A Data Protection Approach for Cloud-Native Applications

Ramaswamy Chandramouli
(Computer Security Division (ITL) – NIST)

Wesley Hales
(Leak Signal Inc.)

Federal Cybersecurity and Privacy Professionals Forum – Nov 19, 2024

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
U.S. DEPARTMENT OF COMMERCE

A Data Protection Approach for Cloud-Native Applications

- **Cloud-Native Applications – Microservices-based Architecture**
- Need for a Centralized Security Infrastructure (Service Mesh) & its Building Blocks
- WebAssembly Framework & WASM Modules
- Functionality of WASM modules & their Security Profile
- Data Protection Approaches – Background
- Why we chose WASM
- Data in-Transit – Categorization & Protection Approaches
- Potential Application Areas
- Conclusions & Summary
- Document Link & Points of Contact

Cloud-Native Applications - Microservices-based Architecture

Cloud-Native Applications (CN-Apps)- Applications distributed over on-premises and multiple clouds

A popular architecture in CN-Apps is one that is Microservices-based.

The application is built using a set of independently scalable, loosely coupled, business process-focused services called *Microservices – implemented as Containers*

Each of the microservices may be hosted on heterogeneous platforms, physically located either on premises (HQ, Branch offices) or in multiple clouds.

A Data Protection Approach for Cloud-Native Applications

- Cloud-Native Applications – Microservices-based Architecture
- **Need for a Centralized Security Infrastructure (Service Mesh) & its Building Blocks**
- WebAssembly Framework & WASM Modules
- Functionality of WASM modules & their Security Profile
- Data Protection Approaches - Background
- Data in-Transit – Categorization & Protection Approaches
- Potential Application Areas
- Conclusions & Summary
- Document Link & Points of Contact

Why is there a Need for a centralized Security Infrastructure (Service Mesh) for Microservices-based Application Architectures

- **Simple Answer: Need for Multiple (Security) Application Services guided by uniform policies at a level of abstraction independent of individual microservices**

Examples of Application Services

- *For discovering the ephemeral containers containing the microservice* – Secure Service Discovery Service
- *For Authentication & Authorization*: Secure Service Identity Service, Secure Token Service (STS) for carrying identity & access rights
- *For Secure Communication* – e.g., Mutual TLS
- *For Network Resilience* - Load balancing, Circuit breaking, Rate limiting etc.
- *For Continuous Monitoring* (the health of the microservices) – Logging Service, Telemetry Service (gathering metrics), Tracing Service

Service Mesh Building Blocks – Data Plane (Proxies) & Control Plane

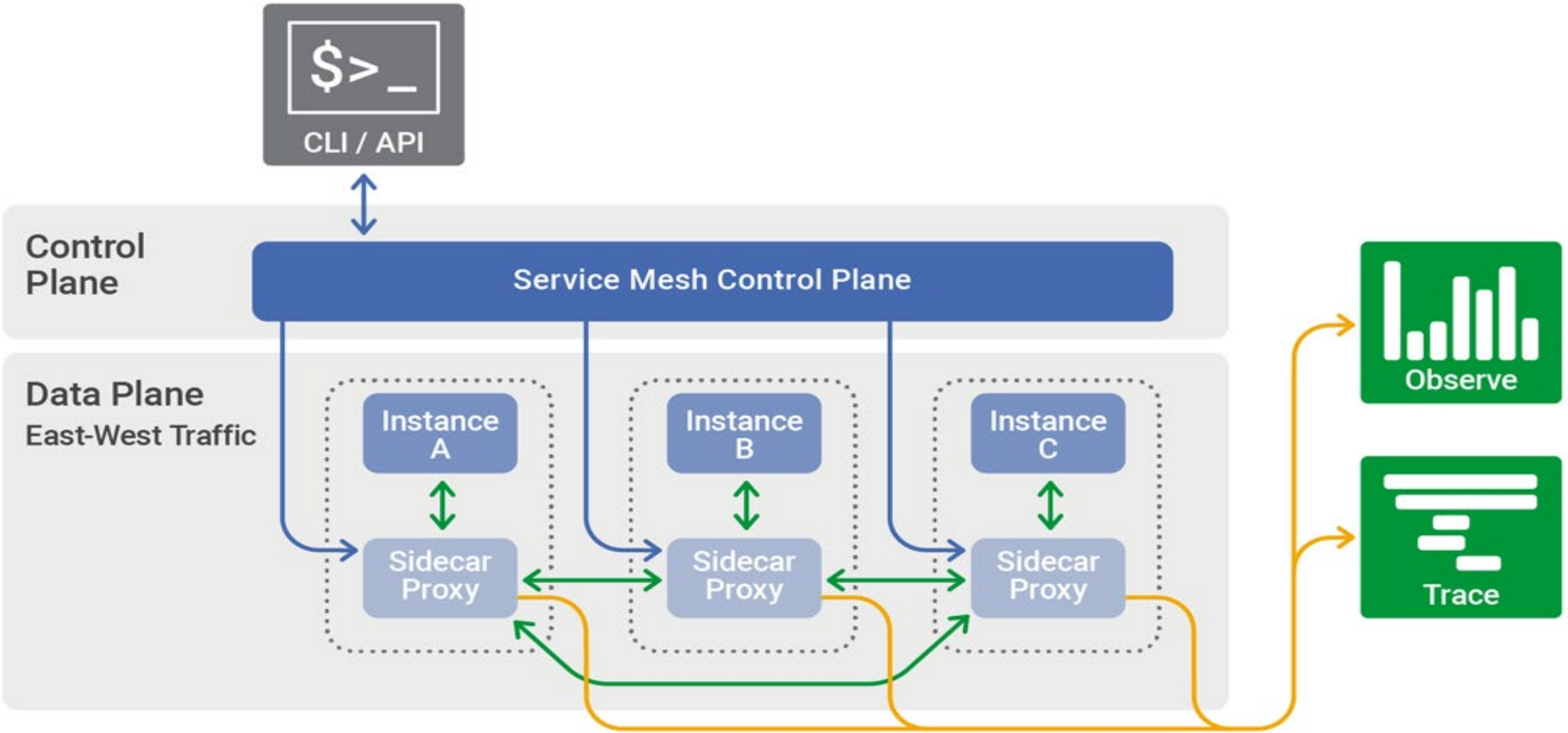
The purpose of a Service Mesh is to provide various Application Services & Enforce run-time policies.

Run-time policies for microservices should be defined outside the application code. Usually enforced by running a proxy

Proxies need to be programmed with policies and need to be deployed & associated with microservices. This function needs another component.

Thus a Service Mesh has two components. Proxies form the data plane and a control plane which consists of interfaces for deploying proxies with its associated policy enforcing code and runtime configurations. The combination of a Data plane & a Control plane forms a Service Mesh instance.

A Service Mesh Instance – A Control Plane and a Data Plane consisting of a set of sidecar Proxies



A Data Protection Approach for Cloud-Native Applications

- Cloud-Native Applications – Microservices-based Architecture
- Need for a Centralized Security Infrastructure (Service Mesh) & its Building Blocks
- **WebAssembly Framework & WASM Modules**
- Functionality of WASM modules & their Security Profile
- Data Protection Approaches - Background
- Data in-Transit – Categorization & Protection Approaches
- Potential Application Areas
- Conclusions & Summary
- Document Link & Points of Contact

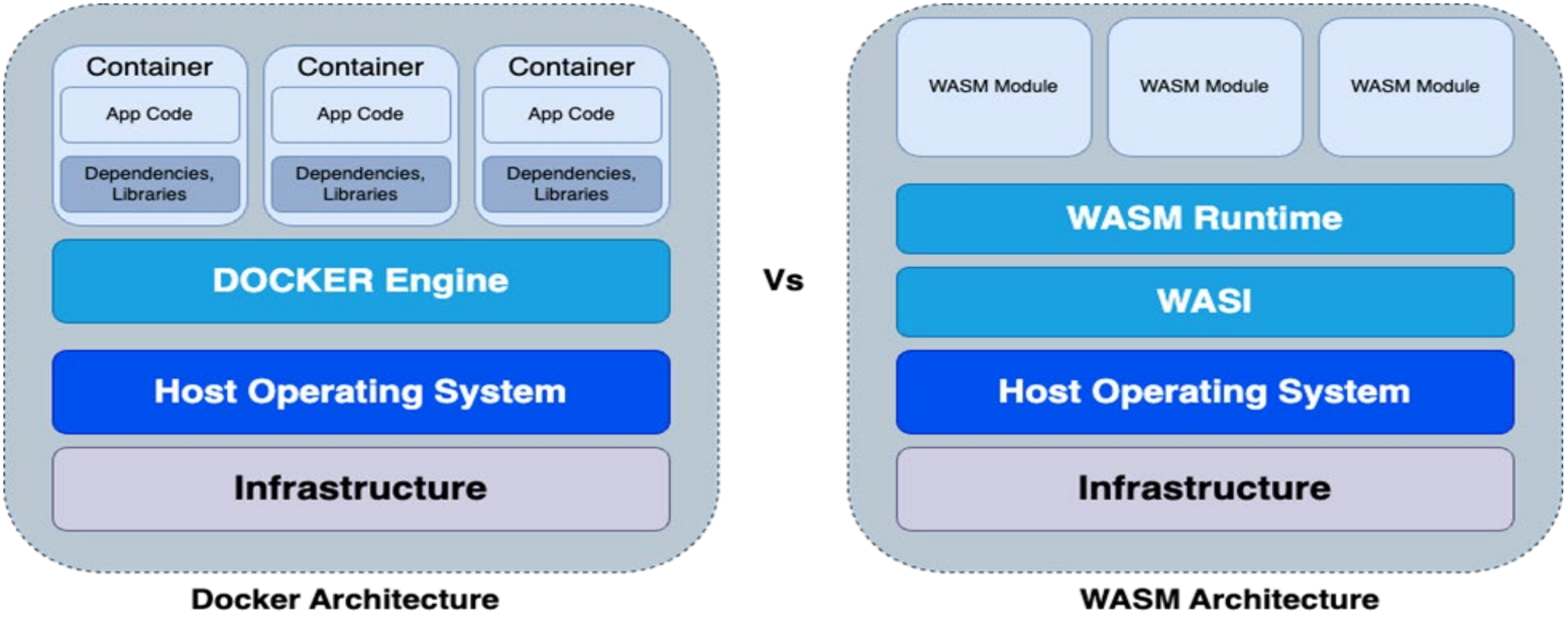
WebAssembly Framework & WASM deployment in Service Mesh Proxies

- **Using the family of APIs provided by proxies, service modules are implemented and housed in proxies. One such service module ecosystem is WebAssembly Module (WASM) that is deployed using an API called Proxy-WASM runtime.**

Steps involved in generating & running WASM

- ***Source Code****: Programs are written in languages that have target WASM compilers available.*
- ***Parsing****: The code is parsed into an Abstract Syntax Tree (AST) structure.*
- ***Compiling****: The code in AST structure is compiled into WASM code in a binary format.*
- ***Loading & Execution****: The WASM runtime loads the WASM module and creates an executable instance by allocating memory etc.*
- ***Code Optimization****:* During runtime, a combination of profiling/optimization takes place to enhance performance.

How do WASM Modules run in Proxies (Comparison with Containers)



Docker Architecture

WASM Architecture

A Data Protection Approach for Cloud-Native Applications

- Cloud-Native Applications – Microservices-based Architecture
- Need for a Centralized Security Infrastructure (Service Mesh) & its Building Blocks
- WebAssembly Framework & WASM Modules
- **Functionality of WASM modules & their Security Profile**
- Data Protection Approaches - Background
- Data in-Transit – Categorization & Protection Approaches
- Potential Application Areas
- Conclusions & Summary
- Document Link & Points of Contact

Functionality of WASM modules in Proxies & Security Profile

- **WASM Modules can be programmed to execute custom logic at various stages of microservices communication – i.e., request-response cycle.**

Examples of WASM program functions

- *Intercept Microservice requests*
- *Inspect payload data*
- *Apply data categorization*
- *Redact data before forwarding the response*

Security Profile of WASM Programs

- Always invoked since they run in proxies
- Run in sand-boxed environment & access to OS and host resources only through WASI API.

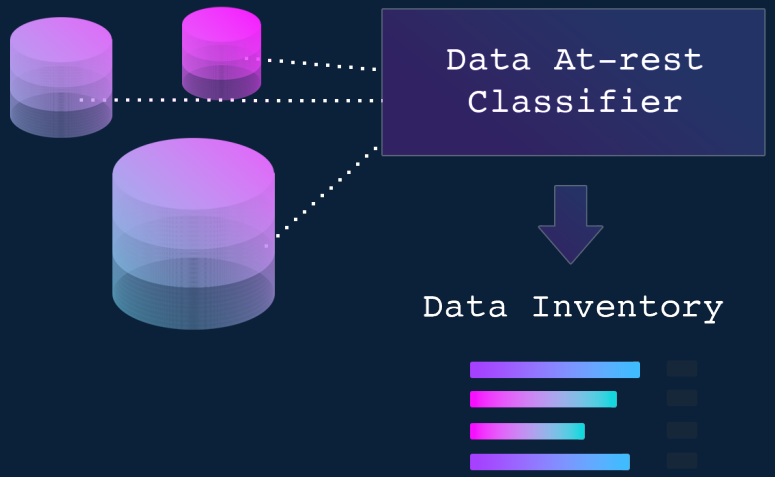
Data Protection Approaches - Background

Traditional Data Classification

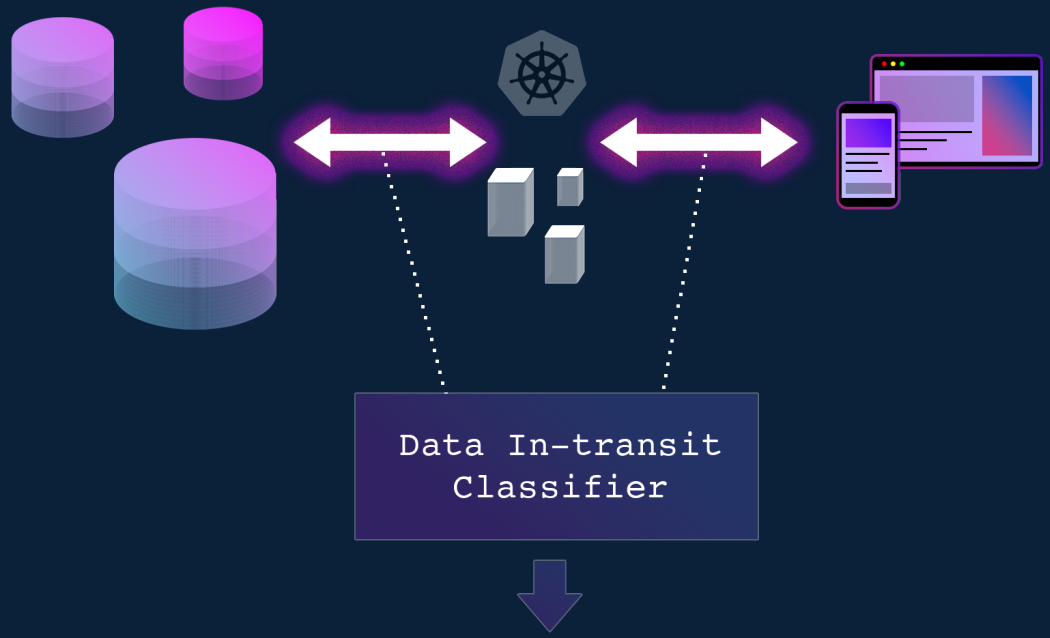
- **Regex**
  - Well known syntax
  - Scales linearly with volume of data
- **Checksum**
  - Logical computation (credit card luhn)
  - Limited to certain data types
- **Machine Learning & Natural Language Processing**
  - Data patterns within structured and unstructured data
  - Handled offline and involves copying data for training purposes

Data Protection Approaches – Background

At-rest classification



In-transit classification



Data Protection Approaches

Data at Rest

Data Inventory

- Existing Data Stores
- M365, Google Drive
- 3rd party/SaaS Data storage

Data in Transit

Realtime Data Inventory

- APIs & Production Workloads

Threat Remediation

- Data Access Observability with Redaction/Masking

Compliance Violation Resolution

- Unknown & Unwanted Data Leakage

Data in-transit Protection & WASM

Why we chose WASM

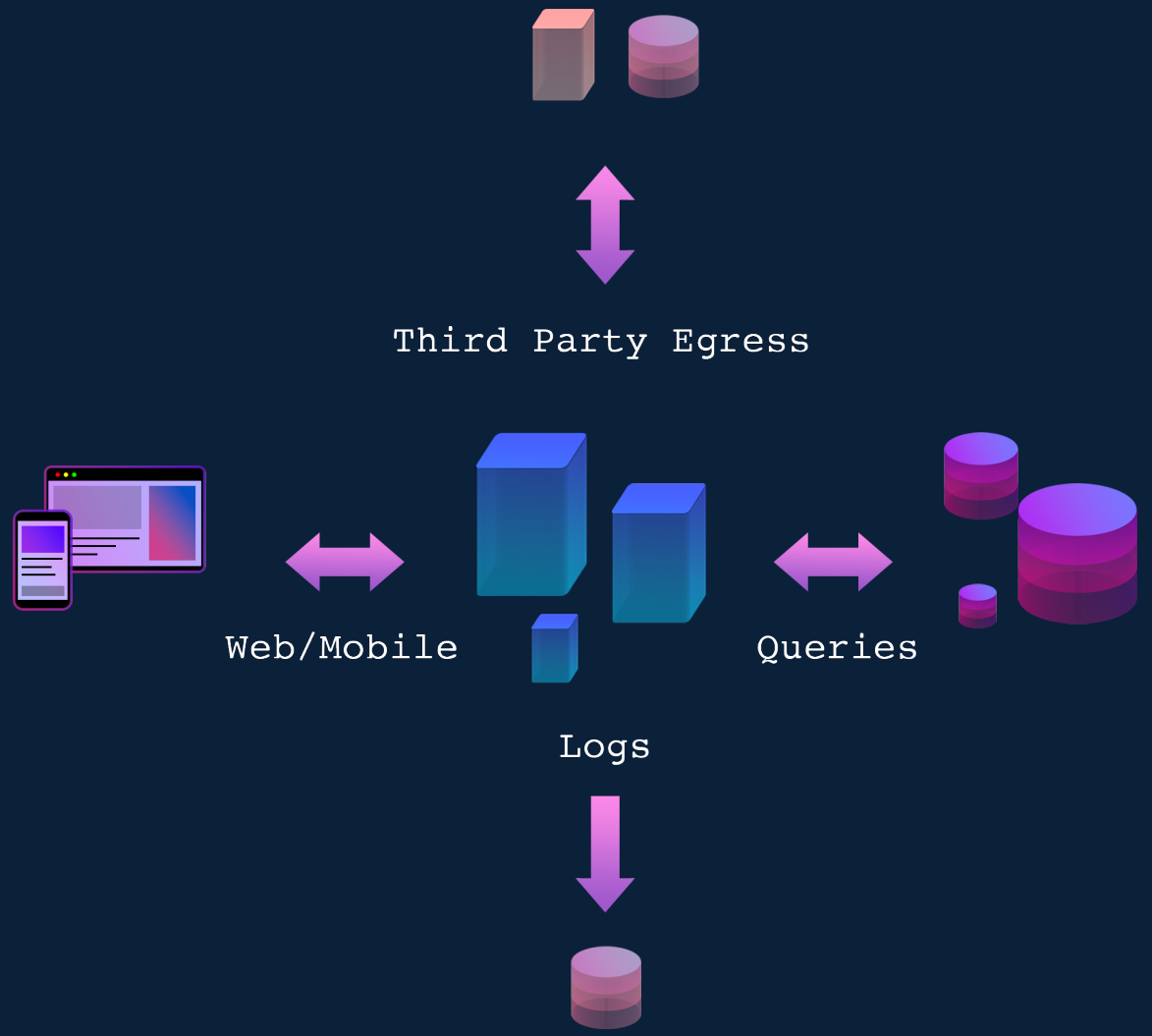
A proxy runtime for advanced data classification

- Sandboxed
- Performant
- Built-in Failover
- Rust compilation target
- No need to recompile the entire proxy
- Plug and play / Easy to deploy
- Data categorization and advanced computations in user space

Data in-transit Protection & WASM

Traditional (non-mesh) Workloads

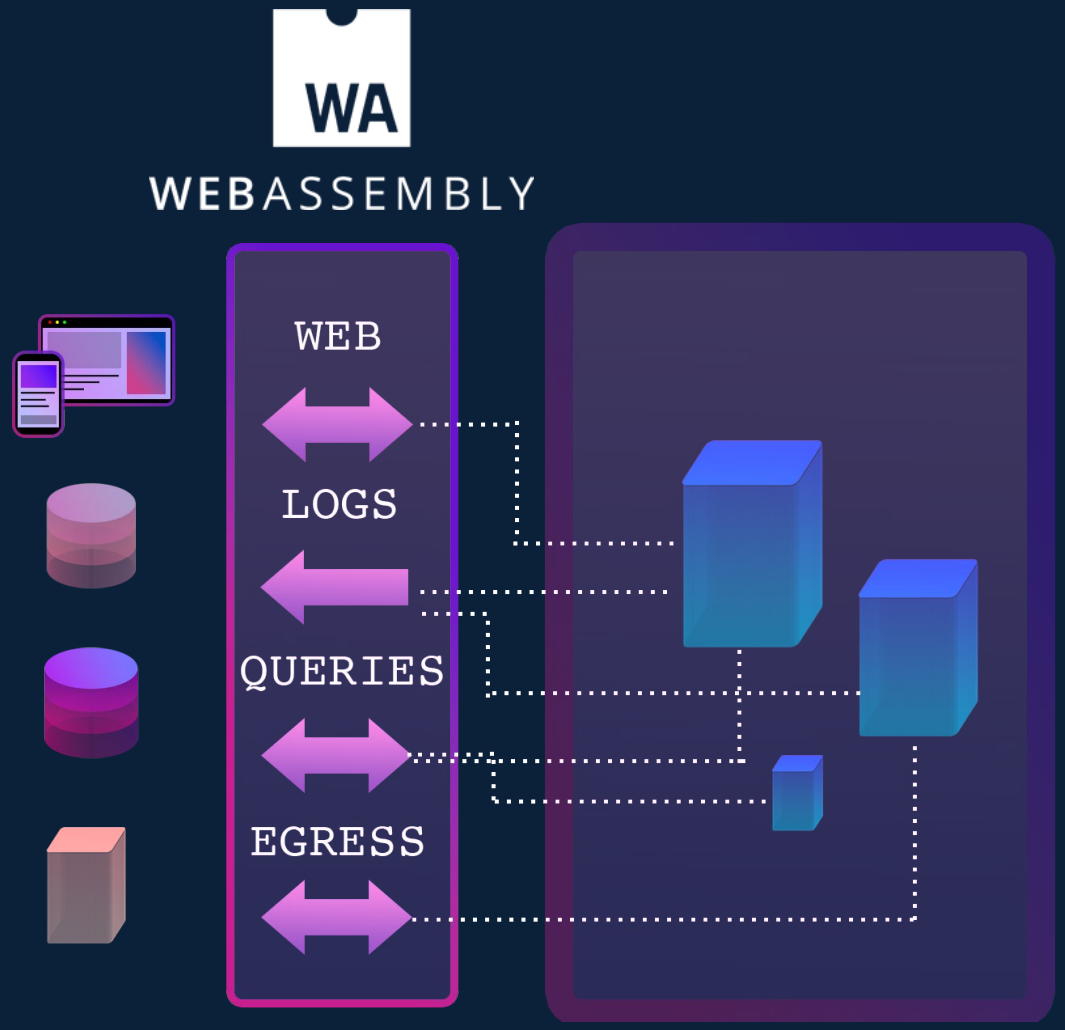
- Data flows in multiple directions across Layer 4-7 protocols.
- No centralized control point for East-West traffic
- Internal data leakage is a common issue



Data in-transit Protection & WASM

Service mesh

- Sidecar and Waypoint proxy architecture
- Easy deployment
- Provides a runtime to classify, redact and mask data in-line with network traffic



Data In-transit Protection Overview

Data Discovery and Categorization:

WASM modules can dynamically identify and categorize data as they traverse the network, ensuring that sensitive information is recognized and handled appropriately.

Dynamic Data Masking (DDM):

WASM modules can apply DDM techniques to redact or mask sensitive information in transit, enhancing privacy and security.

User and Entity Behavior Analytics (UEBA):

WASM modules can analyze user and entity behaviors in real time, detecting anomalies and potential security threats.

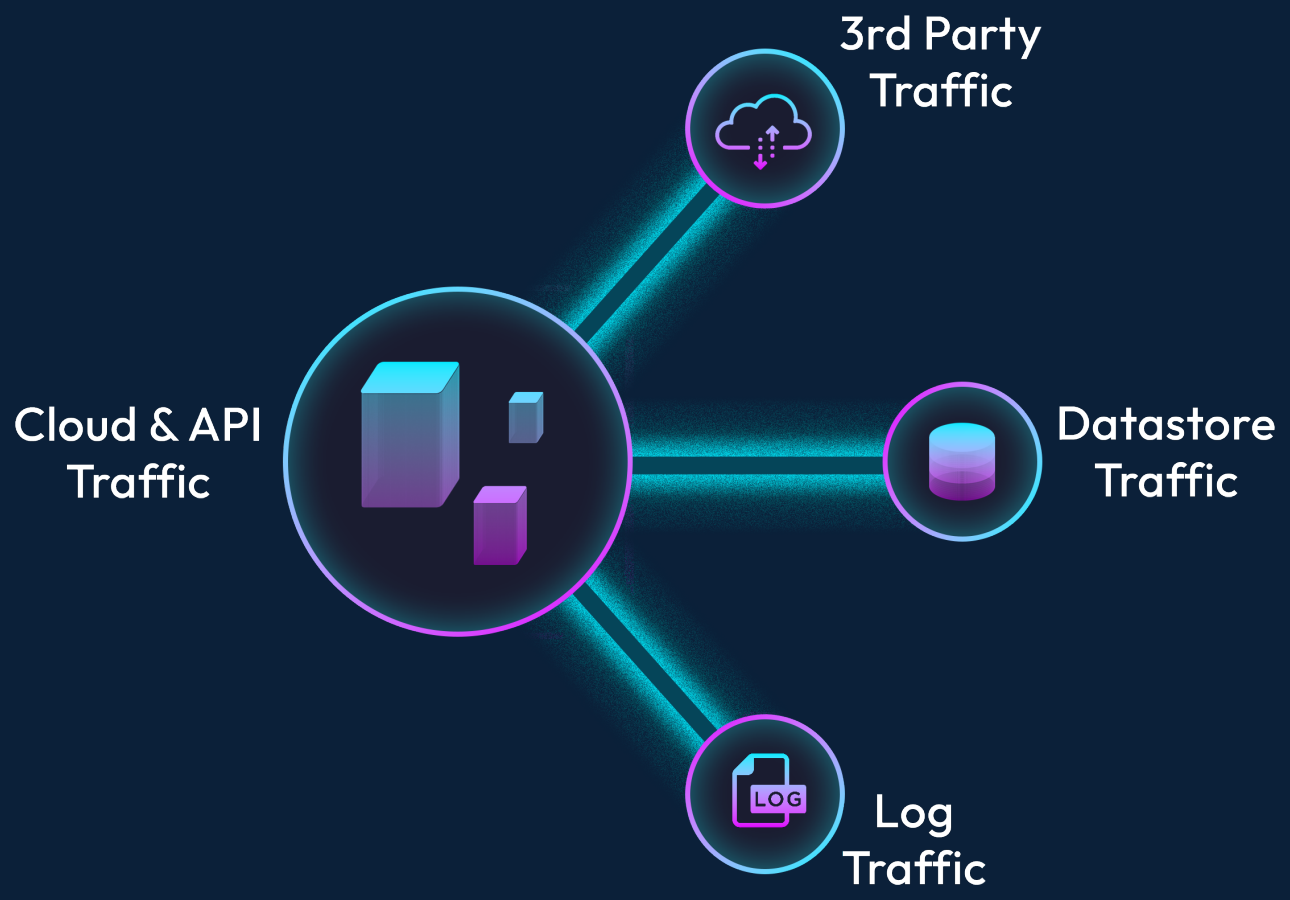
Data Loss Prevention (DLP):

WASM modules can enforce DLP policies by monitoring and controlling data transfers to prevent unauthorized data exfiltration.

New Observability Through Data Flows

Data Flow Observability

- Realtime classification of sensitive data travelling across encrypted layer 4-7 protocols



Data In-transit Categorization

Realtime Categorization Layers

Regex:
Lightweight, performant regex to understand “Does this look like sensitive data” or “Does this look anomalous? (e.g. detonated exploit)”

Data checksum:
Routing number, credit card, phone number

NLP & Content Parsing:
Named entity recognition (NER). Processing of JWT tokens and JSON documents.

Data In-transit Categorization

Async ML-based Categorization

Sensitive Data: Personally Identifiable Information (PII), financial information, health records, etc., which require strong encryption and limited access.

Regulated Data: Data subject to regulatory frameworks (e.g., GDPR, PCI DSS), which needs compliance-based protection measures.

Confidential Data: Business secrets or internal information that needs to be kept secure but isn't governed by external regulations.

Potential Application Areas

API Inventory and Security

- APIs receiving and sending sensitive data (attestation)
- Realtime exploit detection (SQLi, IDOR, RCE)
- Rate limiting based on sensitive data accessed or other request/response signals.

Microsegmentation

- Inventory and observe sensitive data flowing across services
- Segment service based on the sensitive data they handle
- Prevent lateral movement and allow expected data flows

Potential Application Areas

Log Traffic Data Protection

- Observe and redact sensitive data before it's sent into logs and central storage systems.
- Production workloads accidentally log sensitive data and WASM provides a way to prevent this costly compliance issue.

GenAI and LLM Data Protection

- Observe and secure LLM prompts and responses, reducing the risk of prompt injections and training data extraction.
- Content routing based on first-pass categorizations allows for async verification of malicious activity and unwanted LLM responses.

Conclusion & Summary

- Data in-transit categorization happens within the proxy or host where TLS is terminated - allowing for clear text data categorization.
- The WebAssembly (WASM) runtime that exists natively in modern service mesh architectures and existing proxies allows for new approaches to securing data.
- Security, fail over and performance of the WASM module must be established when running in-line with network traffic. Proxy-WASM gives us the platform to ensure safety and failover.

Document Link:

https://csrc.nist.gov/pubs/ir/8505/final

Points of Contact:

1. Ramaswamy Chandramouli (NIST) – mouli@nist.gov
2. Wesley Hales (Leak Signal) - wesley@leaksignal.com

Questions?

Thank you!