Attribute-based Access Control for Microservices-based Applications using Service Mesh

Dr. Ramaswamy Chandramouli (Mouli)
mouli@nist.gov (NIST)

Zack Butcher, zack@tetrate.io (Tetrate Inc)

DevSecOps and Zero Trust Architecture in Multi-Cloud

Wednesday, January 27, 2021

ABAC for Microservices-based Applications using Service Mesh - Outline

- Microservices-based Applications & Service Mesh – A quick Overview
- Requirements for AuthN and AuthZ framework
- ABAC – Overview of Representations
- Pre-requisites for AuthN and AuthZ deployment in Service Mesh
- Existing & Emerging AuthN & AuthZ deployments in Service Mesh
- ABAC Deployment for Service Mesh
- Summary & Contacts

Microservices-based Applications & Service Mesh –A quick overview

- **Application Architecture in cloud-native Applications**: Applications have multiple loosely coupled components called microservices that communicate with each other across the network.

- **Service Mesh**:

A dedicated infrastructure called the service mesh provides all services for the application (e.g., authentication, authorization, routing, network resilience, security monitoring), which can be deployed independently of the application code.

Requirements for AuthN and AuthZ Framework

- The code that is part of this framework is verifiable and non-bypassable (always invoked), thus satisfying the requirements of a security kernel.
- The framework should provide authentication and authorization services at multiple levels: service and end-user.
- The framework should be able to support a diverse set of authorization policies.

ABAC – Overview of Representations

- A platform-neutral text-based language called eXtensible Access Control Markup Language (XACML) Version 3.0, which has been standardized by OASIS.
- Next Generation Access Control (NGAC), whose data structure and operations have been standardized under INCITS 565-2020.

ABAC – Overview of Representations..Contd

Advantages of NGAC

- Standardization of the APIs of functional components (i.e., PEP, PDP, RAP), allowing for the interoperability of these components from different sources.
- Common PEP interface for enforcing policies over both application requests and policy administration requests.
- Graph-based representation of access control data and Use of linear time algorithms for computing access control decisions and performing policy reviews.

Pre-requisites for AuthN and AuthZ deployment in Service Mesh

- *Hosting Platform Application Configuration Data* – Service Identities, Protocols and Ports, Sets of Instances for a service, Namespaces
- *Securely Configured Container deployments* –

  (a) Not configured to run as root

  (b) Not specifying host path volumes

  (c) Configure container file system as *read-only* unless app needs to write to disk

  (d) Explicitly prevent privilege escalation

Pre-requisites for AuthN and AuthZ deployment in Service Mesh..contd

- *Service Mesh Configuration*

(a) A control plane that securely obtains hosting platform configuration data, encodes policies and pushes them into various proxies.

(b) Proxies (data plane) configured with AuthN and AuthZ policies. Modules which can emit metrics & logs to ensure policies are enforced.

(c) CA module that generates signing certs rooted in *org PKI* with capabilities for frequent rotation.

Existing & Emerging AuthN and AuthZ Deployments in Service Mesh

- *Service-level AuthN Policy support*

(a) Ability to define AuthN policies at Global, Namespace, Service, port levels with overrides or Inheritance capabilities with mutual TLS feature.

(b) With Certs for mutual TLS carrying server identity, secure naming service to map service identity to server identity to prove authenticity of the server to run the service and prevent network hijacking.

Existing & Emerging AuthN and AuthZ Deployments in Service Mesh..contd

- *End-user AuthN Policy support*

(a) User credential (usually in the form of a JSON Web Token (JWT) must be part of each request's metadata.

(b) Authentication policy should at the minimum, support instructions for extracting the credentials from the request (e.g., subject, claim fields in JWT) and instructions for validating the credential (e.g., location of public keys for validating the signature in JWT).

Existing & Emerging AuthN and AuthZ Deployments in Service Mesh

- Service-level AuthZ Policy support

(a) *Ability to define Scope (target) AuthN policies at Global, Namespace and Service. Same for Sources.*

(b) Allowed operations (e.g., HTTP/GET operation allowed on /details/PRODUCT-CATALOG App.

(c) Conditions under which access can take place (e.g., possession of a token)

Existing & Emerging AuthN and AuthZ Deployments in Service Mesh..contd

- End-user AuthZ Policy support (two use cases)
- First Use Case: Traditional IAM systems in an organization exist as an external service–

  (a) side-car proxy extracts prinicipal from the request and calls this external service and renders an AuthN and AuthZ verdict. Side-car is PEP and External service is the PDP.
- *Communication between sidecar and external service should be secured using mesh's in-built AuthN and AuthZ capabilities.*

Existing & Emerging AuthN and AuthZ Deployments in Service Mesh..contd

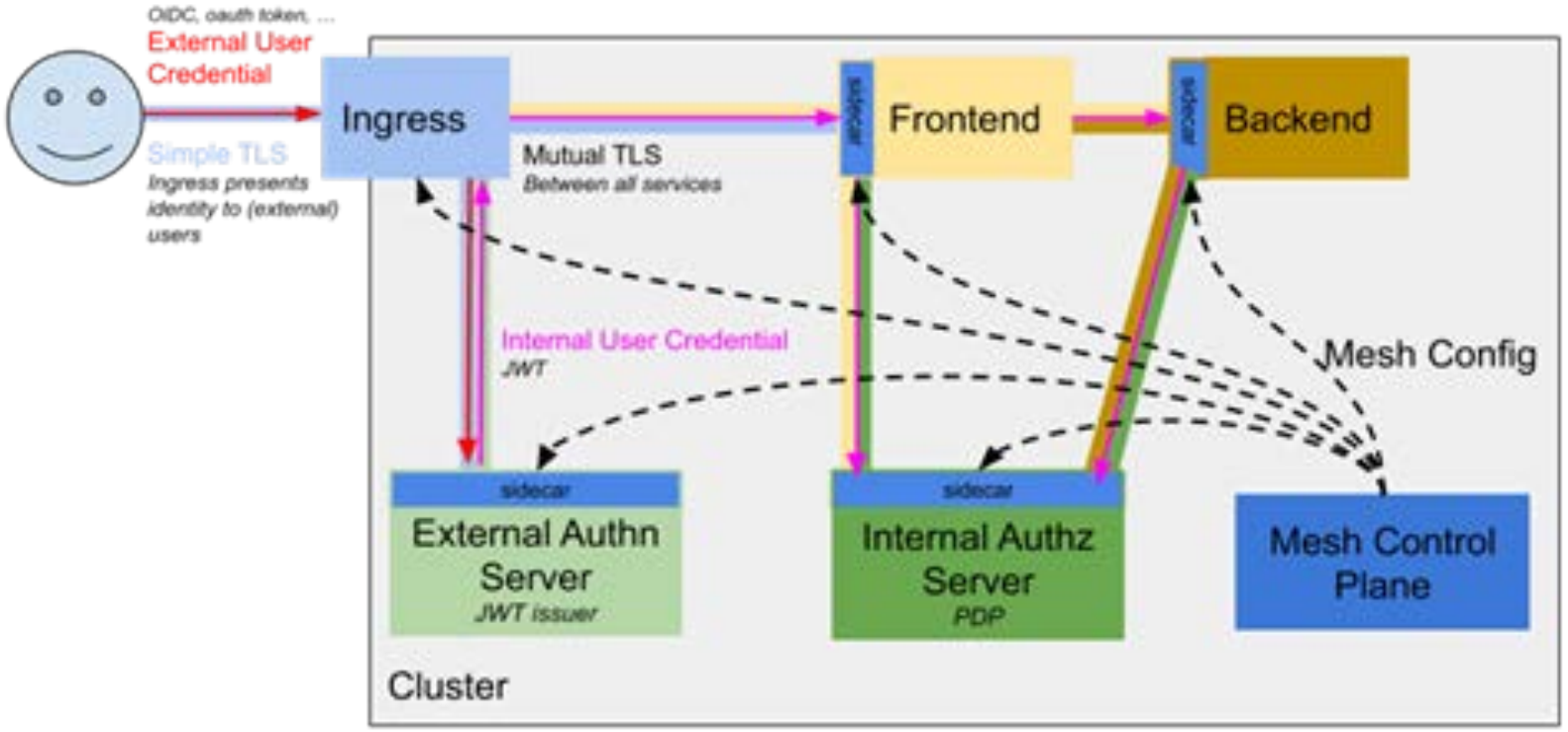
- *End-user AuthZ Policy support (two use cases)..contd*
- Second Use Case: No external authorization system

  (a) End-user credential at Ingress exchanged for an internal trusted credential that conveys user's principal and capabilities.

  (b) JWT token used to carry the credentials. Side-car has a filter to process the token.

  (c) Side-car plays the role of PDP as well as PEP.

End-user credential at Ingress exchanged for an internal trusted credential that conveys user's principal and capabilities.



Existing & Emerging AuthN and AuthZ Deployments in Service Mesh..contd

- *End-user AuthZ Policy support (two use cases)..contd*
- Second Use Case..contd: Functions of sidecar filter

(a) JWT token verification – extract token from request header, verify issuer and audience field, fetching public key and verifying the digital signature on the token.

(b) Matching resources in the request to the claims in the token to determine the rights of the end-user to access the requested sources.

ABAC Deployment for Service Mesh

- Security Assurance for AuthZ Policy Enforcement
- Supporting Infrastructure for SVC-SVC and End-user + SVC-SVC Request types
- Advantages of ABAC AuthZ framework for Service Mesh
- Enforcement alternatives in Proxies

ABAC Deployment for Service Mesh

Security Assurance for AuthZ Policy Enforcement

(a) Sidecar proxies intercept every request from a client or microservice and hence are non-bypassable and thus play the role of PEP.

(b) The code in the proxy is independent of application code and cannot be modified.

(c) The outcome of policy enforcement can be independently verified through shadow operations and live production requests.

ABAC Deployment for Service Mesh

Supporting Infrastructure for SVC-SVC and End-user + SVC-SVC Request types

(a) CA Module in the service mesh control plane that issues certs to all services through its proxies.

(b) Identity Registry – Authorized source for services identities

(c) Service mesh control plane is used to obtain resources attributes, registered users' attributes, etc (which in turn uses services orchestration APIs), thus, playing the role of Policy Information Point (PIP) for external authorization server.

ABAC Deployment for Service Mesh

Supporting Infrastructure for SVC-SVC and End-user + SVC-SVC Request types..contd

(d) The external authorization server plays the role of PDP rendering access control decision in response to call from sidecar proxies.

(e) The authorization server when called from Ingress proxy in some instances may generate a JWT token.

(f) The sidecar proxies makes calls to external authorization server, obtains authorization decision in the form of ALLOW/DENY and enforces those decisions for each request thus playing the role of PEP.

ABAC Deployment for Service Mesh

Advantages of ABAC AuthZ framework for Service Mesh

(a) Resource-level authorization as opposed to method-level authorizations provided by proxy supported native policy objects.

(b) Both types of requests (SVC-SVC and EU+SVC+SVC) are handled using the same runtime components thus providing tight integration between service level and end-user level authorizations.

ABAC Deployment for Service Mesh

Advantages of ABAC AuthZ framework for Service Mesh..contd

(c) The extensible API of the sidecar proxy can be used to integrate with any authorization engine using the access control model suitable for the application and consistent with enterprise-level policies.

(d) ABAC has been found to be one of the most flexible scalable access controls because of its ability to incorporate any number and type of attributes associated with subject, object and environment.

ABAC Deployment for Service Mesh

Advantages of ABAC AuthZ framework for Service Mesh..contd

(e) The NGAC model chosen for ABAC deployment uses a directed acyclic graph for representation of access control permissions and generates access decisions using an algorithm that is linear in terms of model dimensions in arriving at a decision.

(f) The extensible API of the sidecar proxies can be leveraged to incorporate external authorization servers that can be used for both application and data protection (e.g., NDAC)

Summary & Contacts

- Guidance for ABAC deployment in a Service Mesh provided outlining
  - (a) Requirements for Security Assurance
  - (b) Supporting Infrastructure
  - (c) Advantages
- The Publication URL: https://csrc.nist.gov/publications/detail/sp/800-204b/draft
- Contacts

- Ramaswamy Chandramouli (mouli@nist.gov)
- Zack Butcher (zack@tetrate.io)
- Aradhna Chetal (aradhna.chetal@gmail.com)