Sidecar-based Path-aware Security for Microservices

Catherine Meadows, Sena Hounsinou, Timothy Wood, Gedare Bloom

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

UCCS
University of Colorado Colorado Springs

Microservice Architectures

Popular architecture for today's web services

- Decompose web applications into smaller components that interact
- Improves deployment, development, and scalability

Challenges

- Increased attack surfaces
- Large number of components leads to complex security policy enforcement

Figure. Microservice deployment graphs for Amazon and Netflix (from "Navigating the Microservice DeathStar With DeployHub" by Tracy Rogan)

Microservice Sidecar Design Pattern

Sidecars are containerized, peripheral components that intercept incoming and outgoing requests to services for added communication and functionality

- Service mesh architectures help connect microservices

Typical Technologies

- Web applications like *Node.js* and *Python* make up microservice components
- *Envoy Proxy* and *Istio* service mesh provide sidecar routing between microservices

Microservice Sidecar Design Pattern

Sidecars are containerized, peripheral components that intercept incoming and outgoing requests to services for added communication and functionality

- Service mesh architectures help connect microservices

Typical Technologies

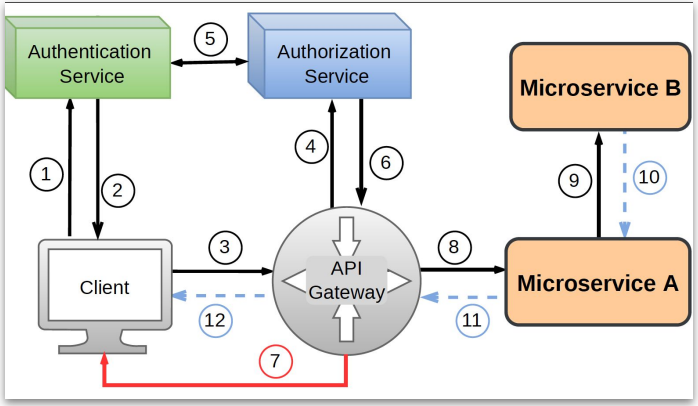
- Web applications like *Node.js* and *Python* make up microservice components
- *Envoy Proxy* and *Istio* service mesh provide sidecar routing between microservices

Sidecars can be an ideal vantage point to provide security services without requiring modifications to each microservice

Token-based Authentication

Clients use an authentication service to acquire an access token based on their privilege level

- Subsequent requests include this token that can be inspected by services for authorization



Current approaches authenticate users at start of request but don't account for complex topology and dynamic nature of microservices

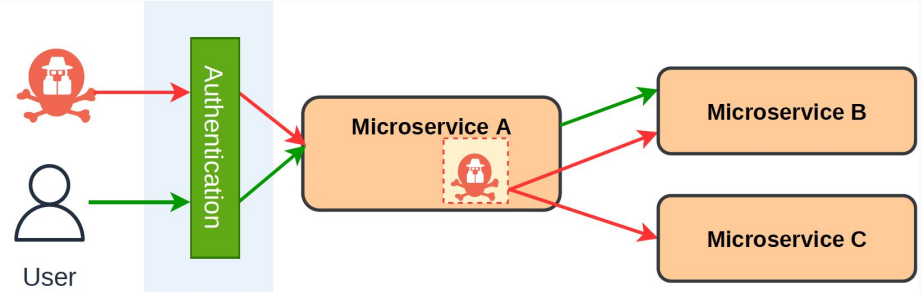
Threats and Vulnerabilities

Client-based Attacks

- Request smuggling
- Token misconfiguration

Microservice-based Attacks

- Malicious services



Microservices may be more vulnerable to attacks where one component becomes compromised and can disrupt the rest of the application since there is limited authentication after passing through gateway

Our Goals

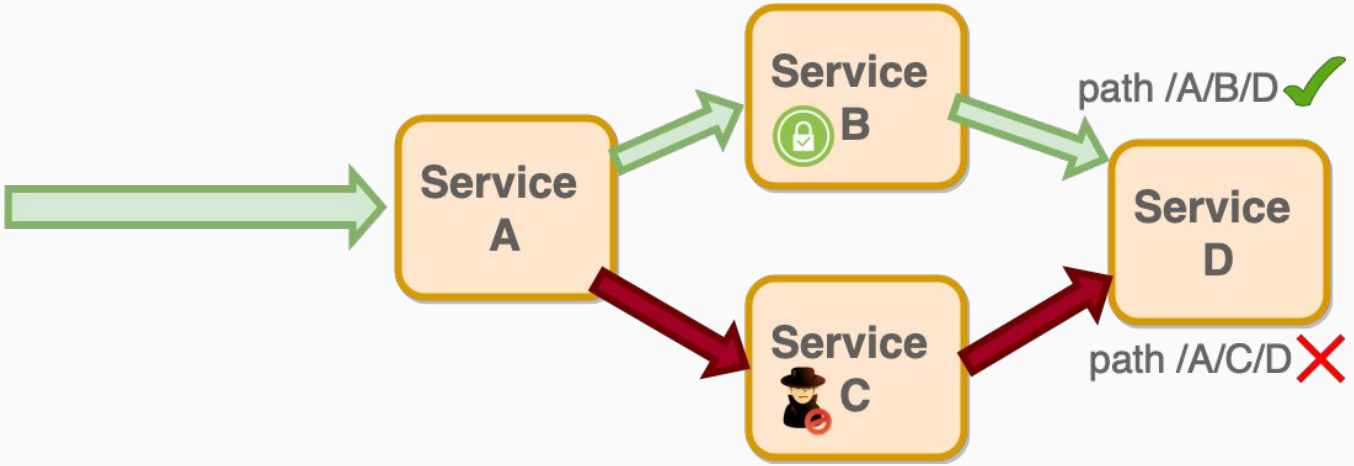
Provide path-aware security for microservices,

Perform authentication checks in sidecars using both client and path information,

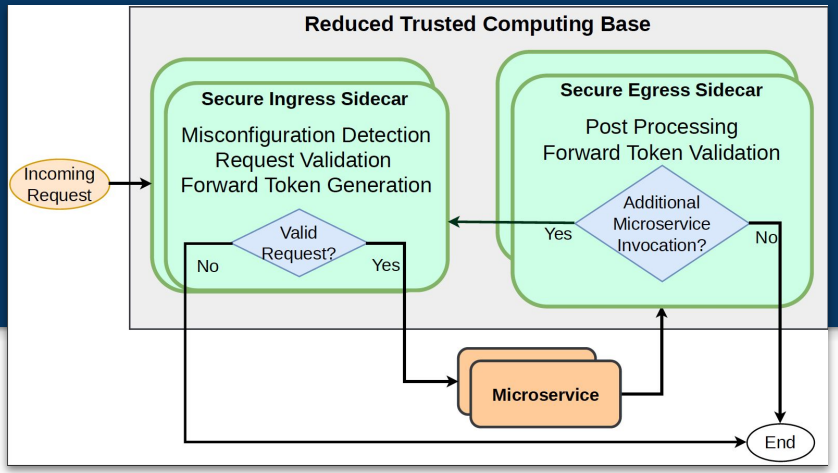
Minimize modifications to existing application code

Benefits of Path Aware Security

Path-aware whitelists or blacklists to detect abnormal request flows that might indicate a compromised service



Design

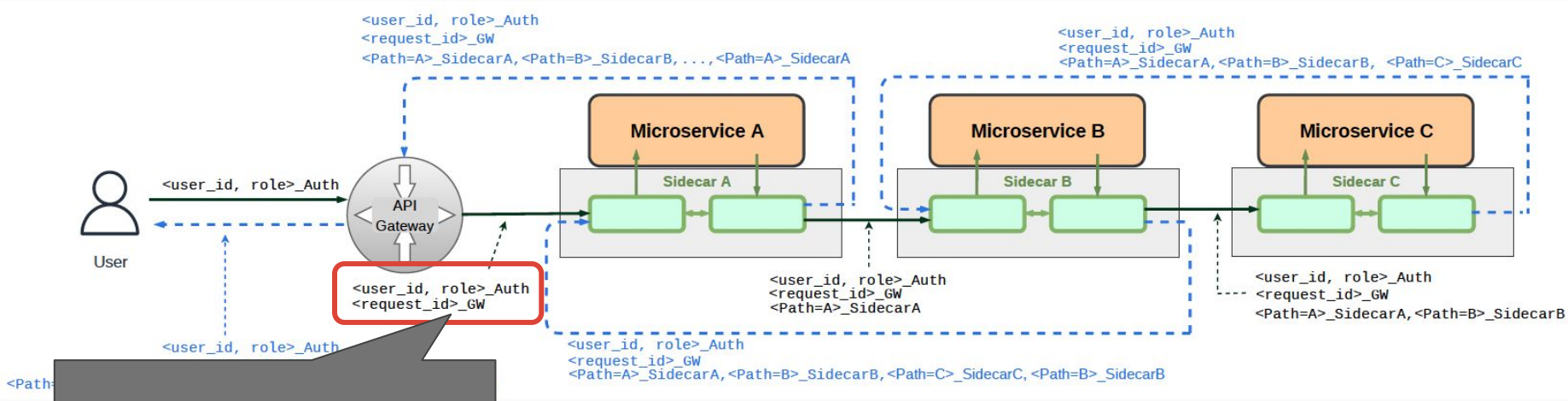


Leverage sidecars to reduce trusted computing base

- Ingress Proxy
  - Request misconfiguration check
  - Path inspection
- Egress Proxy
  - Blocks outgoing requests to unauthorized services
  - Appends signed sidecar token to request header path field (e.g. $\langle Path=A \rangle_{Sidecar\_A} \,||\, \langle Path=B \rangle_{Sidecar\_B}$

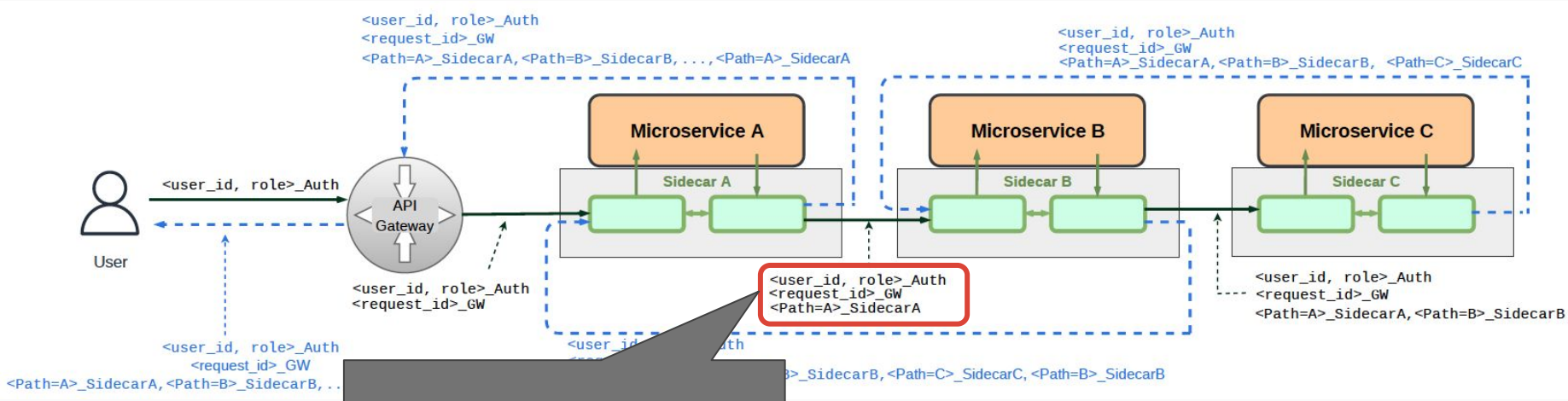
We can use sidecars to both check the incoming path and provide a trusted place to append path information

Request Flow

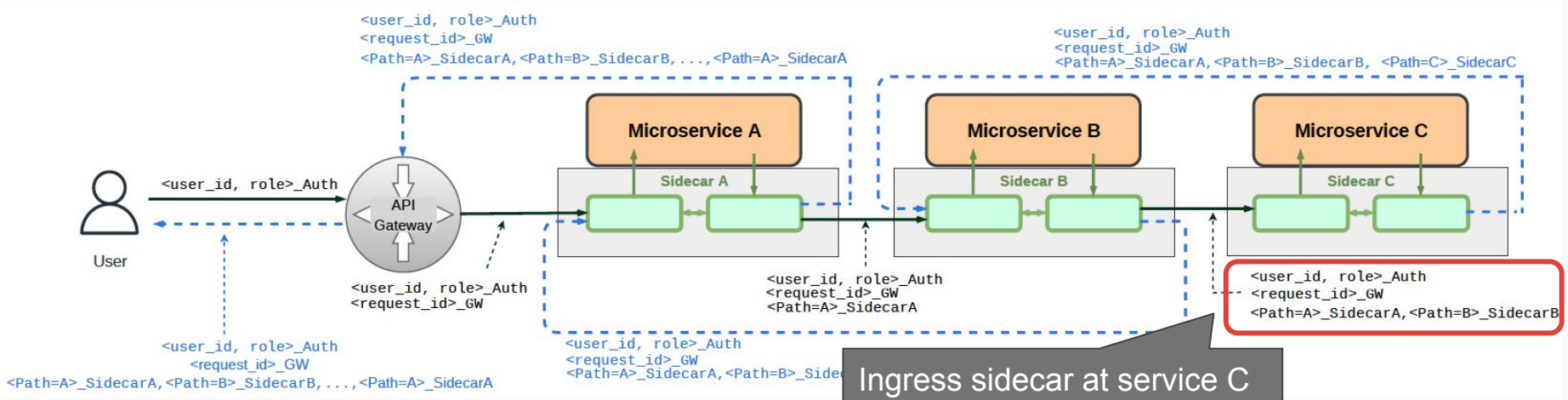


Request has 2 tokens signed by Auth and GW

Request Flow



Request Flow



Ingress sidecar at service C can perform access control based on entire preceding path as well as user

Summary and Future Work

Microservices provide software development benefits but are more complex to manage and secure

Sidecar-based architecture is an ideal approach for seamlessly providing security services to large scale microservice deployments

Embedding path information tokens in requests will allow sidecars to validate whether incoming and outgoing requests meet security policies

Ongoing work explores how to implement security sidecars without requiring trust of the microservice components