It’s all in your head(set):
Side-channel attacks on AR/VR systems

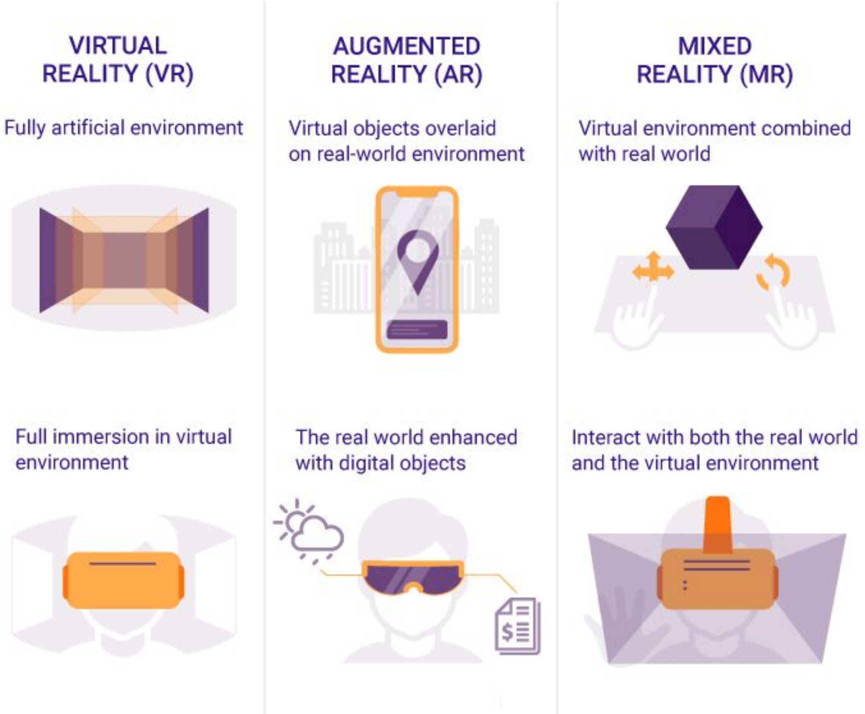
Yicheng Zhang, Carter Slocum, Jiasi Chen and Nael Abu-Ghazaleh,

USENIX Sec’23

Presented by SeungHyun Lee
@ KAIST Hacking Lab

What is AR/VR?

- Virtual Reality (VR) / Augmented Reality (AR) / Mixed Reality (MR)



AR/VR Systems

- Various commercial AR/VR devices for consumer & industrial use

Microsoft HoloLens

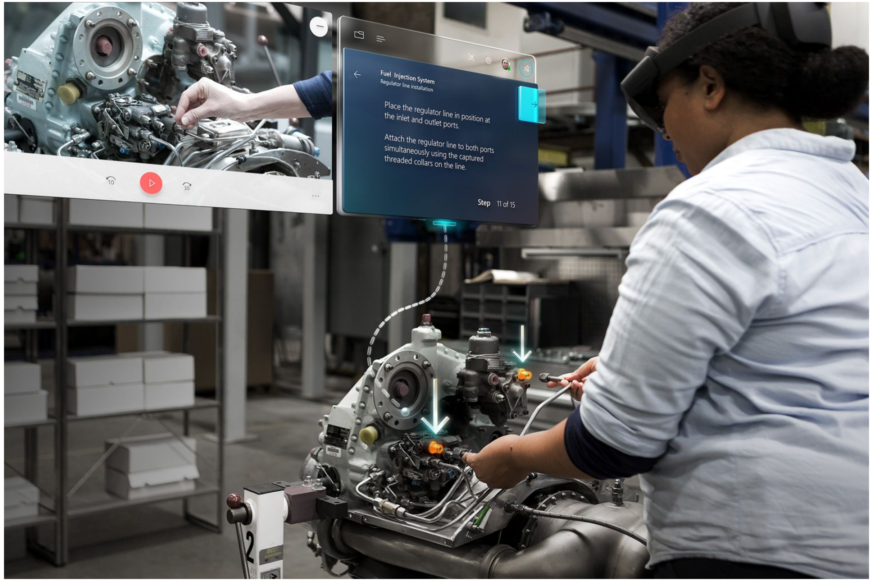
Meta Quest

Vision Pro

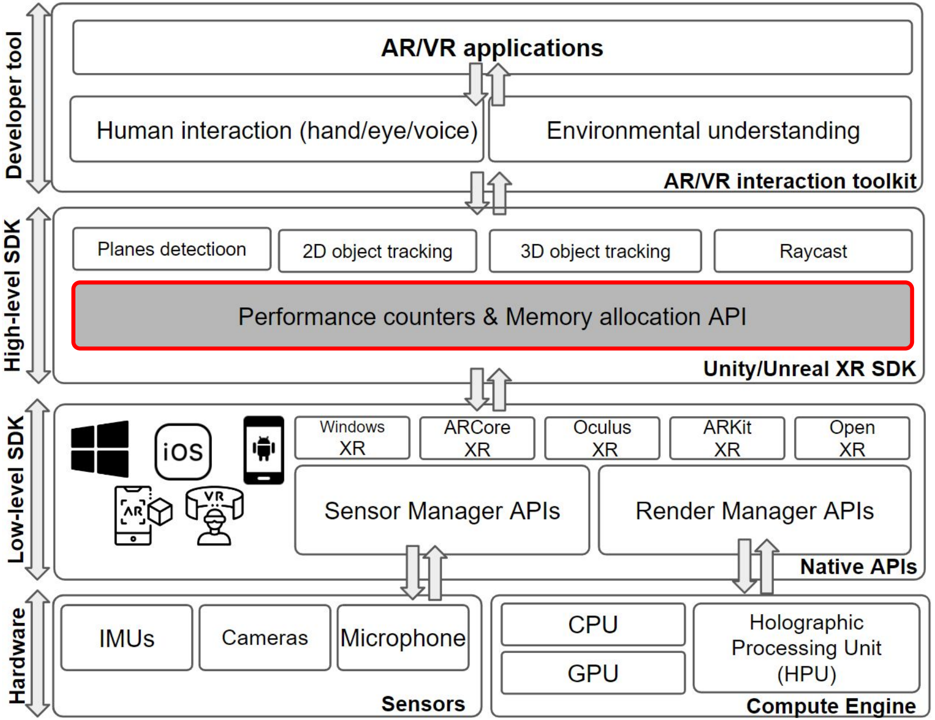
VIVE

Introduction

- **User interacts** with the AR/VR **environment**
- **Multiple apps run concurrently**, each providing a different service
    - Apps downloaded from the App Store, executed on the headset

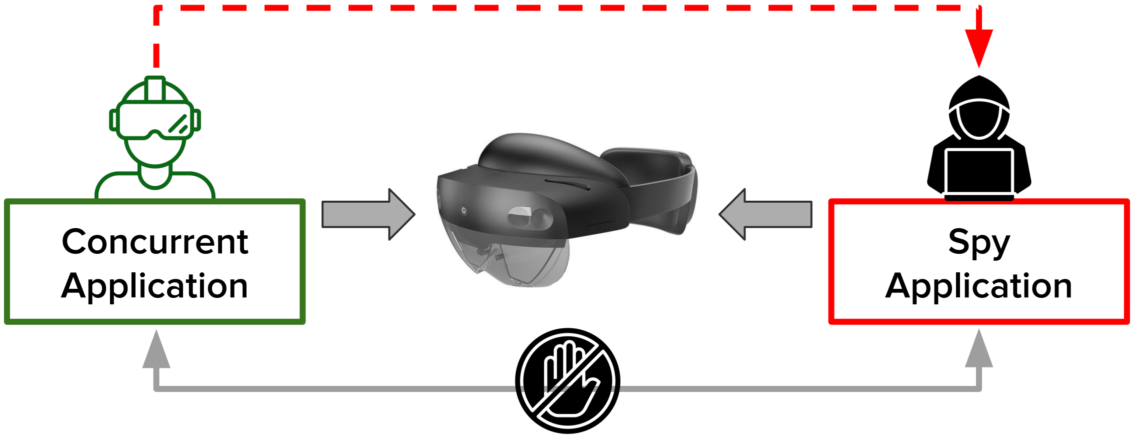


Background: AR/VR Systems Architecture



Threat Model: Software Side-channel Attack

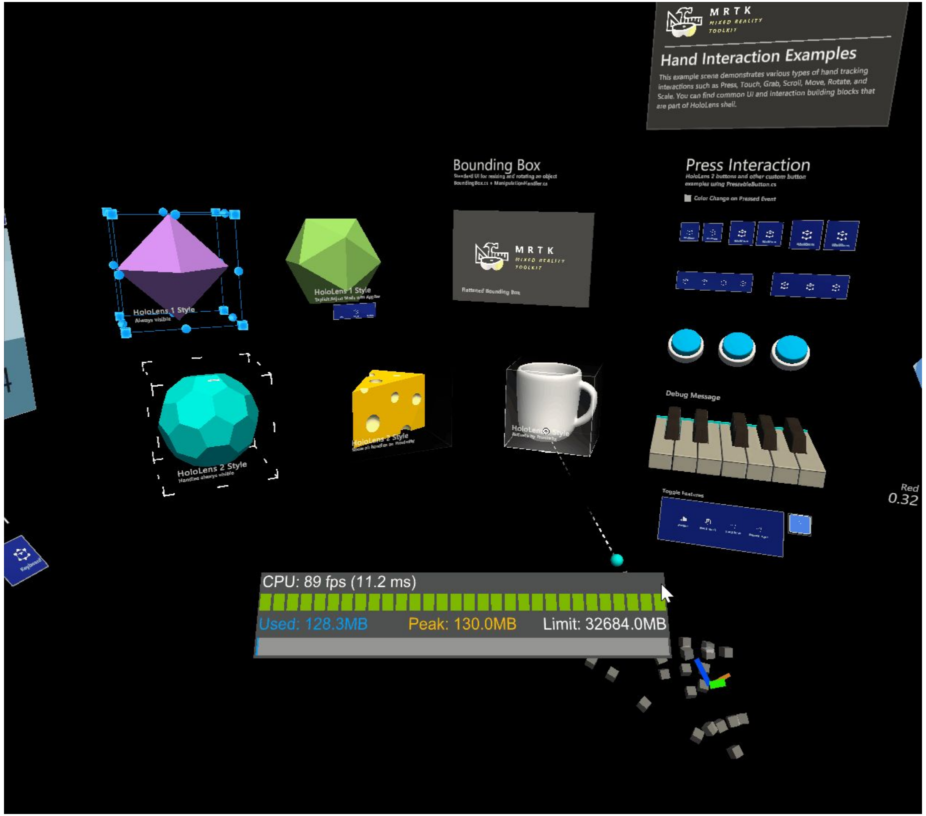
- A malicious program runs in the background
  - Standard application-level permissions
  - No physical access
  - Periodically probes **performance counters & memory allocation APIs**



Leakage Vectors

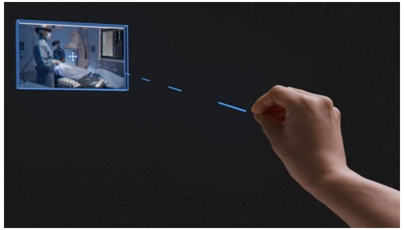
- Rendering Performance Counters:
  - Frame rate: CPU/GPU frame rate, refresh rate, GPU input latency, ...
  - Thread counters: Game/Render thread time, ...
  - Render task counters: Number of draw calls, Vertex count, ...
- Memory Allocation API:
  - App memory usage

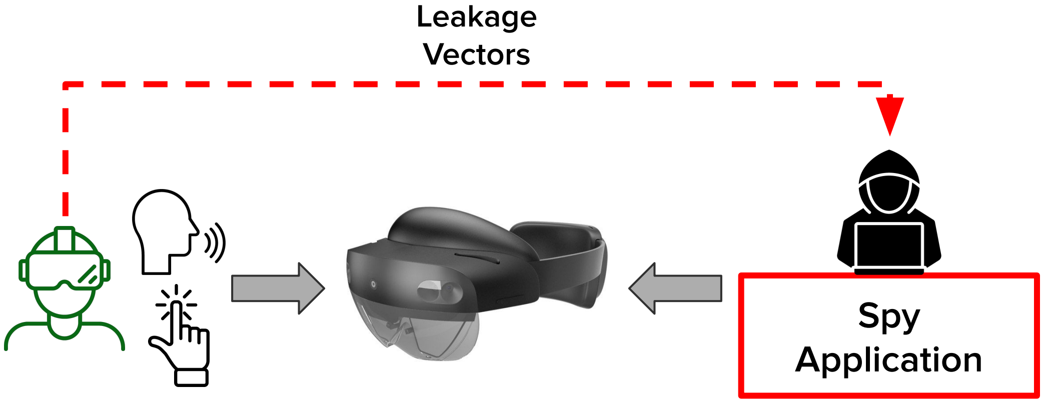
Leakage Vectors



Attack Overview

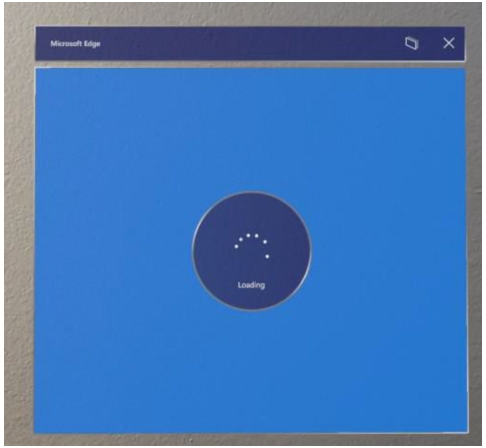
- Demonstrate three classes of attacks
  - Scenario 1. Spying on **user interactions** (Attack 1, 2, 3)

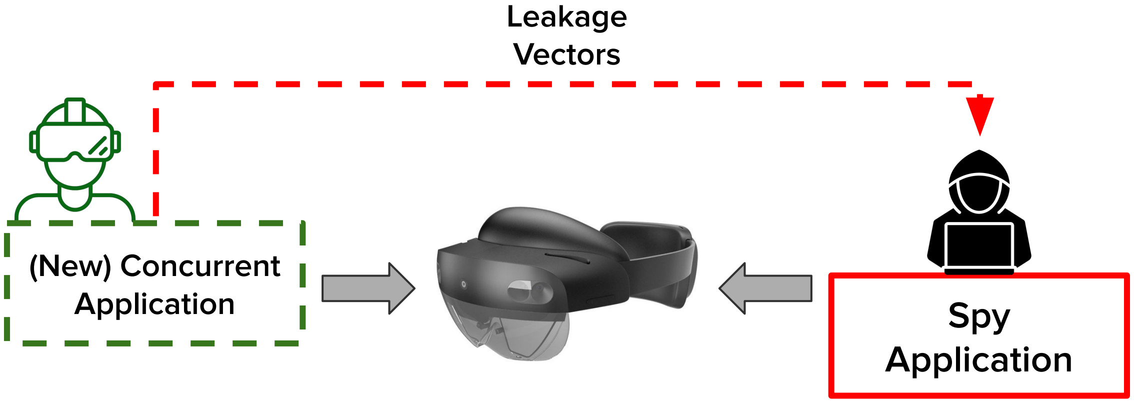




Attack Overview

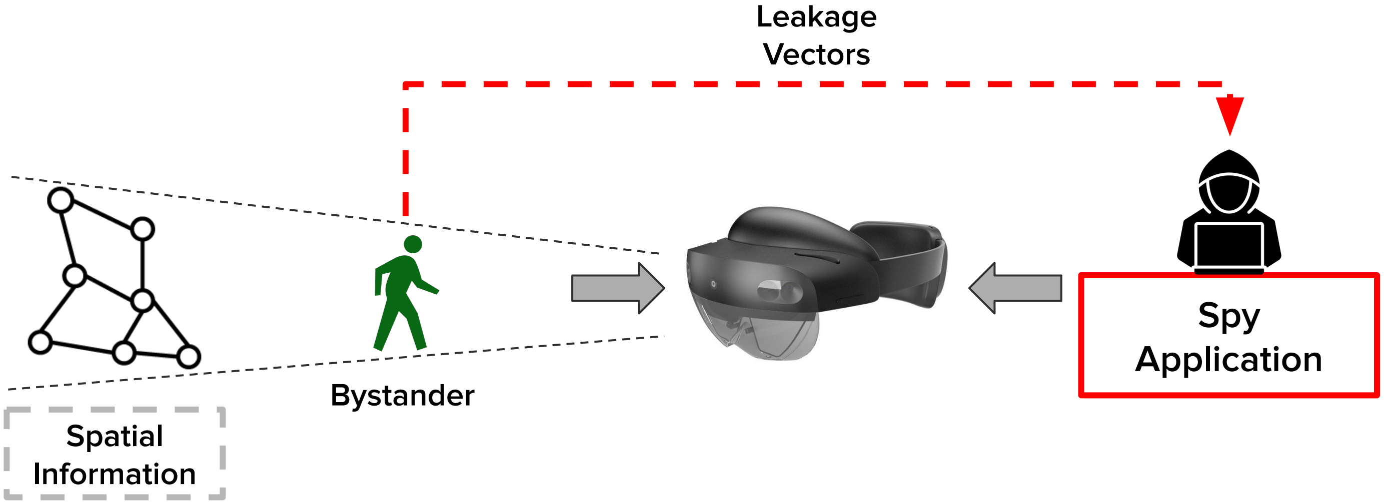
- Demonstrate three classes of attacks
  - Scenario 1. Spying on user interactions (Attack 1, 2, 3)
  - Scenario 2. Spying on **concurrent applications** (Attack 4)



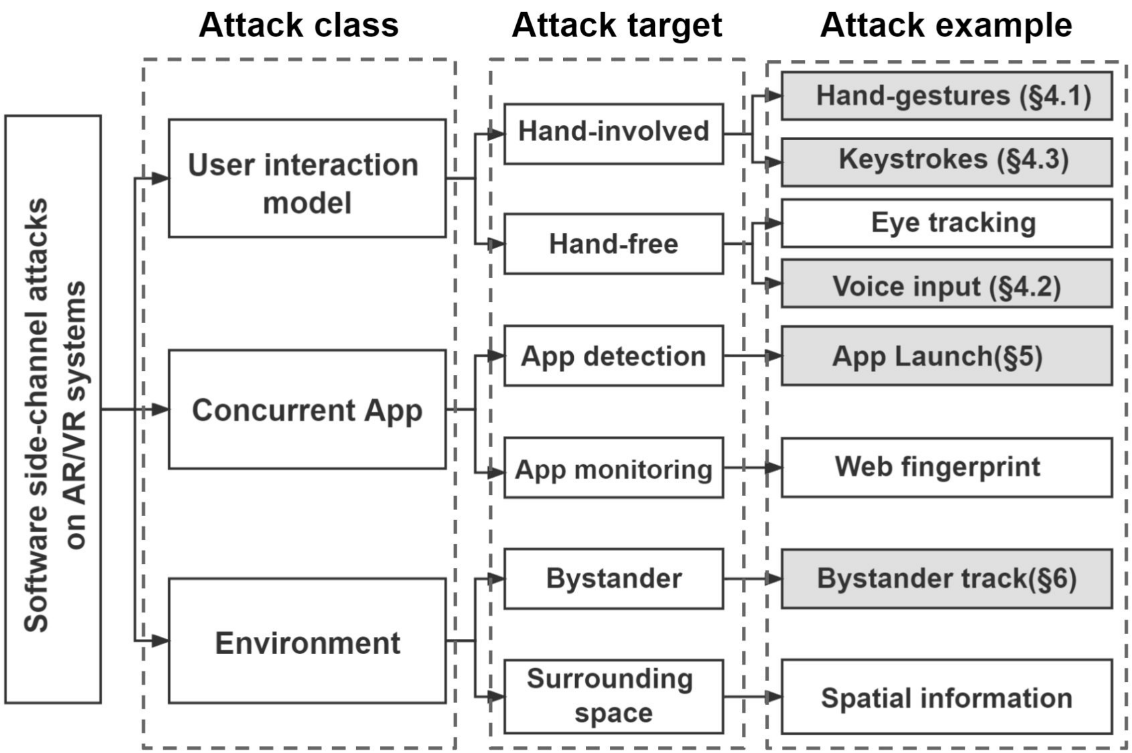


Attack Overview

- Demonstrate three classes of attacks
  - Scenario 1. Spying on user interactions (Attack 1, 2, 3)
  - Scenario 2. Spying on concurrent applications (Attack 4)
  - Scenario 3. Spying on the **real-world (AR) / virtual (VR) environment** (Attack 5)



Attack Overview

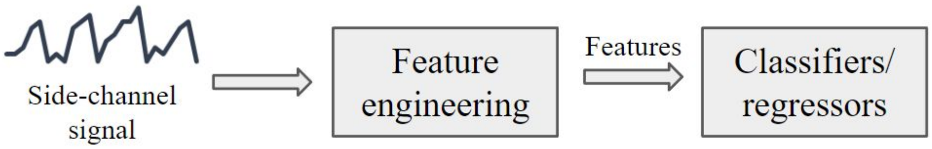


Experimental Setup

- Two representative headsets
  - Microsoft Hololens 2 (AR) - Windows XR SDK
  - Meta Quest 2 (VR) - Oculus XR SDK
- Spy app implemented with both Unity & Unreal Engine
  - Runs as a normal user-space application in the background
- 10 volunteers

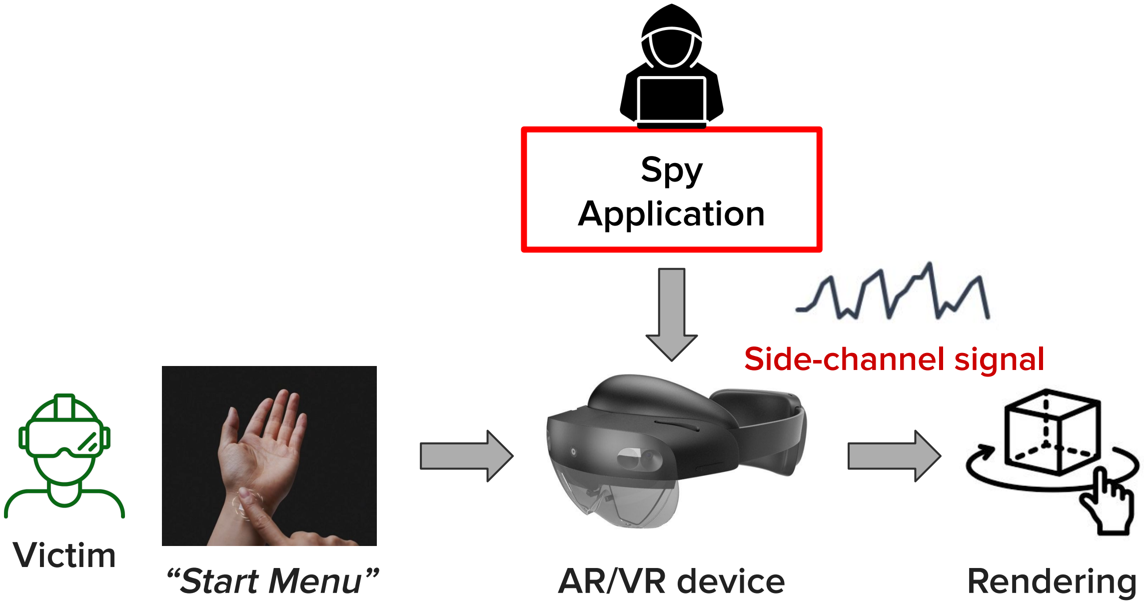
Attack Workflow

1. Record side-channel leakages from the malicious app
   - Training / Testing = 80 / 20%
2. Extract & rank useful statistical features from the time-series data
3. Train classifier candidates for inference attack
   - K-Nearest Neighbors (KNN)
   - Decision Tree (DT)
   - Random Forest (RF)
   - Light Gradient Boosting Machine (LightGBM)
   - Weighted Majority Rule Voting (Voting)



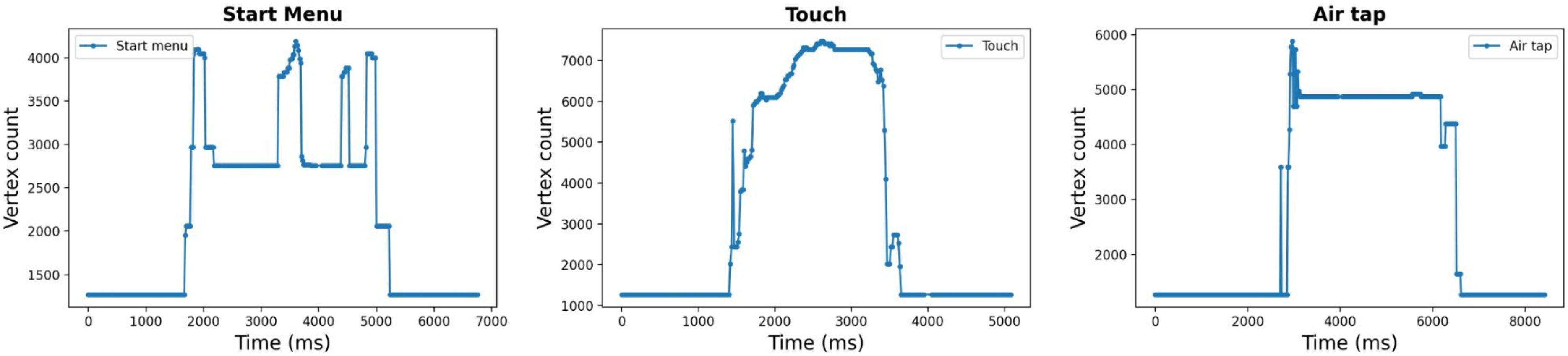
Attack 1: Hand gestures inference

- Victim: Interact with digital artifacts via hand gestures
- Spy: Collect signal patterns to infer victim's hand gestures

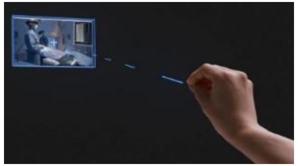


Attack 1: Hand gestures inference

- *“Vertex count”* performance counter
  - Number of vertices in existing 2D/3D scenes

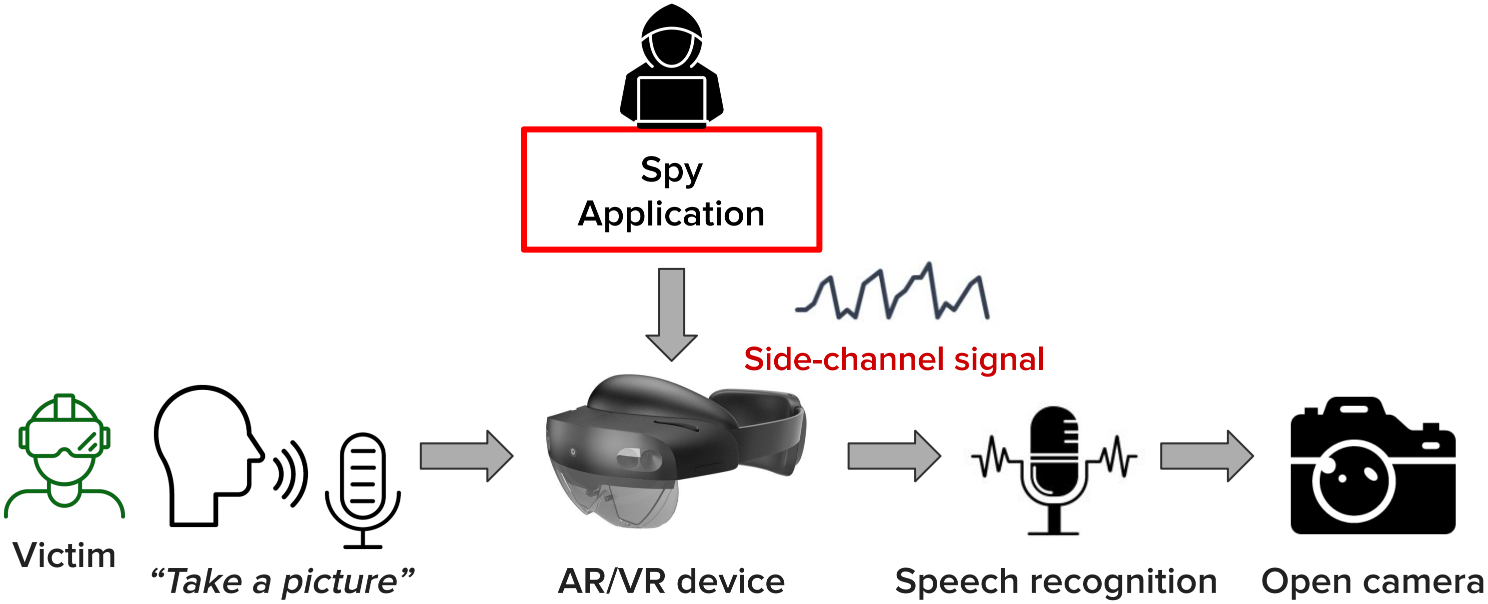


(a) (b) (c)



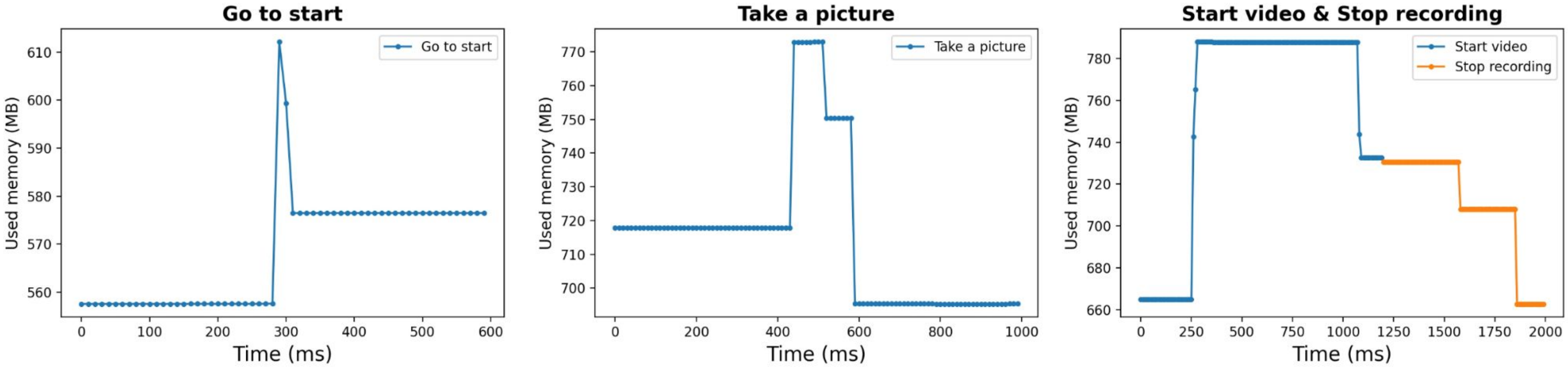
Attack 2: Voice commands inference

- Victim: Communicate with the headset via voice commands
- Spy: Collect signal patterns to infer victim's voice commands



Attack 2: Voice commands inference

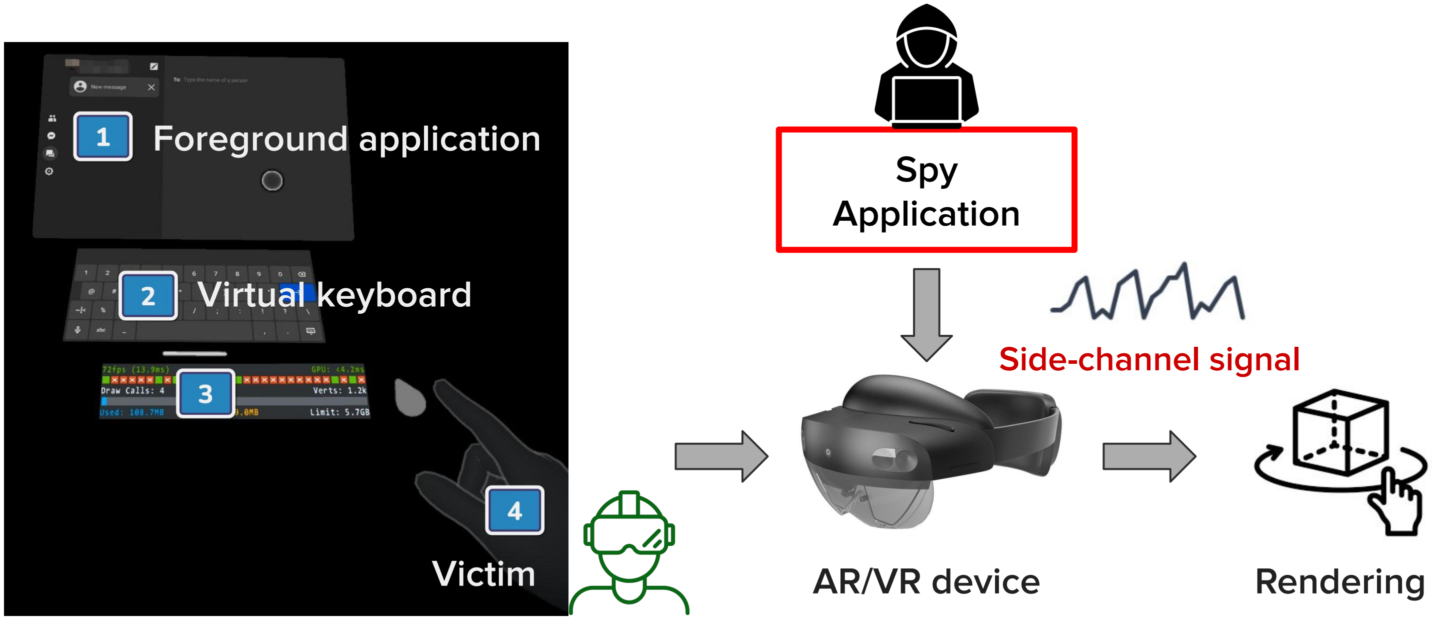
- *“AppMemoryUsage”* API
    - Current memory usage of **spy program***



(a) (b) (c)

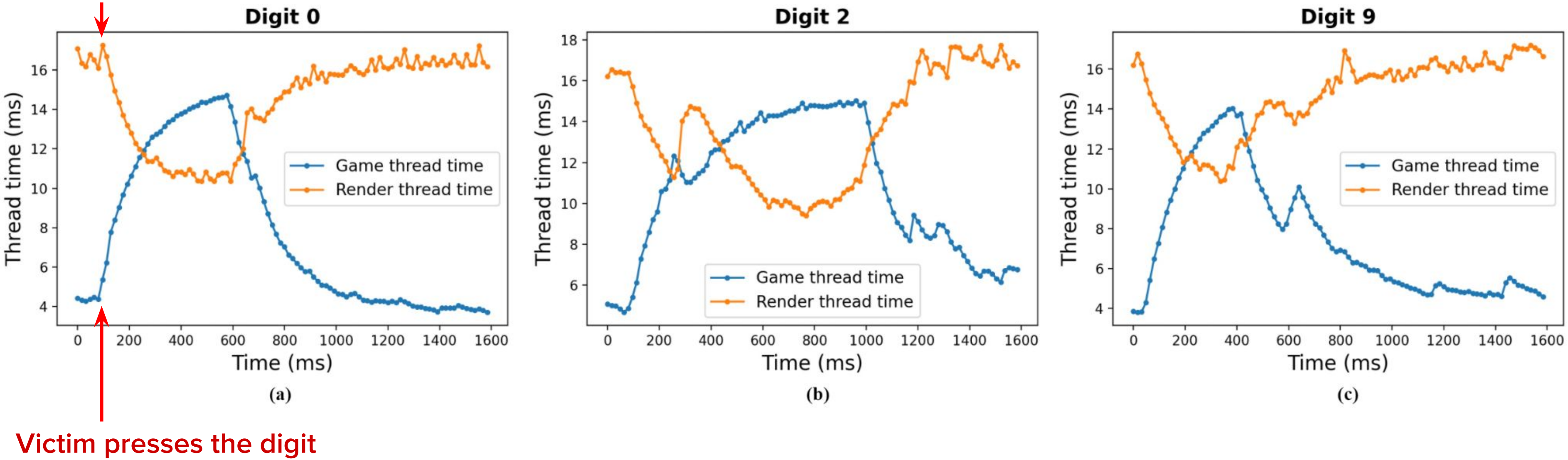
Attack 3: Keystroke monitoring

- Victim: Enters keystrokes through virtual keyboard
- Spy: Monitors performance counters to infer digit inputs



Attack 3: Keystroke monitoring

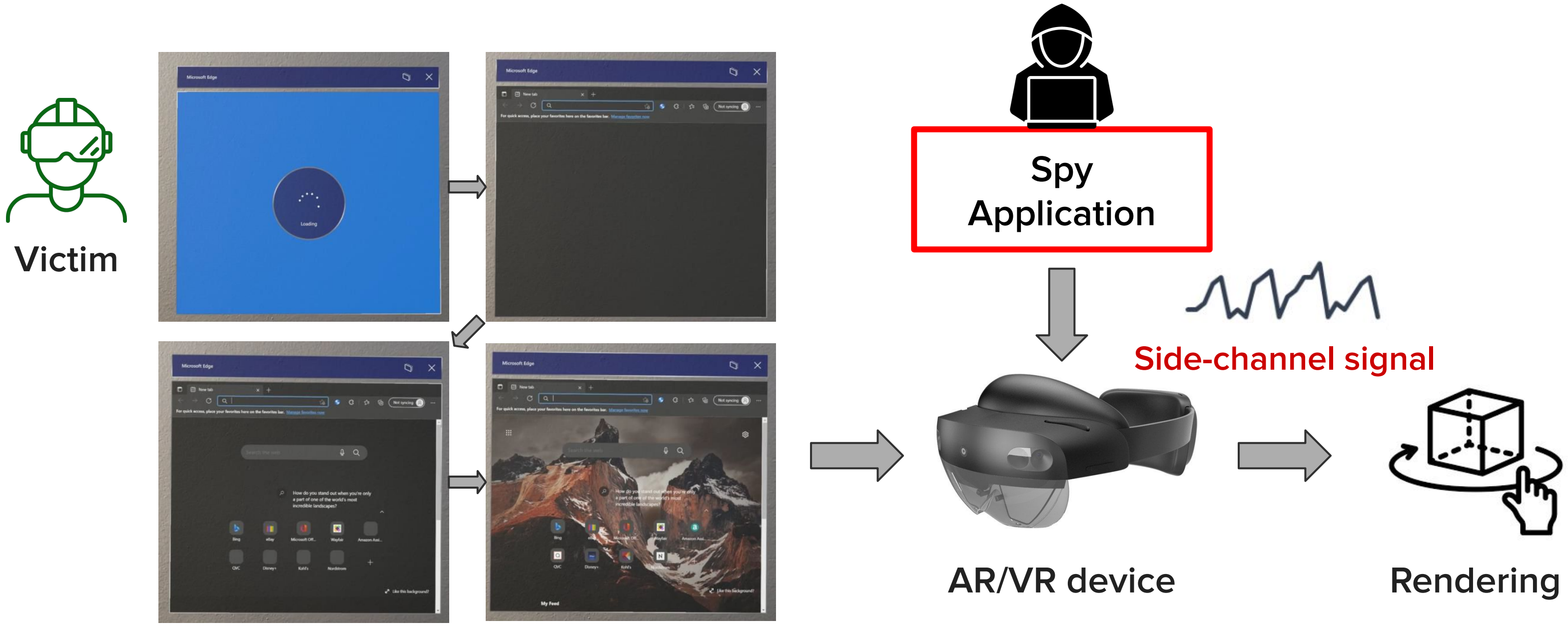
- *"Game thread time"* & *"Render thread time"* performance counter
    - Execution time of game thread & render thread



(a) (b) (c)

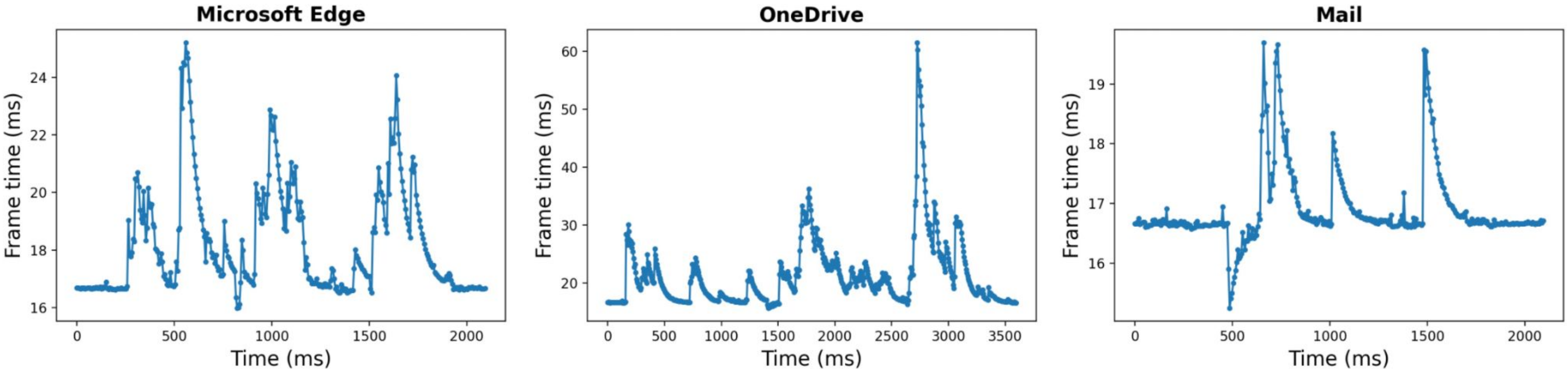
Attack 4: Concurrent app fingerprinting

- Victim: Launches a concurrent application on the AR/VR device
- Spy: Monitors performance counters to identify launched application



Attack 4: Concurrent app fingerprinting

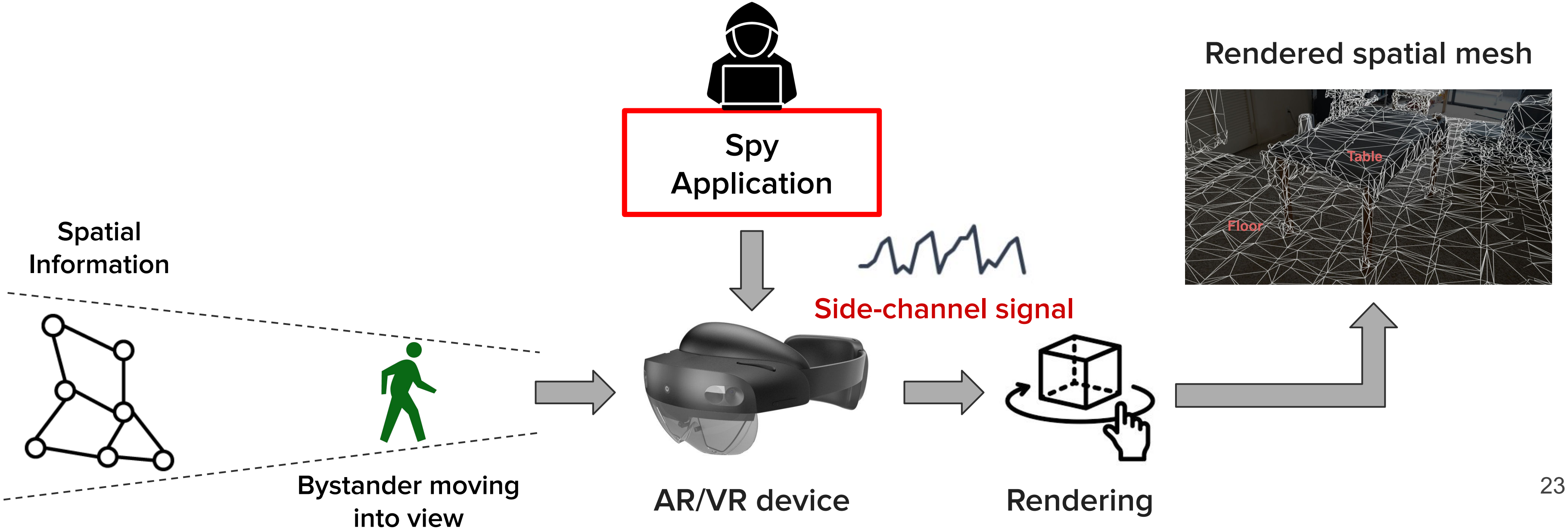
- *“Frame time”* performance counter
    - Time taken between two consecutive frames



(a) (b) (c)

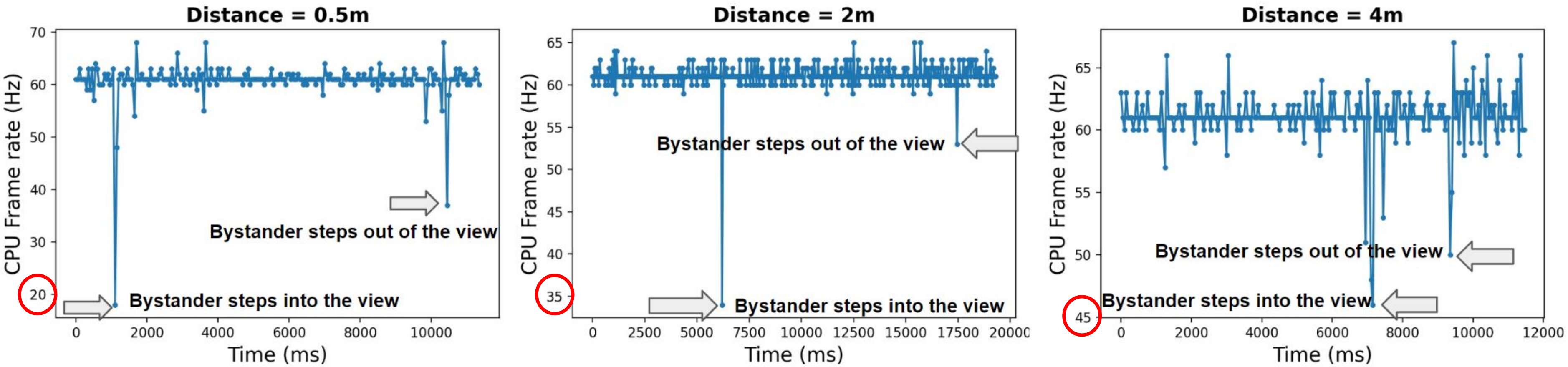
Attack 5: Bystander ranging

- Victim: Bystander steps into the field of view of an AR/VR device
- Spy: Profile leakage vectors to infer bystander distance
  Render spatial mesh of surrounding environment*



Attack 5: Bystander ranging

- *"CPU frame rate"* performance counter
    - CPU time taken between two consecutive frames on the main thread
- The closer the bystander is, the bigger frame rate drop occurs



(a) (b) (c)

Evaluation

- All classifiers reach 89.2% ~ 93.9% correctness (F1 score)
  - Example for Attack 1: Hand gestures inference

	Hololens 2			Quest 2		
	F1	**Prec**	**Rec**	**F1**	**Prec**	**Rec**
KNN	53.6	55.4	54.2	57.9	58.3	58.8
DT	80.0	80.5	80.0	91.3	91.7	91.3
RF	86.6	86.6	86.7	**93.7**	93.8	93.7
LightGBM	84.7	86.7	85.0	89.0	91.9	90.0
Voting	**89.2**	89.3	89.2	91.3	91.9	91.3

Table 3: Hand gesture inference performance: F1 (%), Precision (%), and Recall (%) on Hololens and Quest.

Evaluation

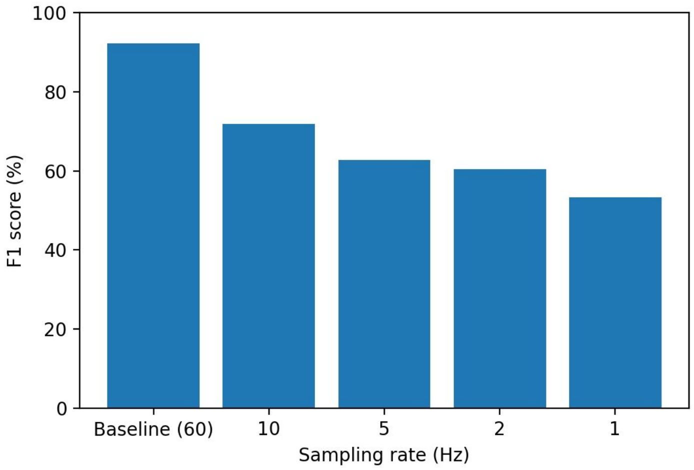
- Most relevant features may differ across devices
  - Example for Attack 1: Hand gestures inference

	Features	
	Hololens 2	**Quest 2**
CPU frame rate	**approximate_entropy**, sample_entropy, permutation_entropy	median
Number of draw calls	benford_correlation	minimum, quantile
GPU frame rate	**approximate_entropy**, sample_entropy, permutation_entropy	root_mean_square
AppMemoryUsage	maximum, abs_energy	**sum_values**, **mean**, **root_mean_square**, abs_energy, c3
Vertex count	**benford_correlation**	minimum

Table 2: Top 10 features for classifying hand gestures on the Hololens and Quest (top 3 features are bolded).

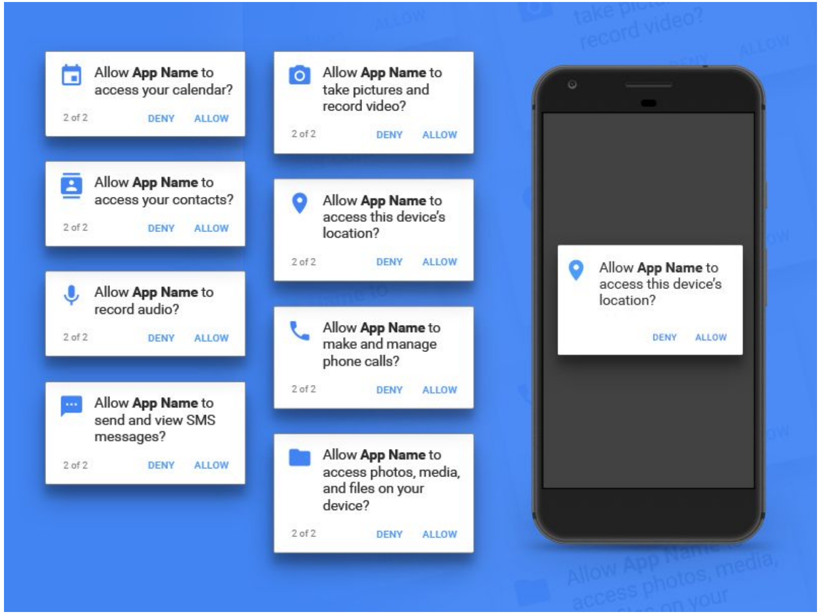
Defense

- Access control on APIs & performance counters
    - Completely block access to potentially leaky APIs & counters (impractical)
    - Limit the precision or rate of performance counters



Defense

- Access control on APIs & performance counters
  - Completely block access to potentially leaky APIs & counters (impractical)
  - Limit the precision or rate of performance counters
  - Permissions-based system



Defense

- Access control on APIs & performance counters
    - Completely block access to potentially leaky APIs & counters (impractical)
    - Limit the precision or rate of performance counters
    - Permissions-based system
- Monitor abnormal monitoring or contention
    - False positives and overheads

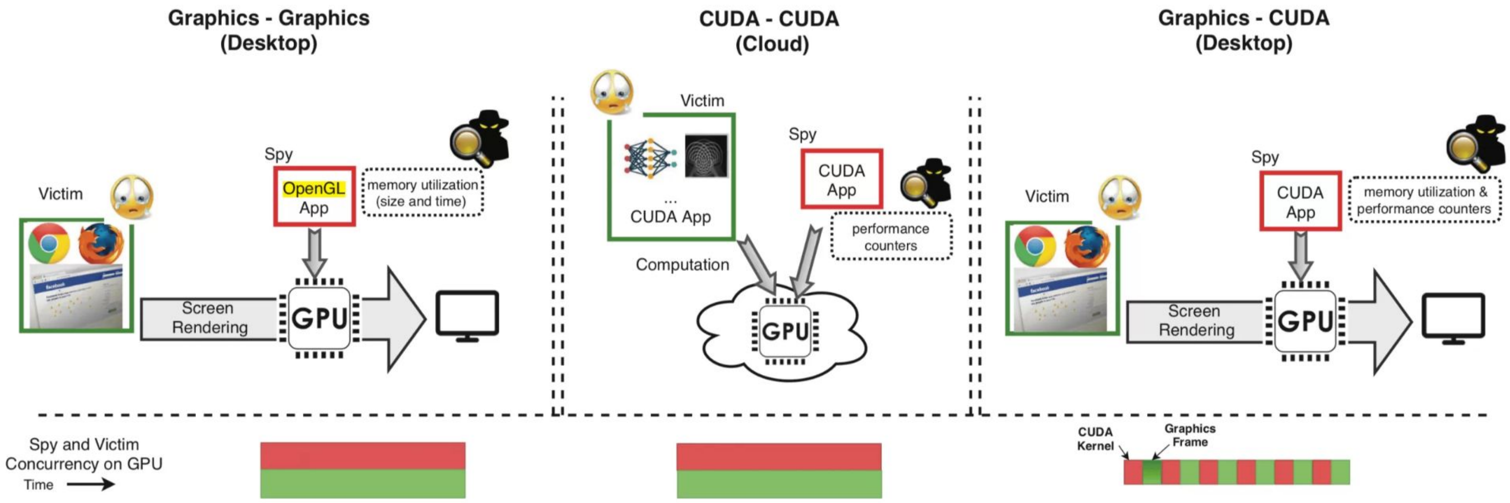
Conclusion

- A new software side-channel attack on AR/VR systems
  - First to use rendering performance counters
- Presented a taxonomy of software side-channel attacks on AR/VR devices
- Demonstrated 5 end-to-end attacks against commercial AR/VR devices
- Suggested mitigations

- Future works?
  - Multi-user AR/VR systems
  - Better profiling systems for AR/VR

Limitation

- Only uses functionality exposed by high-level SDKs
  - There may exist low-level functionalities not exposed by high-level SDKs
- Attacks are simply variants of well-known side-channel attacks
  - Naghibijouybari et al., **Rendered Insecure: GPU Side Channel Attacks are Practical** [CCS'18]



Related Works

- Kohno et al., **Display Leakage and Transparent Wearable Displays: Investigation of Risk, Root Causes, and Defenses** (Microsoft Technical Report, 2015)
  - Headset display leakage to a bystander
- ◑ Ling et al., **I Know What You Enter on Gear VR** [CNS’19]
  - Infer keystrokes via video recording (stereo camera) or motion sensor readings (SW)
- Arafat et al., **VR-Spy: A Side-Channel Attack on Virtual Key-Logging in VR Headsets** [VR’21]
  - Infer keystrokes via Wi-Fi channel state information (CSI) waveform side-channel
- Reddy et al., **Hidden Reality: Caution, Your Hand Gesture Inputs in the Immersive Virtual World are Visible to All!** [Sec’23]
  - Infer keystrokes via video recording

Related Works

- Meyer-Lee et al., **Location-leaking through Network Traffic in Mobile Augmented Reality Applications** [IPCCC'18]
  - Location inference attacks on mobile AR apps by probing network traffic information
- Shi et al., **Face-Mic: inferring live speech and speaker identity via subtle facial dynamics captured by AR/VR motion sensors** [MobiCom'21]
  - Biometrics and content inference from user speech by motion sensor side-channels
- Luo et al., **HoloLogger: Keystroke Inference on Mixed Reality Head Mounted Displays** [VR'22], Slocum et al., **Going through the motions: AR/VR keylogging from user head motions** [Sec'23]
  - Infer keystrokes by head motion tracking (+ even when typing by hand)

Related Works

- Future works: Privacy for multi-user AR/VR systems
  - Nair et al., **Unique Identification of 50,000+ Virtual Reality Users from Head & Hand Motion Data** [Sec'23]
    - Identification attack on 50k+ VR users from biomechanical data
  - Nair et al', **Going Incognito in the Metaverse: Achieving Theoretically Optimal Privacy-Usability Tradeoffs in VR** [UIST'23]
    - Differential privacy to obscure sensitive attributes on demand, a.k.a. "VR Incognito Mode"
- Microarchitectural side-channel & hardware attacks
  - Low-level hardware performance counters in CPU/GPU
  - Transient execution vulnerabilities
  - Rowhammer attacks on DRAM
  - Bus/Port contention side channels, and many more…

Good Questions

- Considering the potential impact on legitimate applications (precision reduction...), how to balance security measures to avoid interference with legitimate AR/VR applications? Is it still feasible without a real impact? (**Valentin**)

- Automatic tool to detect misbehavior application running on AR/VR systems? (**Hobin**)

- Does architectural similarity guarantee the same side-channels? (**Dongok**)

Best Questions

- For smartphones, app permission management is partly left to the user. What about implementing a similar idea in AR/VR devices? What role can the users have in controlling access to sensitive data and functionalities on their own devices? (**Valentin**)

- There are many side-channel attacks by monitoring memory consumption or other measurements. In other words, what are the differences in the research except for the target? (**Hyeon**)

- If two or three of these situations occur at the same time, will it be possible to distinguish the user's behavior? (**Seungmin**)

Thank You!