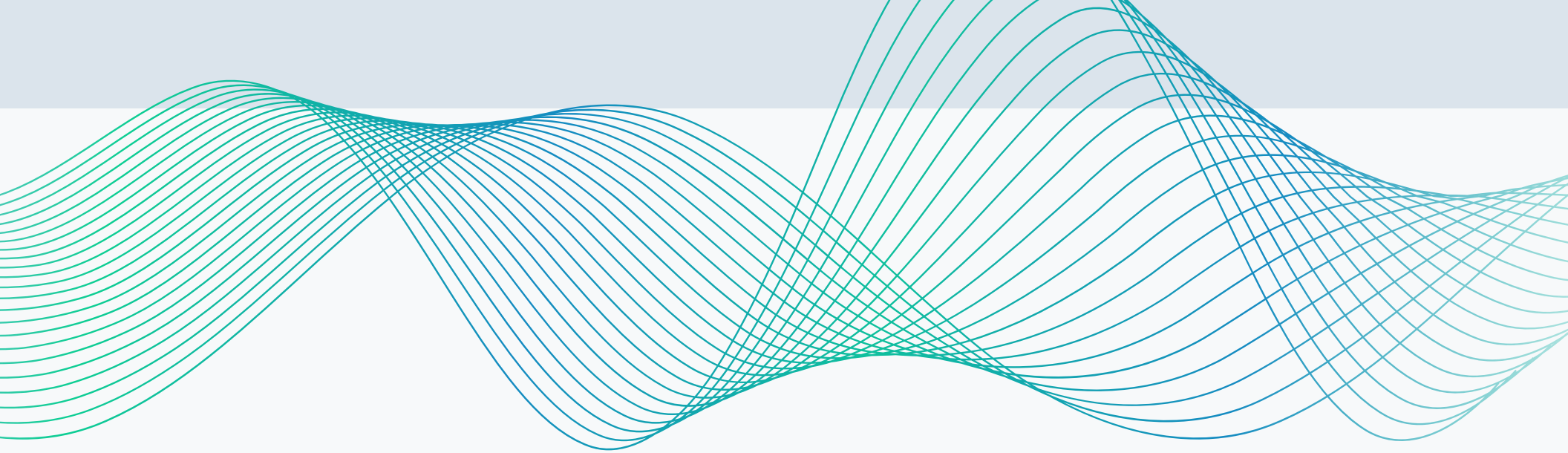


Establishing zero trust security for modern cloud architectures

How your organization can ensure safer cloud architecture by applying a zero trust network security model

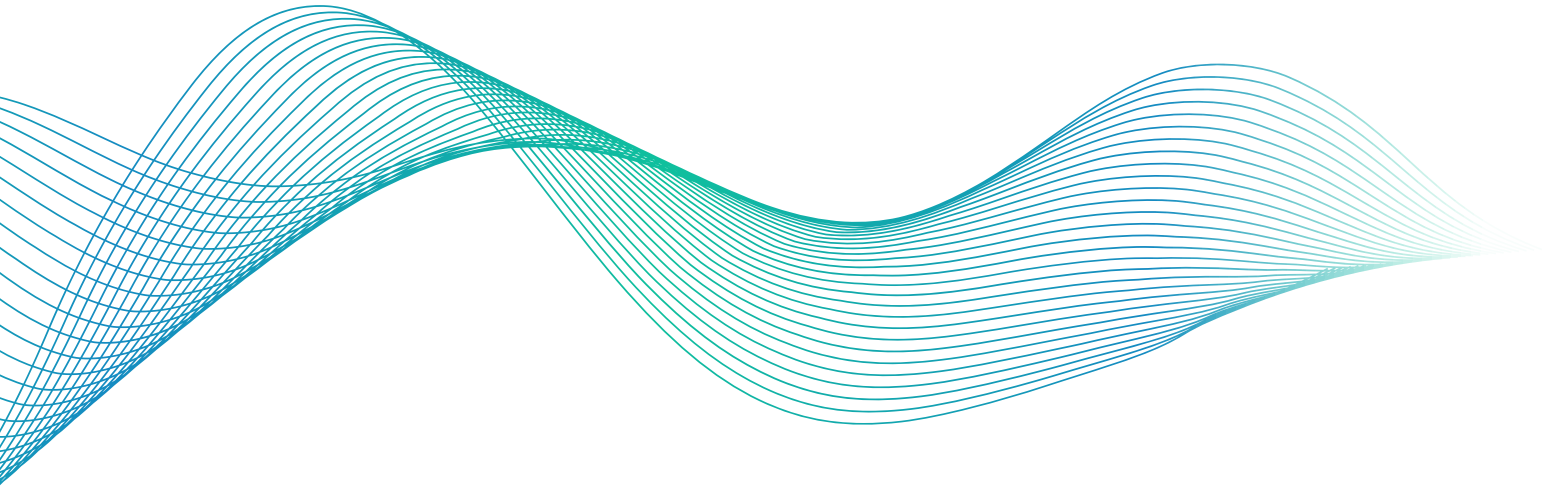
Introduction

Modern application architectures have evolved from static monoliths to distributed microservices, typically running on containers and Kubernetes, and often across multiple zones such as on premises or in a public cloud. This trend is now combined with new methodologies like DevOps, CI/CD, and GitOps to increase the rate of software innovation. The ripple effects of these changes are being felt across every functional area of IT and challenging traditional operational models for provisioning, updates, monitoring, management, and security.

Security becomes especially challenging in a fast-paced, distributed, dynamic application environment where the traditional organization boundaries are blurred. These changes in application architectures are also transforming how applications are secured, forcing organizations (or users) to rethink perimeter-based trust models. The goal of **zero trust security models** is to allow applications developers to have freedom in creating new innovation, but ensuring that application communications can ultimately be secured in these dynamic environments.

In this eBook we will outline concepts for evolving your security approach to address these dynamic environments for your modern applications.

Table of Contents

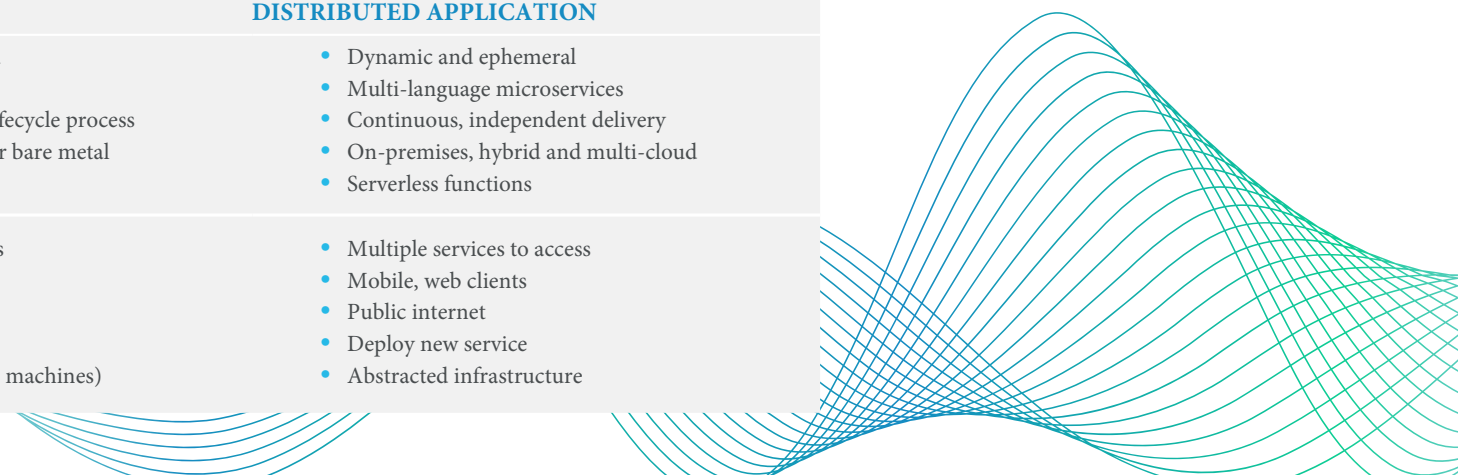


Breaking security with application modernization

Modern application architectures are designed to increase the frequency of delivery. Microservices are made of potentially hundreds of small application services loosely coupled together to leverage agile development and continuous delivery practices. New technologies have emerged to enable their development, packaging and deployment including containers, Kubernetes, git, continuous delivery, service mesh, and more. The operational best practices are still evolving to understand how best to gain visibility, observability, and control over these additional layers of abstraction.

Additionally, these new applications behave differently and challenge existing operational models and tools for monitoring and security. Instead of being fixed to a single physical machine, IP, or operating system, microservices can share OS kernels, automatically scale up and down, and are frequently orchestrated to run on different hosts. Sometimes they are running as serverless functions in which the underlying details of the infrastructure are unknown to the operator. With a growing number of assets (users, devices, services) outside the corporate firewall and distributed points of access, the traditional approach of just having a secure perimeter is not enough. The concept of what is trusted must change to include a more diverse portfolio of modern microservices, legacy applications, and infrastructure.

	MONOLITH	DISTRIBUTED APPLICATION
CHARACTERISTICS	• Static and long-lived • Monolithic • Waterfall software lifecycle process • On-premises VMs or bare metal	• Dynamic and ephemeral • Multi-language microservices • Continuous, independent delivery • On-premises, hybrid and multi-cloud • Serverless functions
SECURITY PROFILE	• One service to access • Dedicated clients • On campus or VPN • Patch in place • Fixed identities (IPs, machines)	• Multiple services to access • Mobile, web clients • Public internet • Deploy new service • Abstracted infrastructure



Get inside the perimeter for zero trust networks

Introduced in 2010 by Forrester Research, the "zero trust network" model is based on the belief that organizations should not automatically trust anything outside ***or inside*** the organization. Instead microservices should verify everything (device, end user, system) before granting permission to access any system.

Why is the shift to zero trust significant?

BEFORE	AFTER
• Do not trust anything outside the firewall	• (Still) do not trust anything outside the firewall
• Secure the perimeter	• (Still) secure the perimeter
• Provide a single point of entry	• Provide multiple points of entry and exit
• Trust anything inside corporate firewall	• Do not trust anything inside the corporate firewall either

The two most important changes are related to the following:

1. **You can no longer blindly trust anything inside the corporate firewall**
   In this context, "anything" means all devices, people, and systems. Previous models focused on protecting the network perimeter and entry onto it, so everything on the "inside" was assumed safe. This shift assumes that internal systems and end user accounts are susceptible to attack, takeover, or unintentional errors that can compromise other systems.
2. **Points of entry are becoming many and variable**
   With the adoption of SaaS, public cloud, and bring-your-own device programs, entry and exit points are now variable and often accessed over the public internet versus VPN.

THE BASIC FUNDAMENTALS OF THE ZERO TRUST MODEL INCLUDE:

Eliminated trust: No user or device should be trusted without proper authentication and authorization.

- **Least-privileged access:** Users should receive the minimum amount of access necessary.
- **Microsegmentation:** Security perimeters and network components are broken into smaller segments with individual access requirements.
- **Risk management analytics:** All network and application traffic should be logged and inspected for suspicious activity.

1 https://www.forrester.com/report/Five+Steps+To+A+Zero+Trust+Network/-/E-RES120510

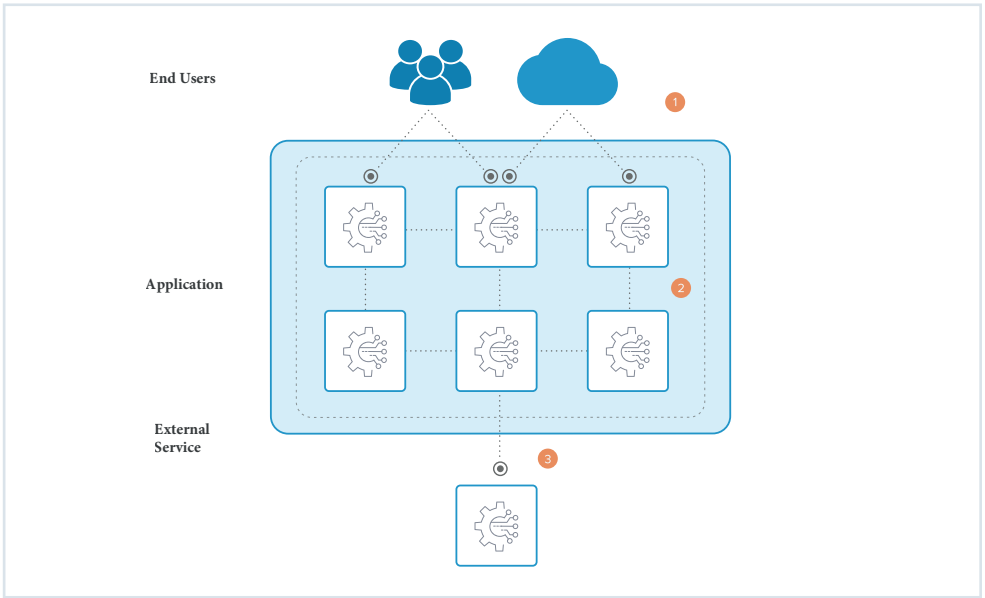
2 TechTarget article: https://searchsecurity.techtarget.com/definition/zero-trust-model-zero-trust-network

Applying zero trust for microservices

As we evaluate the development of microservices through the lens of zero trust, requirements emerge on how to practically implement those principles into the application architecture. As a collection of loosely coupled services, the network between the microservices becomes a critical factor in delivering a properly functioning application and becomes the leverage point for control and security.

The diagram below outlines the traffic patterns of microservices. Consider the numbered sections as areas to implement zero trust security principles:

1. **Ingress:** Traffic coming into the cluster may be referred to as "north-south" traffic. This is when end users or devices try to access an application service either from within or beyond the corporate firewall.
2. **Inter- and intra-cluster**: Traffic between the services may be referred to as "east-west" traffic. Depending on your environment, you may have many applications sharing the same cluster of compute resources.
3. **Egress:** Traffic leaving the cluster to an external service.



The rise of Envoy Proxy API gateways and Istio service mesh

The rise of open source, cloud native technologies like containers, Kubernetes, Istio service mesh, and Envoy proxy have made it possible to address security for microservices environments in new ways. The smart approach centralizes some aspects of security (like policies and configuration) while decentralizing others (enforcement and logging) so that the implementations and observability can scale linearly with the distributed application services.

Consider the same diagram from the previous page with zero trust principles applied using proxies at the edge and as sidecars to control, secure, and monitor the application traffic into and within the cluster. At the edge, API Gateway functionality (using Envoy proxy) is configured by Gloo Gateway to help validate the traffic and verify the requester before establishing trust and granting access to a service. Inside the cluster, service mesh functionality (using Envoy proxy) is configured by Gloo Mesh to only grant access between designated services and can encrypt communications if needed.

GLOO API GATEWAY INFRASTRUCTURE

A portfolio of tools to enable, secure, and manage modern application service connectivity.

GlooGateway

GLOO GATEWAY

An enhanced Envoy Proxy API gateway and ingress controller for Kubernetes. Gloo Edge is a lightweight yet powerful control plane to configure and manage Envoy Proxy in facilitating, shaping, and securing incoming application traffic.

GlooMesh

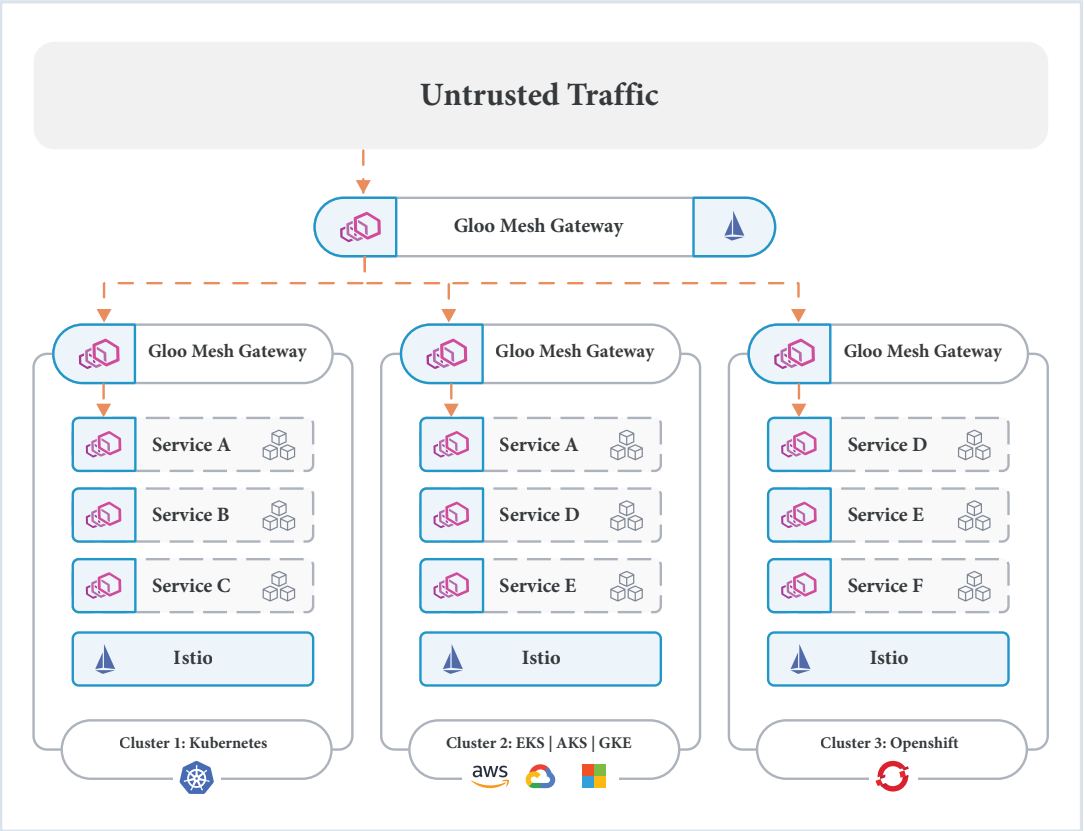
GLOO MESH

Unified management plane for single or multi-cluster Istio service mesh environments handling installation, configuration, and operations. Production support and long term support (LTS) for validated, upstream Istio software is included.

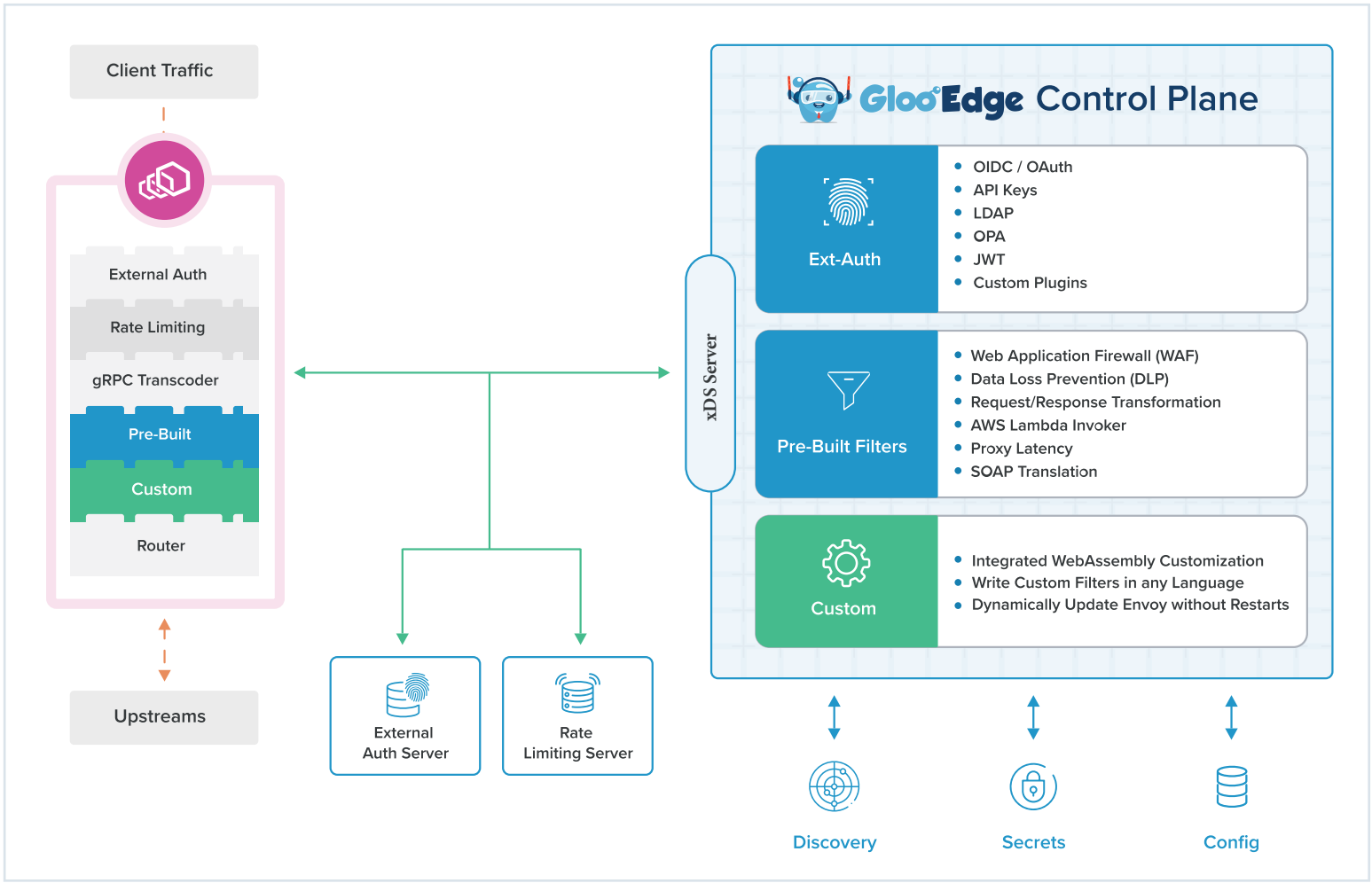
Connect, secure, and control

In the diagram below, untrusted inbound user and application edge traffic (north-south traffic) from the Internet is filtered by Gloo Gateway and secured before being directed to appropriate microservices. Security rules are implemented in Envoy Proxy filters, and logging is aggregated. Similarly Gloo Mesh core functions deliver the inter- and intra-service security controls and enforcement between microservices. Security policies are implemented centrally and federated everywhere for consistency and reduced risk. Solo's release of Envoy Proxy and Istio software is FIPS 140-2 compliant for added security.

1. **Ingress (Gateway):** There are three ways to establish trust and validate fidelity of incoming traffic to protect your systems.
   - **Web Application Filter (WAF):** Inspect, filter, and block malicious traffic and only allow safe traffic into the environment.
   - **Data Loss Prevention (DLP):** Protect loss and leakage of sensitive data (PII, Credit Cards, etc.)
   - **Authentication and Authorization:** Identify and authenticate end users (clients) and only grant access to the authorized services.
   - **Rate Limiting:** controls the volume of traffic to ensure services are not overloaded.
2. **In Cluster (Service Mesh):** Secure and encrypt communications from ingress to the services in the cluster. Only grant access between designated services and not throughout the entire cluster.
3. **Egress:** Authenticate and grant secure access to external services to complete transactions.
4. **Observability:** End to end observability of application traffic patterns and anomalies to quickly resolve issues.



In the diagram below, there is an example of how Envoy Proxy filters can directly secure traffic, while the Gloo Gateway control plane manages policies for authentication, authorization, web application firewall (WAF), data loss prevention (DLP), and other custom rules.



Conclusion

New application architectures require more than changes to code. The paradigm for operations and security must also shift to account for the disruption caused by the architectural changes. Modern applications challenge previous security conventions but have also spawned a new ecosystem of open source and commercial technologies that address these scenarios.

The potential of Envoy Proxy and Istio is reshaping networking from microservice architectures. They act as all-purpose aggregation points, presenting opportunities for traffic shaping, policy control, and observability which can be leveraged to improve overall application security. Zero trust is a comprehensive model for security encompassing devices, end users, systems, and both internal and public networks. The core principles of zero trust can be applied to modern applications in conjunction with Envoy as API gateway and Istio service mesh.

Solo.io, the leading application networking company, delivers a service mesh and API platform for Kubernetes, zero trust, and microservices. The three components of the Gloo Platform – Gloo Gateway, Gloo Mesh and Gloo Network – enable enterprise companies to rapidly adopt microservice applications, as part of their cloud journey and digital transformation. Solo.io delivers open source solutions, and is a community leader in building the technologies of the future.

Founded in 2017 in Cambridge, MA, Solo is backed by Altimeter Capital, Redpoint Ventures, and True Ventures.

Learn More

- Visit our website at www.solo.io
- Get started with a trial solo.io/trial
- Request a personalized demo solo.io/demo