33RD USENIX SECURITY SYMPOSIUM

That Doesn't Go There: Attacks on Shared State in Multi-User Augmented Reality Applications

Carter Slocum[1*], Yicheng Zhang[1*], Erfan Shayegani[1], Pedram Zaree[1],
Nael Abu-Ghazaleh[1], Jiasi Chen[2]

yzhan846@ucr.edu

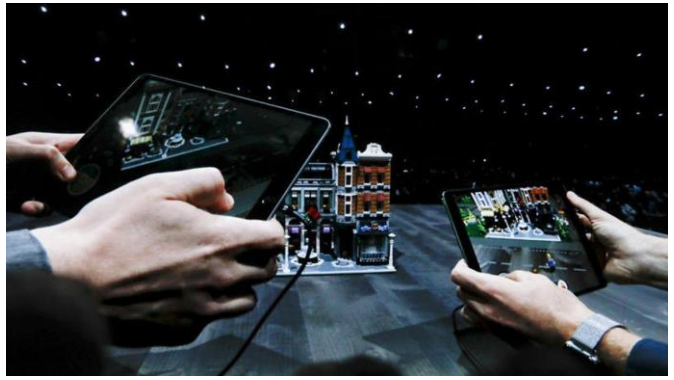
[1]*University of California, Riverside*

[2]*University of Michigan*

**Equal contribution*

Multi-user augmented reality apps

- A growing number of AR applications facilitate multi-user interactions with shared holograms



- These applications are supported by major industry players

ARCore by Google

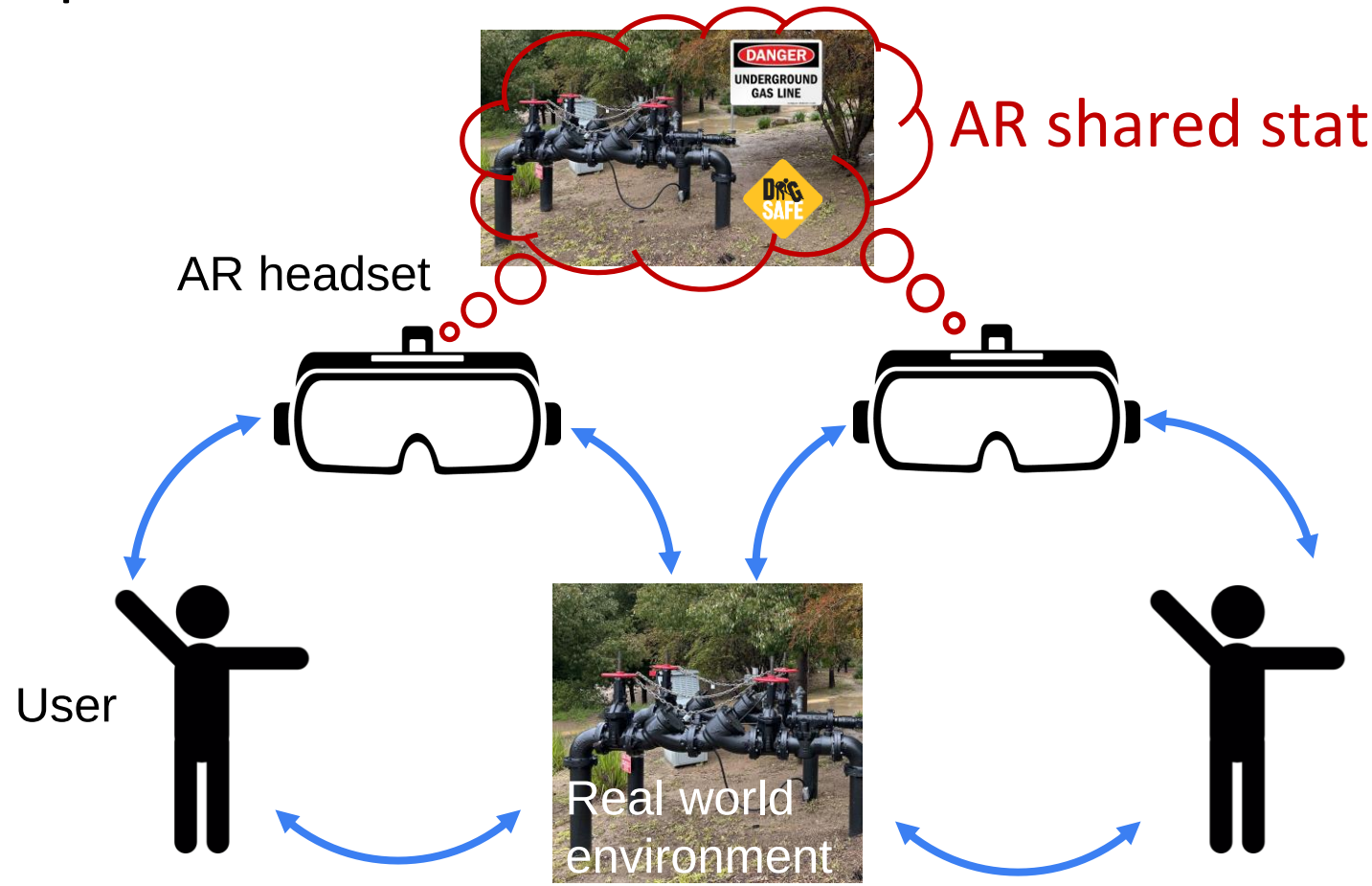
ARCore Geospatial API Challenge

Apple ARKit

What new security risks arise for multi-user AR?

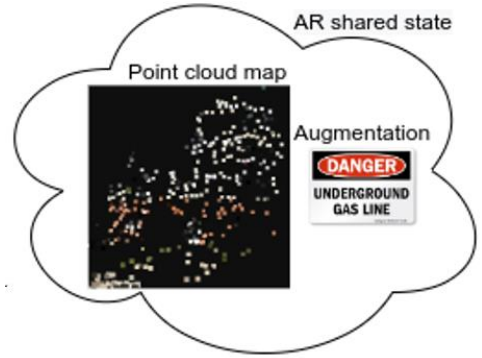
- AR devices sense the real world to create a shared AR experience

  → This exposes new attack surfaces!



Outline

- Background: “Shared State” in Augmented Reality.



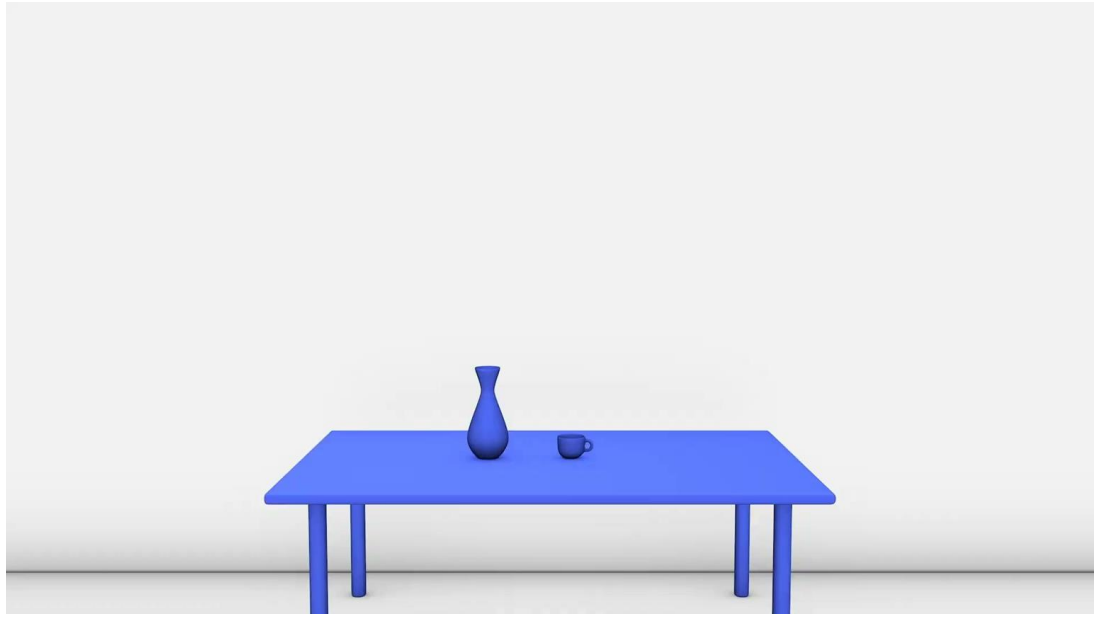
- Threat Model.

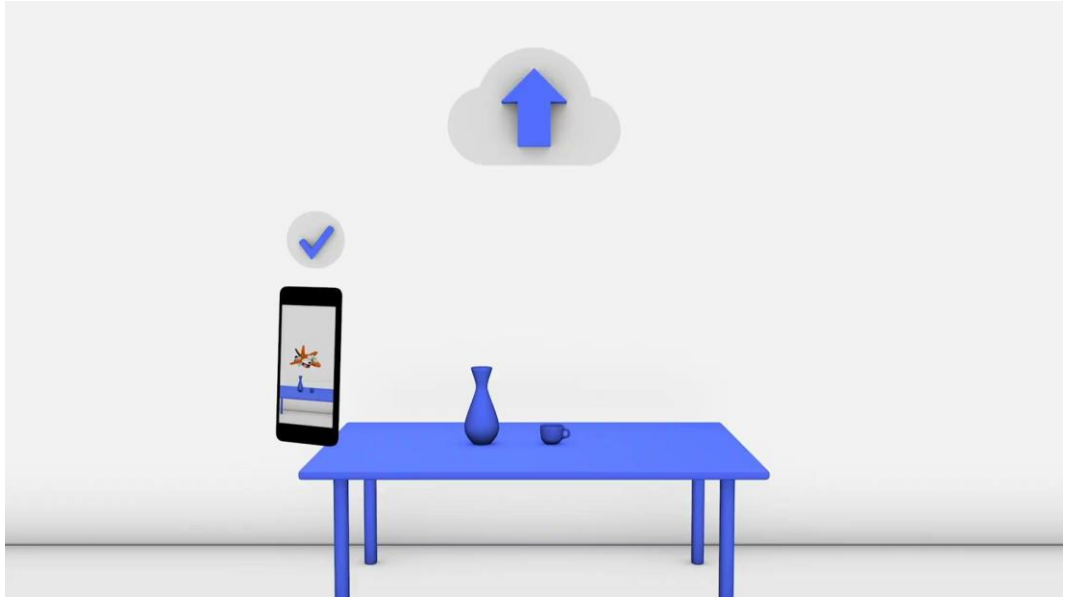
- Three Scenarios of Attacks.

- Mitigation.

Background on multi-user AR

- AR devices read/write to a shared state in order to view holograms



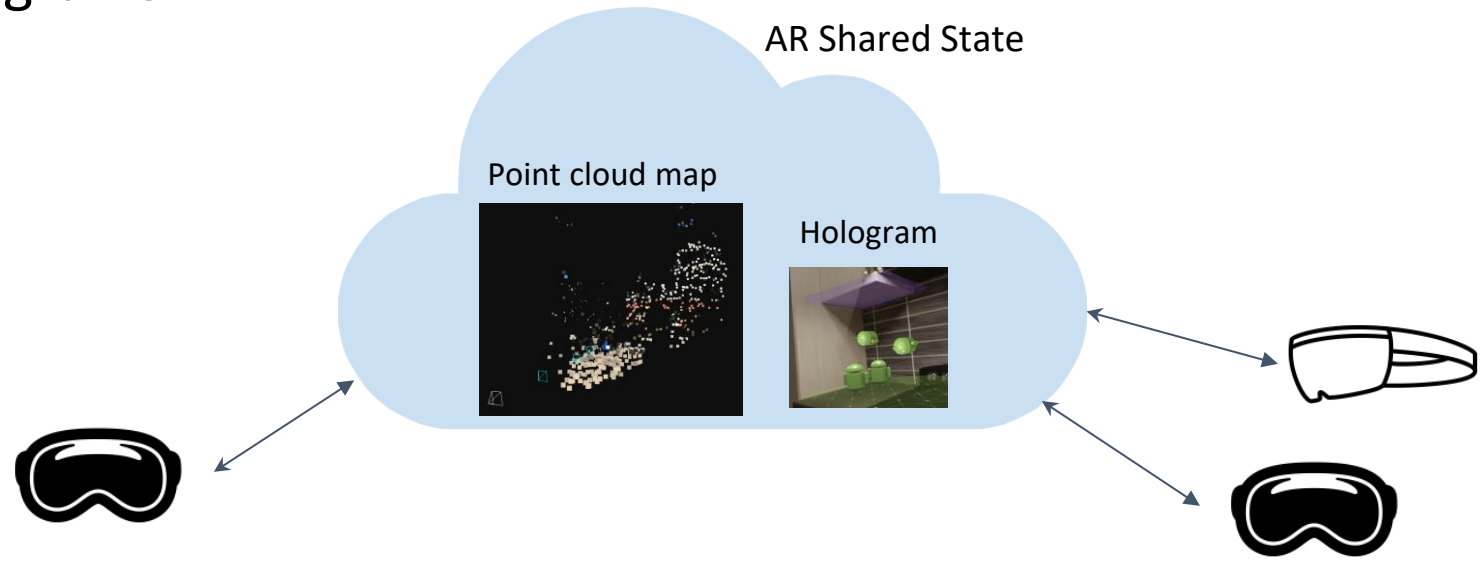


What if an attacker poisons the shared state?

Source: https://developers.google.com/ar/develop/cloud-anchors

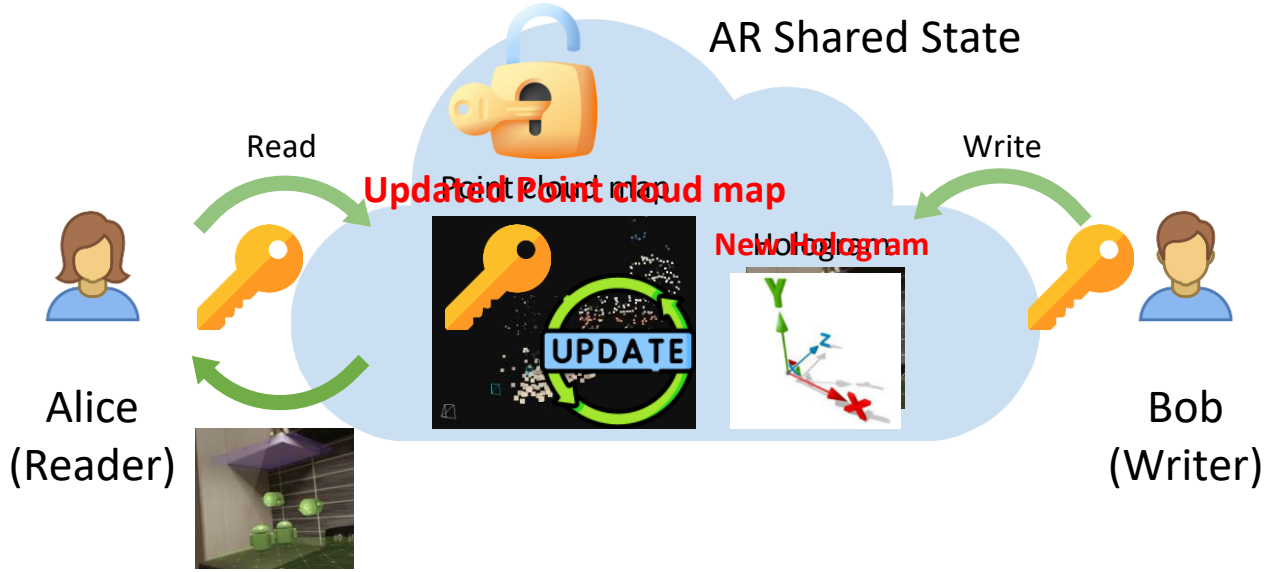
What is “Shared State” in augmented reality?

- Shared State: A collective set of information necessary for enabling interactive and consistent experiences among multiple users.
- Shared State contains:
  - Visual feature map of real world (point cloud map).
  - Holograms.



How do clients communicate with the Shared State?

- Read and write operations
  - Key = real-world environment (point cloud, IMU, GPS)
  - Value = hologram
- Examples
  - Google ARCore: `hostCloudAnchor`, `resolveCloudAnchor`



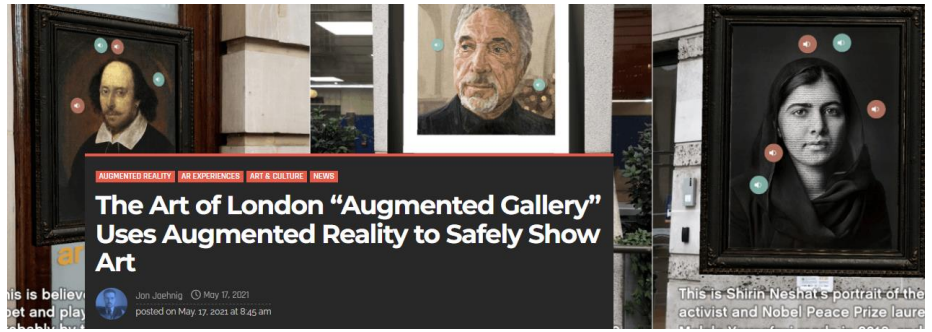
AR Shared State Taxonomy

- We examined commercial multi-user AR frameworks
- Propose the following taxonomy
  - Local: small local areas (e.g., indoor room)
  - Global: outdoor, world-scale (e.g., Pokemon Go)

	Non-curated	Curated
Local	**Scenario A: Cloud Anchor** *Keys*: camera, IMU *Attacks*: read, write	**Commercial scenario not found.** *Keys*: camera, IMU *Attacks*: read
Global	**Scenario C: Mapillary** *Keys*: camera, IMU, GPS *Attacks*: write	**Scenario B: Geospatial Anchor** *Keys*: camera, IMU, GPS *Attacks*: read

AR Shared State Taxonomy

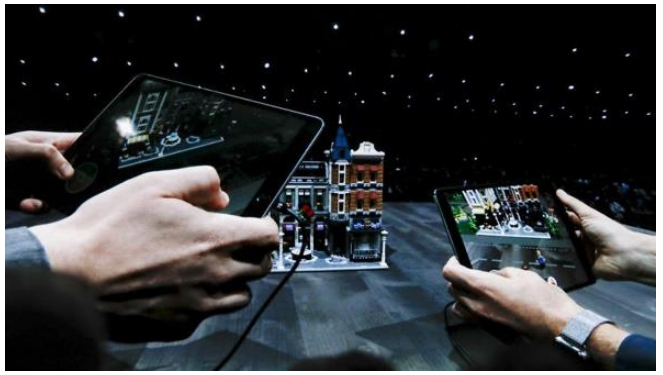
- Curated Shared State.
  - Curated maps are constructed by “curators”.
  - Only curator can write in shared state.
  - But non-curator can read from shared state.



Example of curated AR Shared State: Augmented art gallery

- Non-curated Shared State.
  - All users are allowed to Read and Write in shared state.

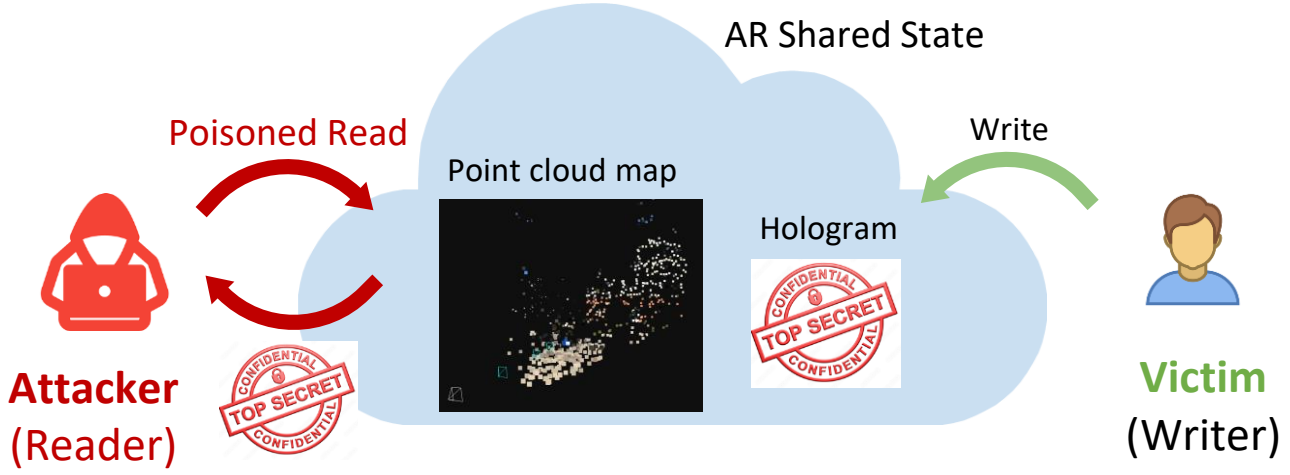
	Non-curated	Curated
Local	**Scenario A: Cloud Anchor** *Keys*: camera, IMU *Attacks*: read, write	**Commercial scenario not found.** *Keys*: camera, IMU *Attacks*: read
Global	**Scenario C: Mapillary** *Keys*: camera, IMU, GPS *Attacks*: write	**Scenario B: Geospatial Anchor** *Keys*: camera, IMU, GPS *Attacks*: read



Example of non-curated AR Shared State: On-the-fly game

Threat model: Read attack

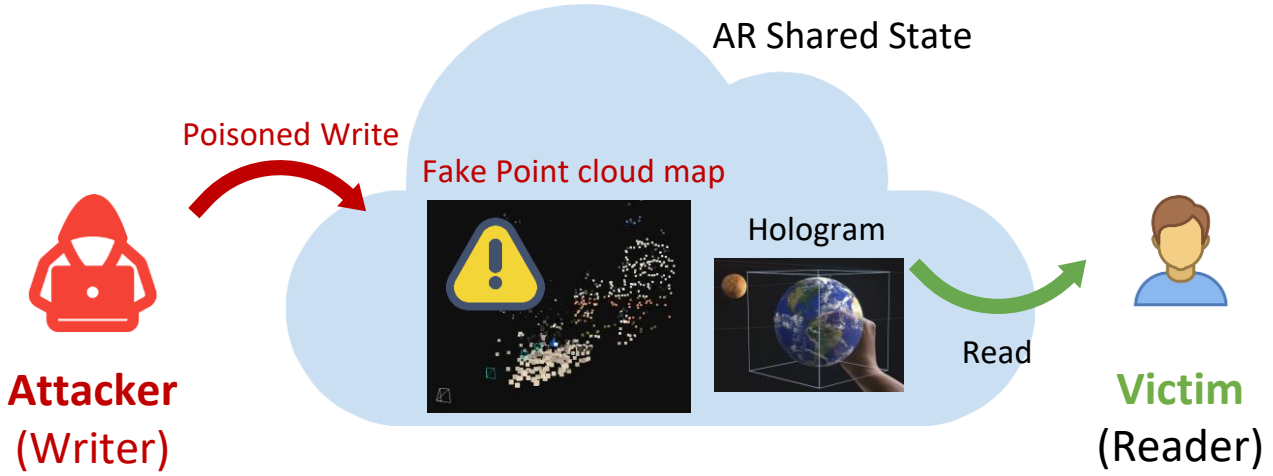
- An attacker participates in a multi-user AR application
  - Uses an unmodified AR application to access shared state
  - As a regular user, no special permissions
- Read attack:



Attacker extracts sensitive information stored within the shared state created by victim.

Threat model: Write attack

- Same threat model as Read attack
- Write attack:



Attacker manipulates shared state to deceive subsequent victim user!

Three Attack Scenarios

	Non-curated	Curated
Local	**Scenario A: Cloud Anchor** *Keys*: camera, IMU *Attacks*: read, write	**Commercial scenario not found.** *Keys*: camera, IMU *Attacks*: read
Global	**Scenario C: Mapillary** *Keys*: camera, IMU, GPS *Attacks*: write	**Scenario B: Geospatial Anchor** *Keys*: camera, IMU, GPS *Attacks*: read

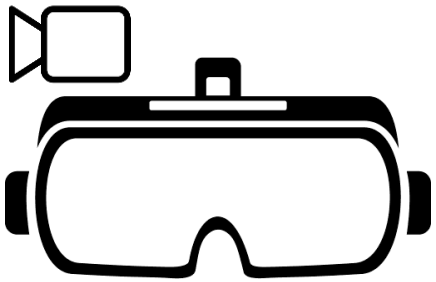
- Scenario A: Local, Non-Curated Shared State.
  - Platform: Google's Cloud Anchor API.
  - Attacker can read or write.

- Scenario B: Global, Curated Shared State.
  - Platform: Google's Geospatial API.
  - Attacker can only read.

- Scenario C: Global, Non-Curated Shared State .
  - Platform: Mapillary.
  - Attacker can read or write.

Scenario A: Remote read attack

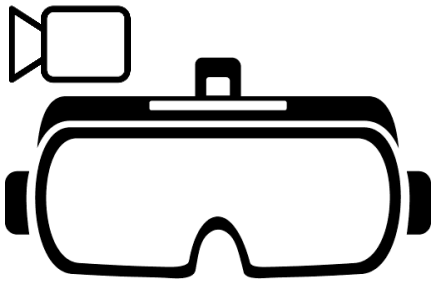
1. Attacker has control of own device
2. Show inputs to camera

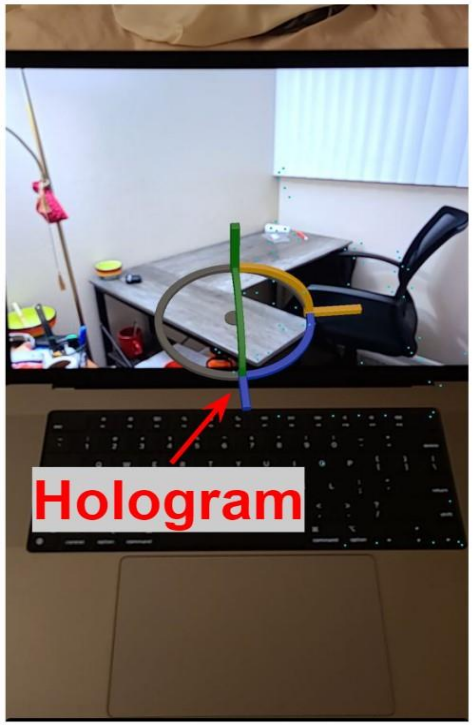


View hologram at physical location

Scenario A: Remote write attack

1. Attacker has control of own device
2. Show inputs to camera





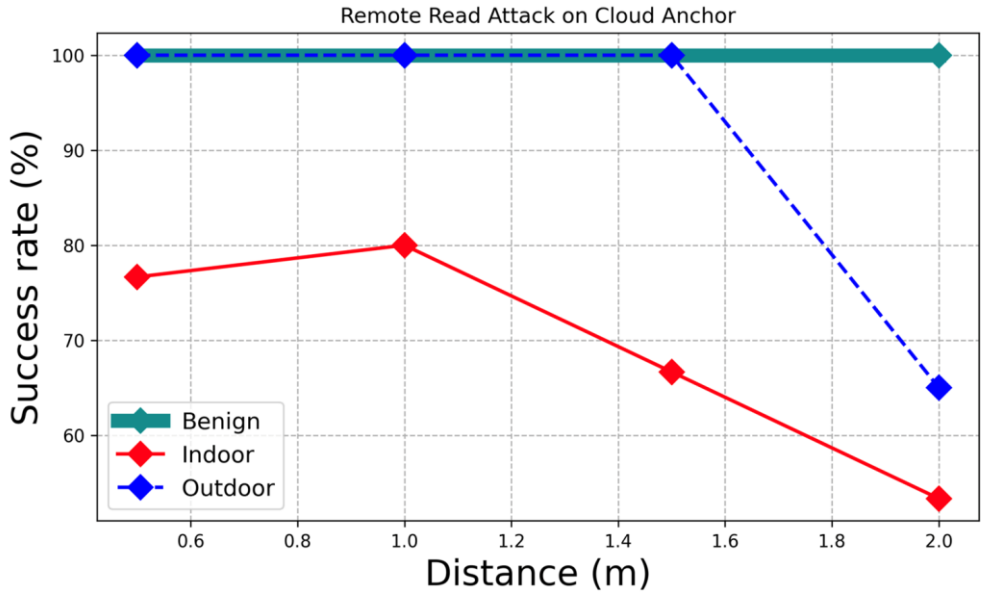
Write hologram at remote location

Scenario A: Evaluation

- Six different environments.
- Samsung Galaxy S20 Android phone with Google ARCore support.
- Good and robust success rate among three attacks.

Environment	Attack success rate	
	Static scene	Add clutter
Office desk	8/16	7/16
Bedroom desk	6/16	4/16
Bedroom bed	10/16	8/16
Outdoor garden	1/16	0/16
Outdoor BBQ	16/16	15/16
Outdoor pool	15/16	14/16

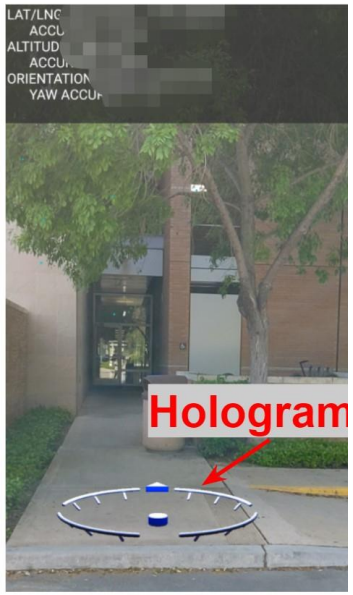
Remote Write Attack Success Rates



Effect of Distance on Remote Read Attack

Scenario B: Remote read attack

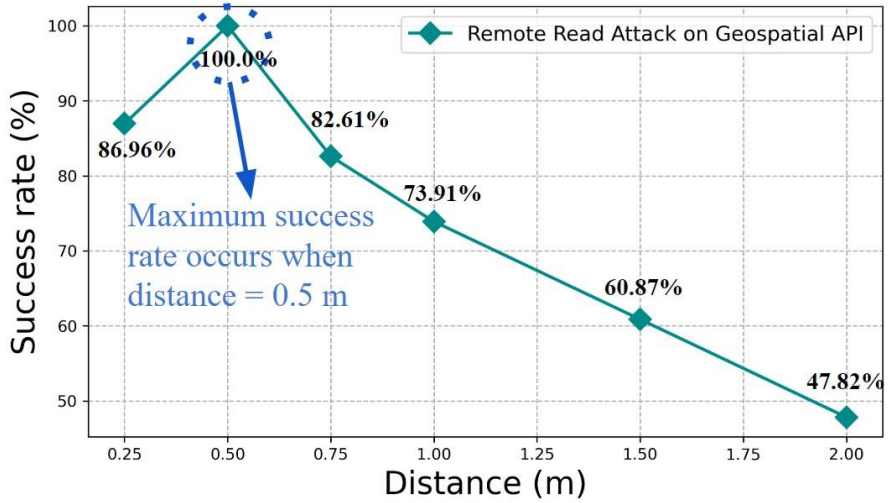
- Attacker reads a hologram from a remote location.
- Attacker deceives Google's Geospatial API
  - Fake camera: photograph of location
  - Fake GPS: GPS spoofing app



Write hologram at physical location

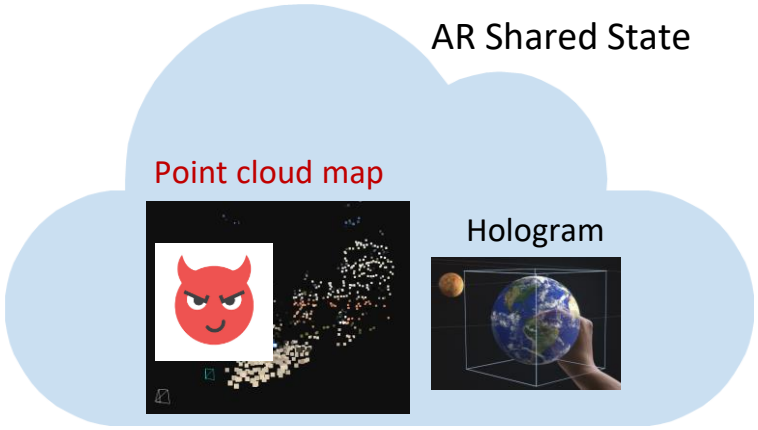
Scenario B: Evaluation

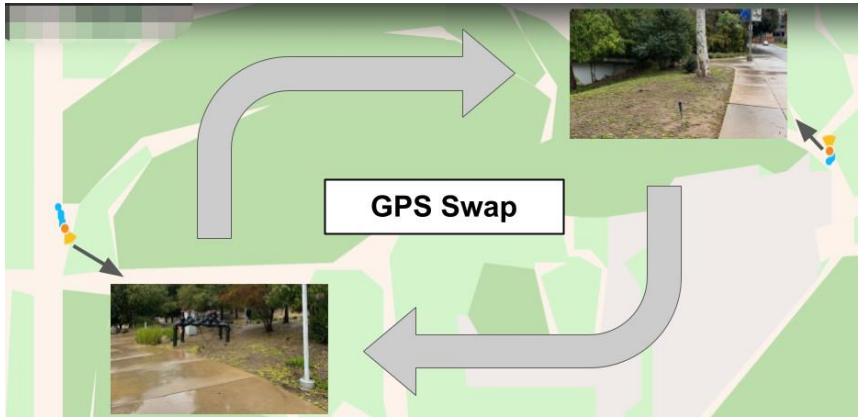
- 23 holograms at various locations within our campus.
- Samsung Galaxy S8 and the Samsung Galaxy S21 with Google Geospatial API support.
- Good and robust success rate through all locations.



Scenario C: Poisoned write

- Poisoned write to the Shared State's point cloud map
- Attacker deceives point cloud generation algorithms
  - Fake GPS: Swap GPS coordinates of two images sequences by editing image metadata
- Experiments done in a Mapillary sandbox with permission
  - No public users were affected





Attack 2 Preview: Example on Mapillary

No attack: Desired annotations

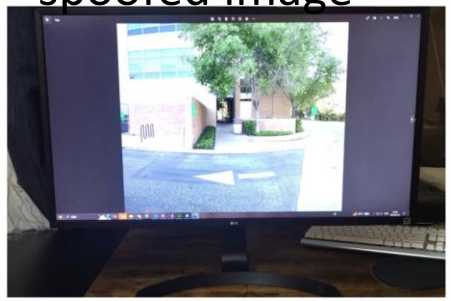
With attack: Annotations swapped

Dangerous scenario!

Mitigation Using Multi-Modal Sensors

- How to detect fake camera inputs?
- Idea: Use additional sensor modalities
  - AR devices equipped with depth sensor, Lidar, etc.

RGB camera of spoofed image



Depth camera of spoofed image

- How did we evaluate this defense?

  CNN: ResNet-18 network to detect spoofed images

  Dataset: 15 real scenes, 300 pairs of color and depth image of each scene

  Same process to collect images in front of monitor showing spoofed image

  Training: 12 scene for training; 3 scenes for test

  Precision: 84.22%

- Other potential mitigations
  - Clean-Slate System Design
  - Real Space Security
  - Local Moderators

Summary

Paper

AR devices sense information about a common reality

↓

Info shared across apps and systems

↓

Attack opportunities!

Demo

defense

- Multi-user application attacks on shared world state (**First**)
  - Read/write holograms despite not being physically present
  - Demonstrated on 3 commercial AR frameworks

- Easy mitigation strategies (e.g., multi-modal sensing) are effective
  - But require additional sensors and compute

Thank you!
Questions?