Stateful Least Privilege Authorization for the Cloud

Leo Cao*, Luoxi Meng*, Deian Stefan, Earlence Fernandes

August 15, 2024

UC San Diego

*Equal contribution.

Cloud Authorization

Alice

① "Create a Calendar event."

② "Give me a token to c
Alice's Calendar even

zoom

Sign in with Google

zoom

Zoom wants access to your Google Account

lumeng@ucsd.edu

When you allow this access, Zoom will be able to

View and edit events on all your calendars.
Learn more

Make sure you trust Zoom

You may be sharing sensitive info with this site or app. You can always see or remove access in your Google Account.

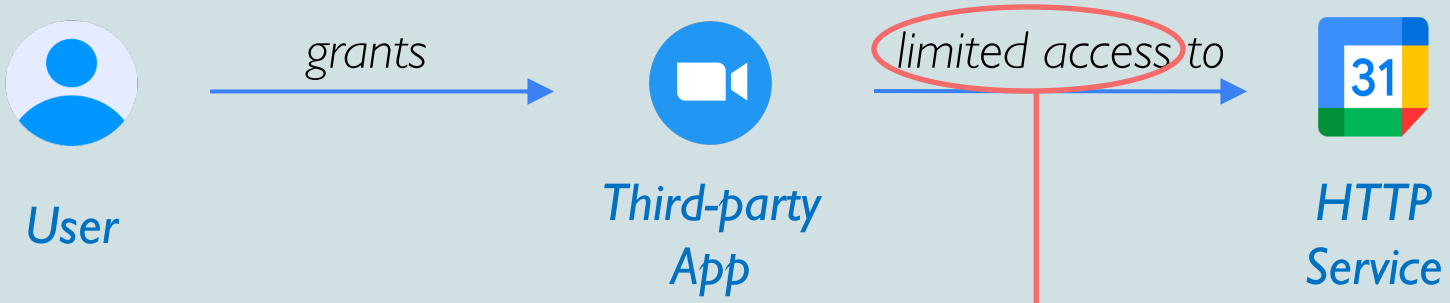
Learn how Google helps you share data safely.

See Zoom's Privacy Policy and Terms of Service.

Cancel | Continue

Cloud Authorization

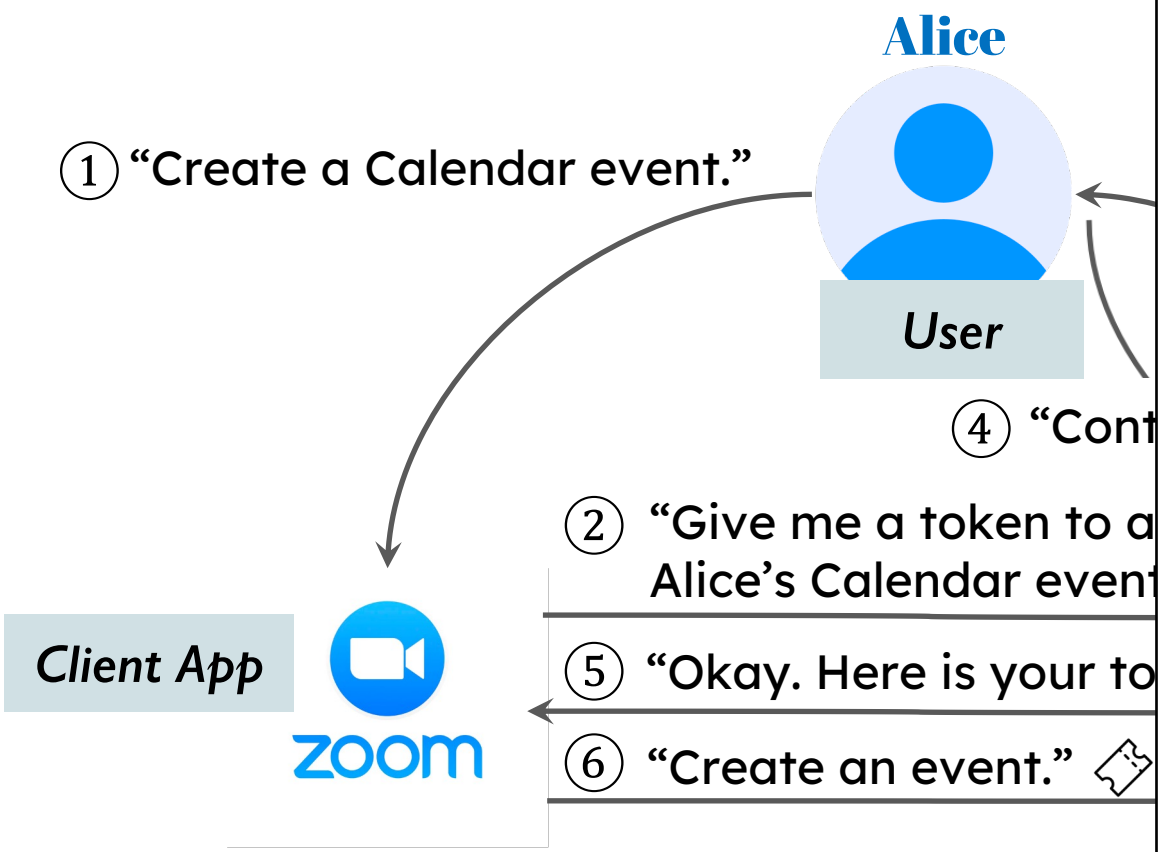
OAuth:

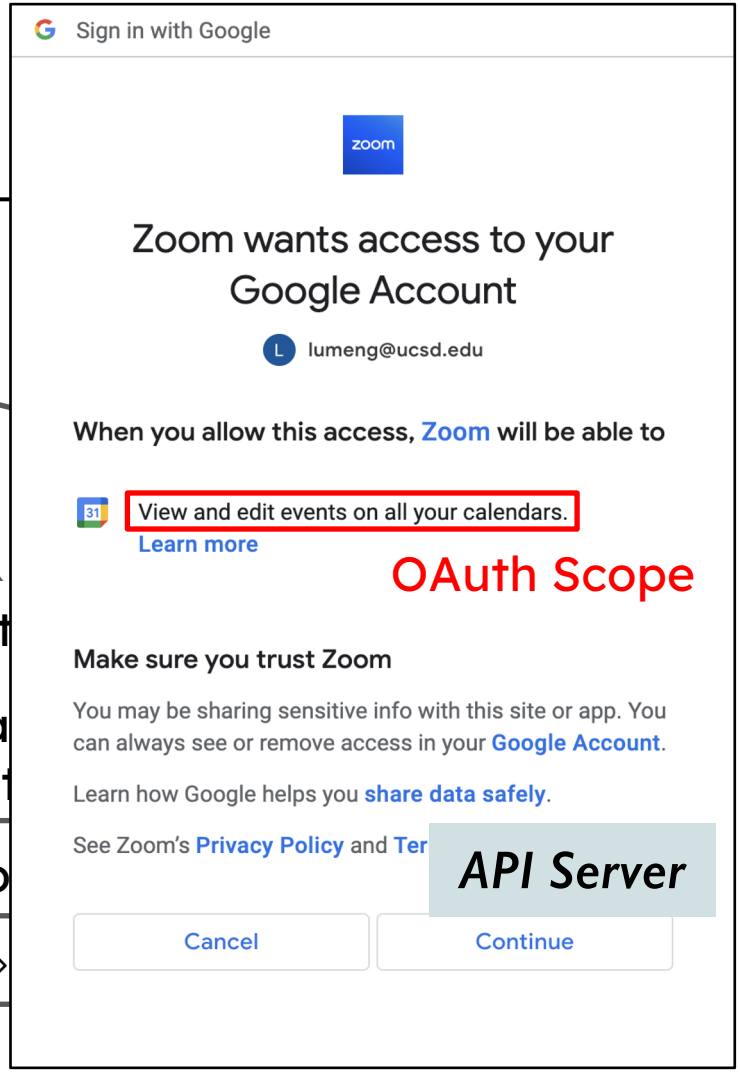


without revealing the user's credentials.

OAuth Scope: The set of permissions requested by the third-party application.

OAuth


Alice
User
① "Create a Calendar event."
④ "Cont
② "Give me a token to a
Alice's Calendar event
Client App
zoom
⑤ "Okay. Here is your to
⑥ "Create an event."


Sign in with Google
zoom
Zoom wants access to your Google Account
lumeng@ucsd.edu
When you allow this access, Zoom will be able to
View and edit events on all your calendars.
Learn more
OAuth Scope
Make sure you trust Zoom
You may be sharing sensitive info with this site or app. You can always see or remove access in your Google Account.
Learn how Google helps you share data safely.
See Zoom's Privacy Policy and Ter
API Server
Cancel
Continue

Token Leakage & Abuse

Bearer token gives access to any token bearer.

/ Blog Menu Try GitHub Copilot Contact sales

Security

Security alert: Attack campaign involving stolen OAuth user tokens issued to two third-party integrators

On April 12, GitHub Security began an investigation that uncovered evidence that an attacker abused stolen OAuth user tokens issued to two third-party OAuth integrators, Heroku and Travis-CI, to download data from dozens of organizations, including npm. Read on to learn more about the impact to GitHub, npm, and our users.

The Register

Exposed Hugging Face API tokens offered full access to Meta's Llama 2

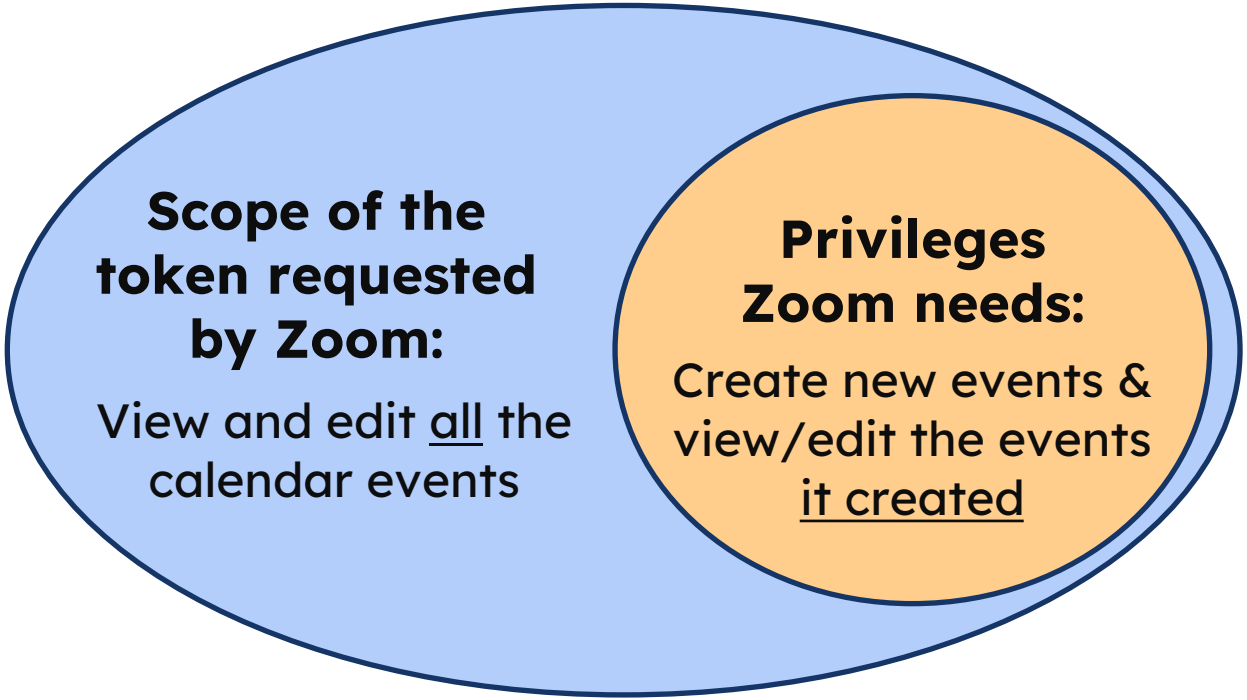
With more than 1,500 tokens exposed, research highlights importance of securing supply chains in AI and ML

#202781

407

Chained Bugs to Leak Victim's Uber's FB Oauth Token

Overprivilege of Tokens



Tokens are given more permissions than what they need.

Key Question:

How to minimize the token's privilege while retaining functionality?

Cause of Overprivilege #1: Server-Defined Permissions

Travis CI Blog Docs

`repo` Grants read and write access to code, commit statuses, collaborators, and deployment statuses for public and private repositories and organizations. We need this level of access because GitHub does not provide the `read:org` (read-only) scope for private repositories.

Client finds a "best match".
"Best match" ≠ "perfect match".

Server cannot anticipate every possible way a client uses their API.

Cause of Overprivilege #2: Statelessness

How to enforce that "Zoom can only create new events and view/edit the events it created"?

- Need information of which events are created by Zoom.

Event ID	Operations performed by Zoom
0x01	Insert, get
0x02	Insert

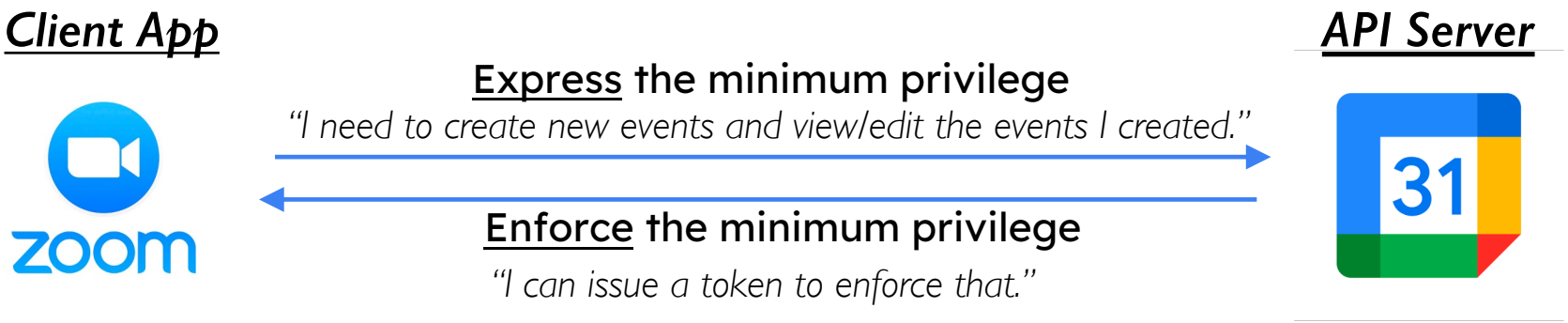
Indicates that the event is created by Zoom.

State: the operations that the client app has taken.

- *Not possible using current stateless machinery.*

Contributions

- We design a ***least privilege*** authorization model that



- We introduce two abstractions in cloud authorization:
  - ***Client-defined permissions*** w/ WebAssembly.
  - ***Statefulness,*** enabling a new class of least-privilege authorization policies.

Threat Model

- We assume that the client app developer and API server are trusted.

- The client app *itself* is vulnerable to security problems that can result in bearer tokens being stolen or leaked.
  - Phishing attacks
  - Infrastructure Misconfiguration
  - …

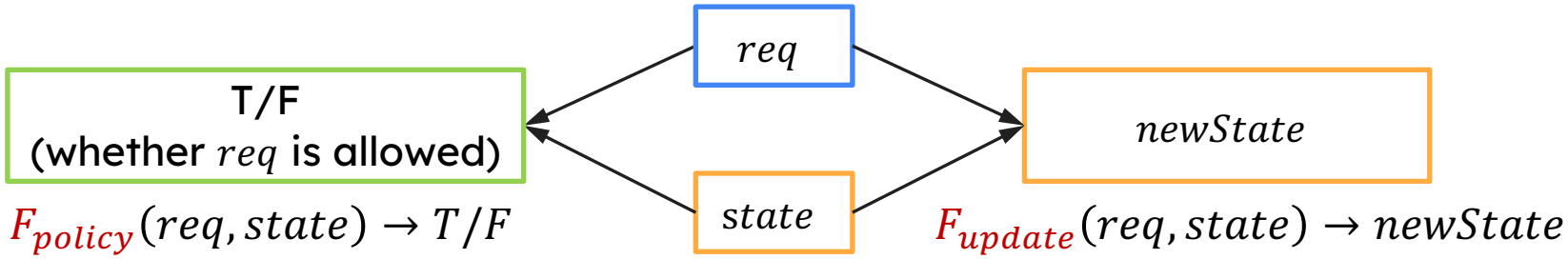
- **Security Goal:** Limit the abuse during the vulnerability window.

Core Insight:

Client app developer always knows the app's minimum privilege requirements.

Client-Defined Permissions

- Introducing ***state*** – a log of operations the client app has executed on the server.

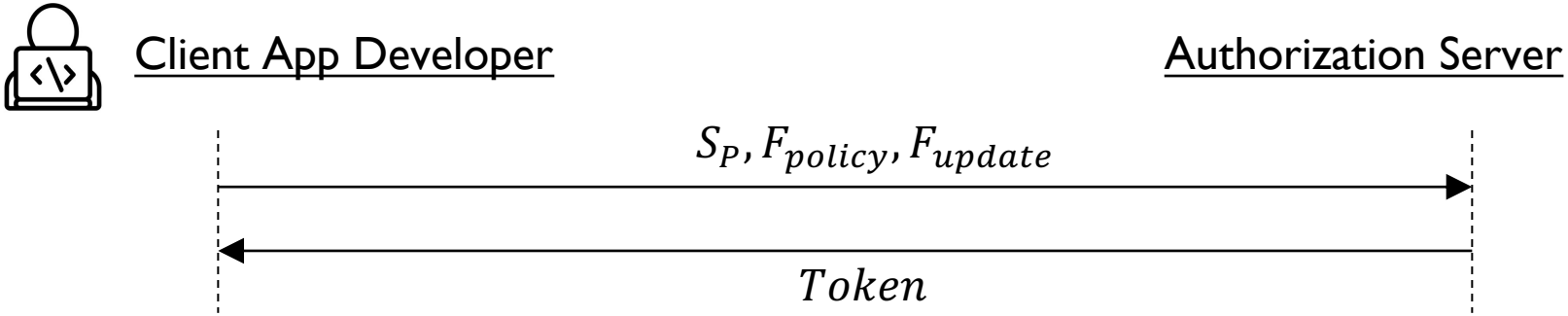


An attenuation policy that further limits the scope of bearer tokens.

A state updater that defines and updates the state.

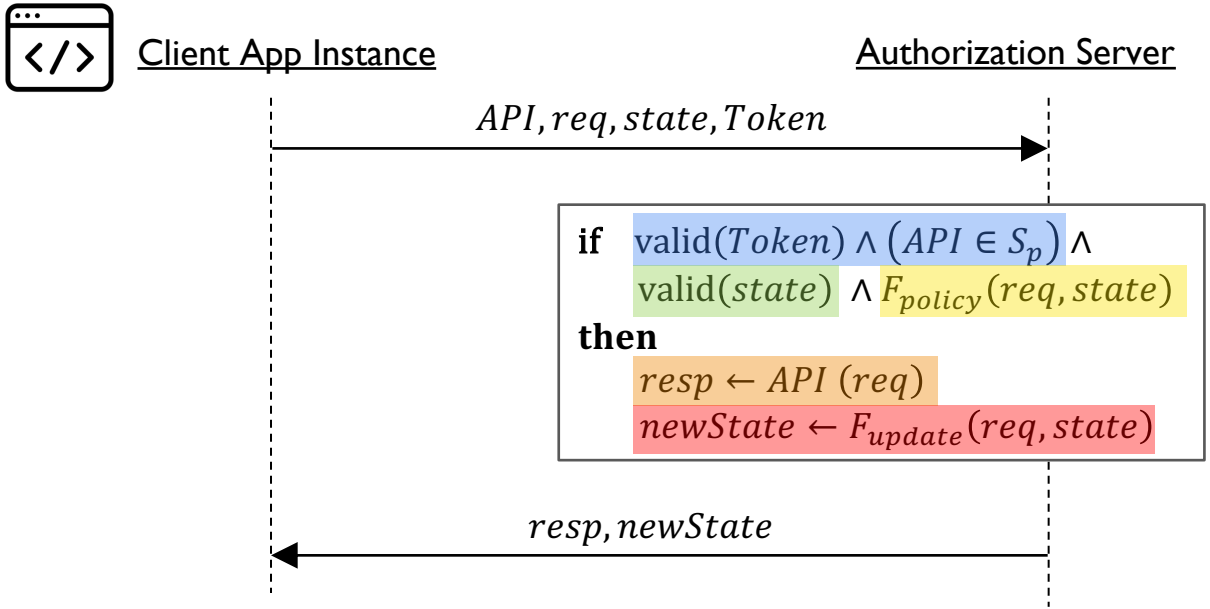
- Client app developer supplies F_{policy} and F_{update} in WebAssembly.
  - Flexibility
  - Portability
  - In-process sandboxing

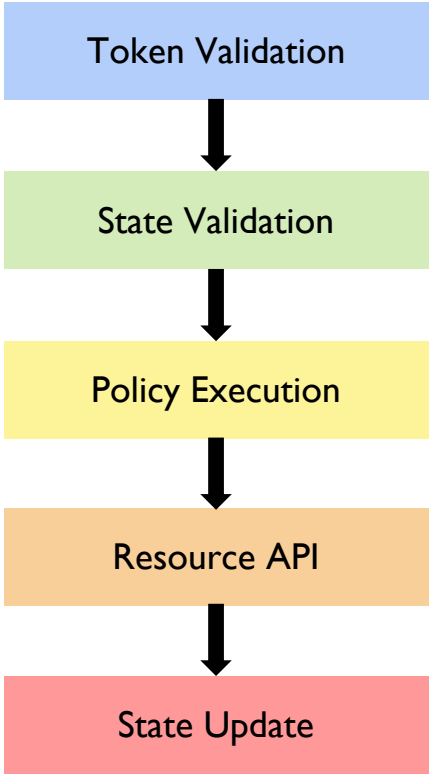
Stateful Authorization Model: Client Registration



S_p: server-define scope
F_{policy}: client-defined attenuation policy
F_{update}: client-defined state updater

Stateful Authorization Model: Token Usage



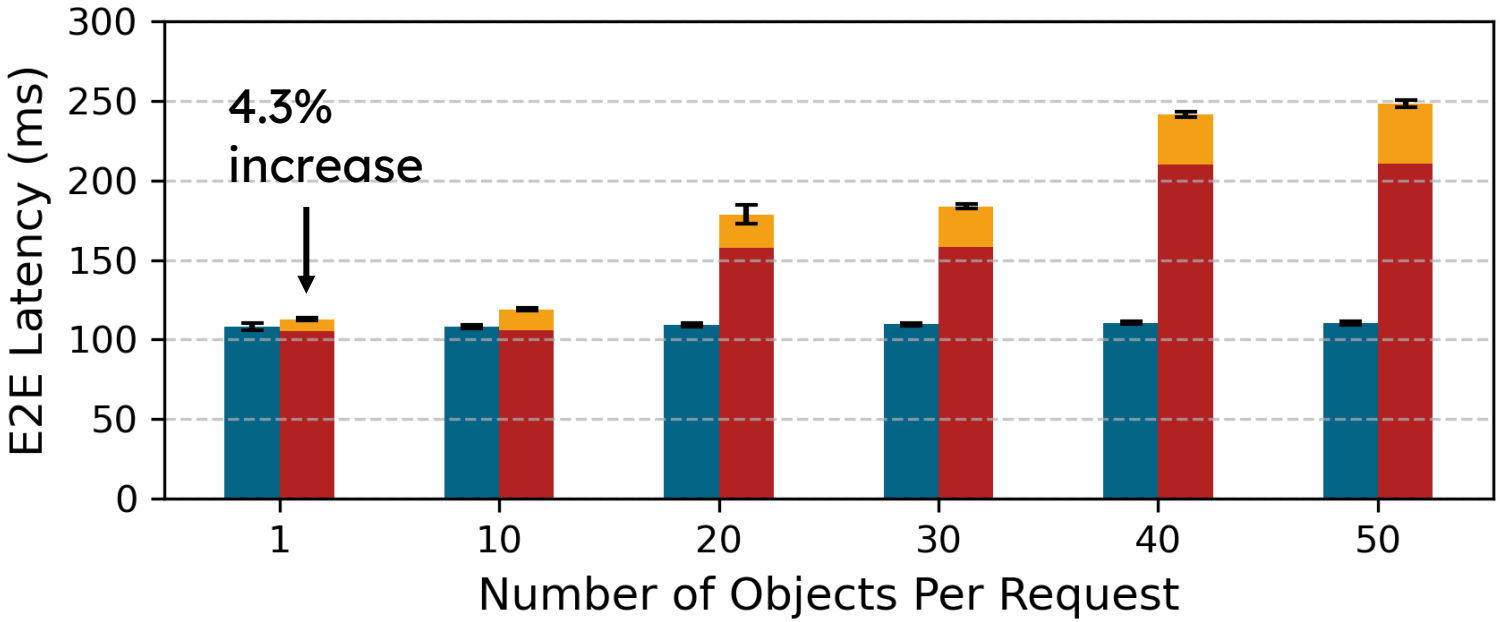


S_p: server-define scope
F_{policy}: client-defined attenuation policy
F_{update}: client-defined state updater

Case Studies

Policy Type	Client App	Service API	Least Privilege Policy
Access-only-created	zoom	31	Only view/edit the events created by Zoom
Read-at-most-once	Trip Planner		Read each email at most once
Write-at-most-once	TravisCI	GitHub	Update each check run at most once

Evaluation - Latency



Summary

Server-defined Permissions

Statelessness

Fundamental Overprivilege

Stateful Least Privilege Authorization Framework

Enable the client app to *express* its minimum privilege

Enable the API server to *enforce* the minimum privilege w/ state information

Thank you!

UC San Diego

Luoxi Meng
lumeng@ucsd.edu