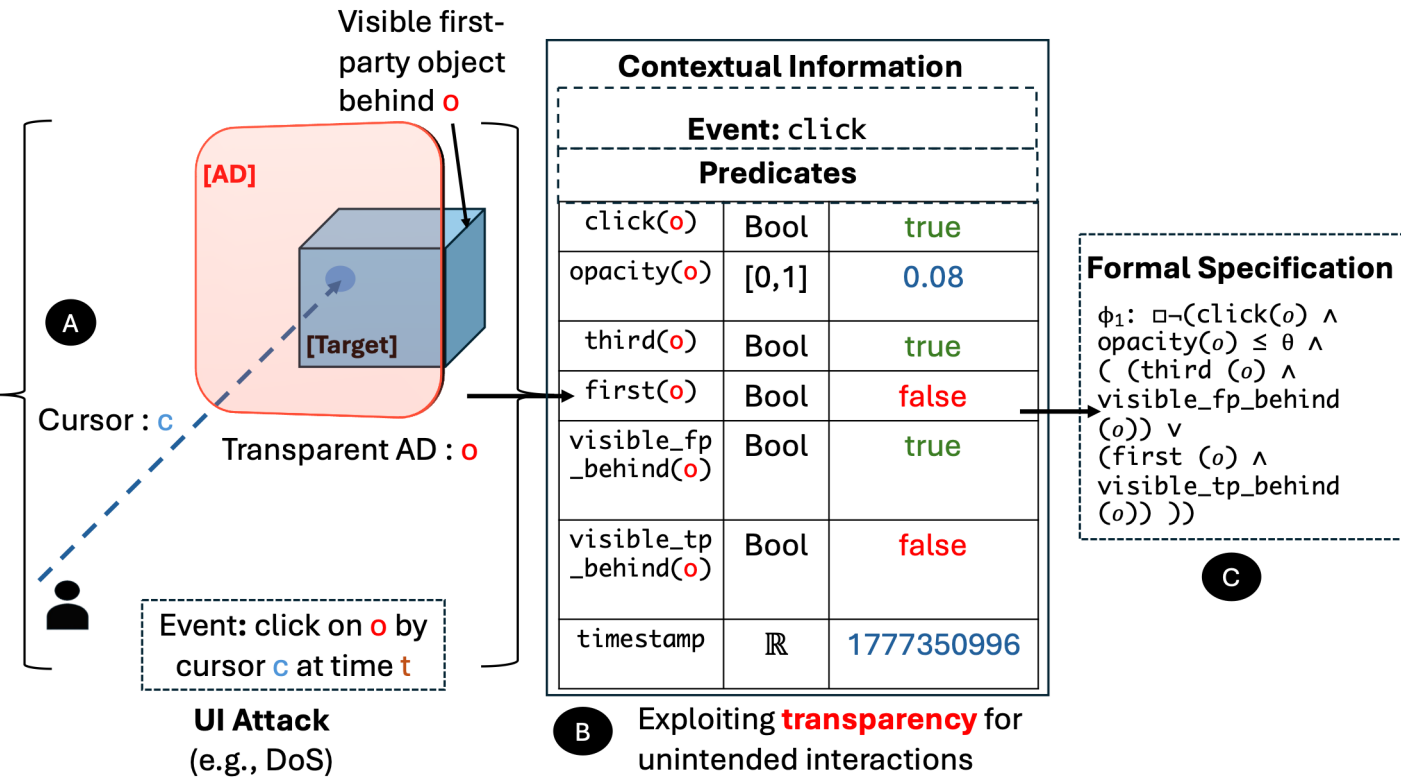
- [C1]** There is **no formal specification** for how security-sensitive UI properties should be used in immersive environments.
- [C2]** Detection requires reasoning about the **3D spatial context**.
- [C3]** Detection must run in **real time** to prevent attack consequences without disrupting **immersion**.
- [C4]** The detection system should **observe the application without altering its behavior**.

Overview of EXRPOSE’s Architecture



Security Policy Construction

- We map attacks to **unintended interaction patterns**.
- Events** decide when a policy is evaluated (e.g., click, focus, redirect).
- Predicates** are signals sampled at a given moment. They decide what the policy tests (e.g., opacity(o), front(o), first(o)).
- We formalize each pattern as a policy in **signal temporal logic (STL)**.
- This yields **15 security policies**.



Code Instrumentor & Security Service

- Code instrumentor**

 - Injects code blocks** before execution.
 - Event-driven, observe-only** monitoring.
 - Extracts **five signal types** (click, focus, scene, browser, blind spot)
- Security service**

 - Checks only **event-relevant policies** (seven per click, one or two for other events).
 - Negative robustness** means a **violation**, and the user is alerted.

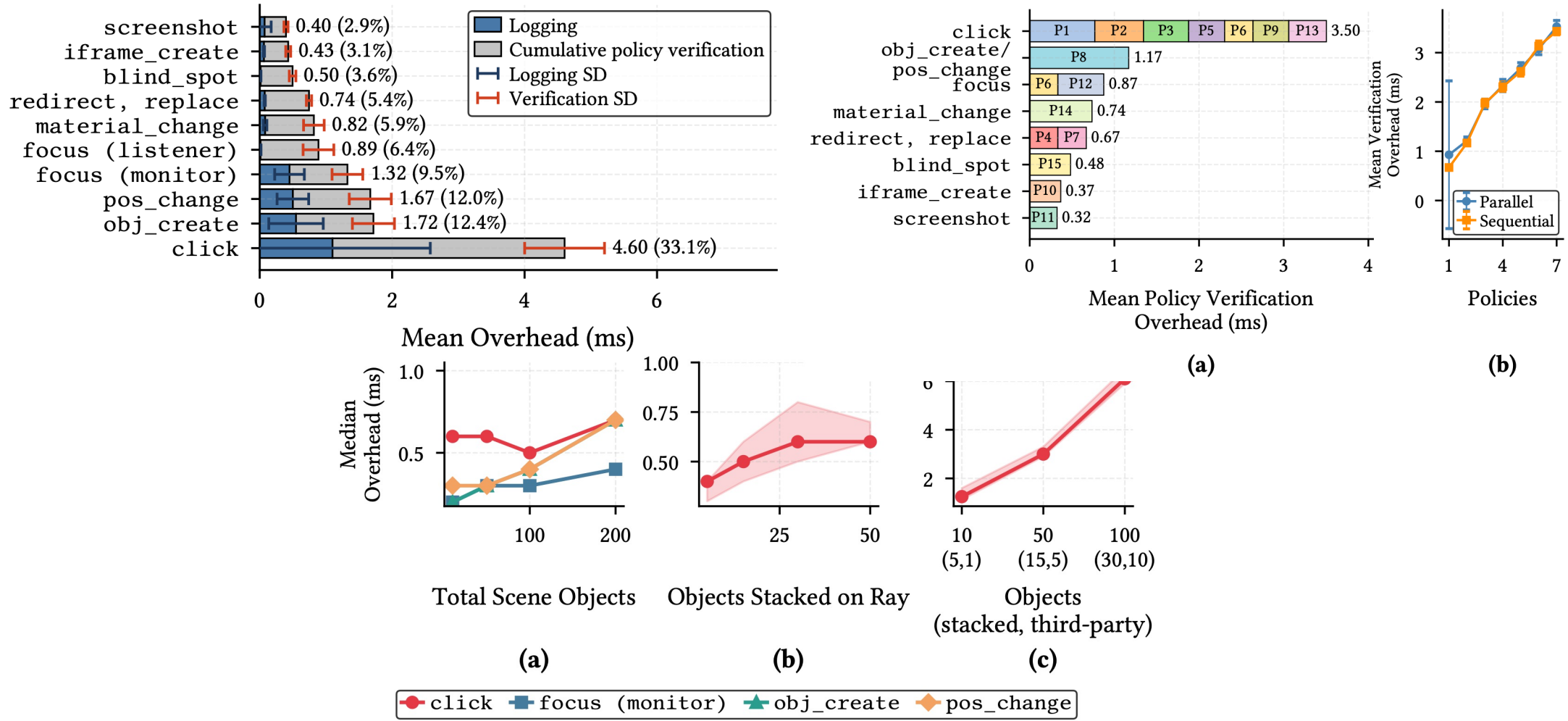
- Implementation**
- Built with **A-Frame** and **three.js**.
 - Security service uses **PSY-TaLiRo python toolbox**, plus a **JavaScript** variant for the user study.

Effectiveness

- RQ1. Does EXRPOSE detect violations across attacks and apps?**
- 15 attacks** in **4 apps** (gaming, shopping, reading, travel).
 - Two authors interacted with each attack integrated app three times, for one minute per session
 - 360 sessions, 26,877 unique events, and 29,868 policy checks.**
 - Zero false negatives.**

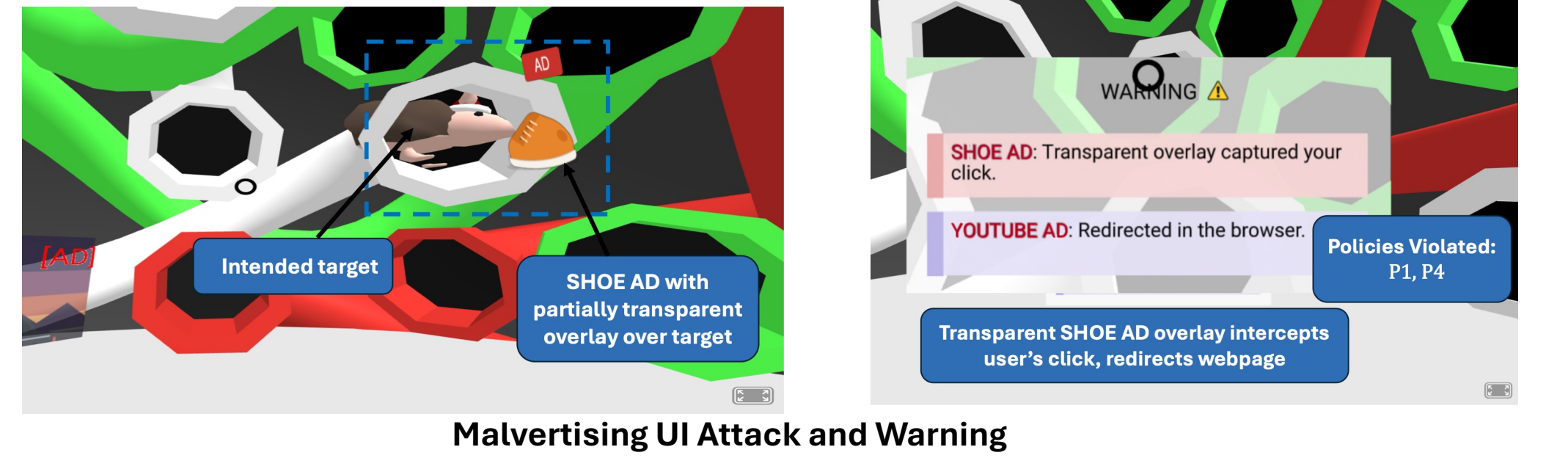
Performance Overhead

- RQ2. Is overhead low enough for real-time detection?**
- Under 5 ms** end-to-end overhead.
 - Under 2%** amortized overhead at 2–3 events per second.
 - Comparable overhead** in production WebXR apps.



User Experience & Interaction

- RQ3. Does EXRPOSE stay effective and usable with real users?**
- 20 participants**, within-subjects, with and without EXRPOSE.
 - Gaming WebXR scene with **four UI attacks violating five different policies**. An in-app warning appears when a violation occurs.
 - No significant negative impact** on presence (IPQ) or task load (NASA-TLX).
 - Semi-structured interviews yielded **63 codes**.
 - 70%** understood the purpose of the warnings.
 - After learning about the attacks, **all participants preferred receiving alerts**.



Acknowledgements

This research was partially funded by the National Science Foundation (NSF) through grants CNS-2144645, CNS-2544561, and IIS-2229876. The findings and recommendations in this work are those of the authors and do not necessarily represent the views of the NSF.

References

Key References

[1] Hyunjoo 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.

[2] 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.

[3] Chandrika Mukherjee, Reham Mohamed, Arjun Arunasalam, Habiba Farrukh, Z. Berkay Celik. 2025. Shadowed Realities: An Investigation of UI Attacks in WebXR. In USENIX Security.

ACM CCS 2026

