KubeCon CloudNativeCon
North America 2026

Managing Network Security in Multi-Tenant Environments with Istio and Cilium

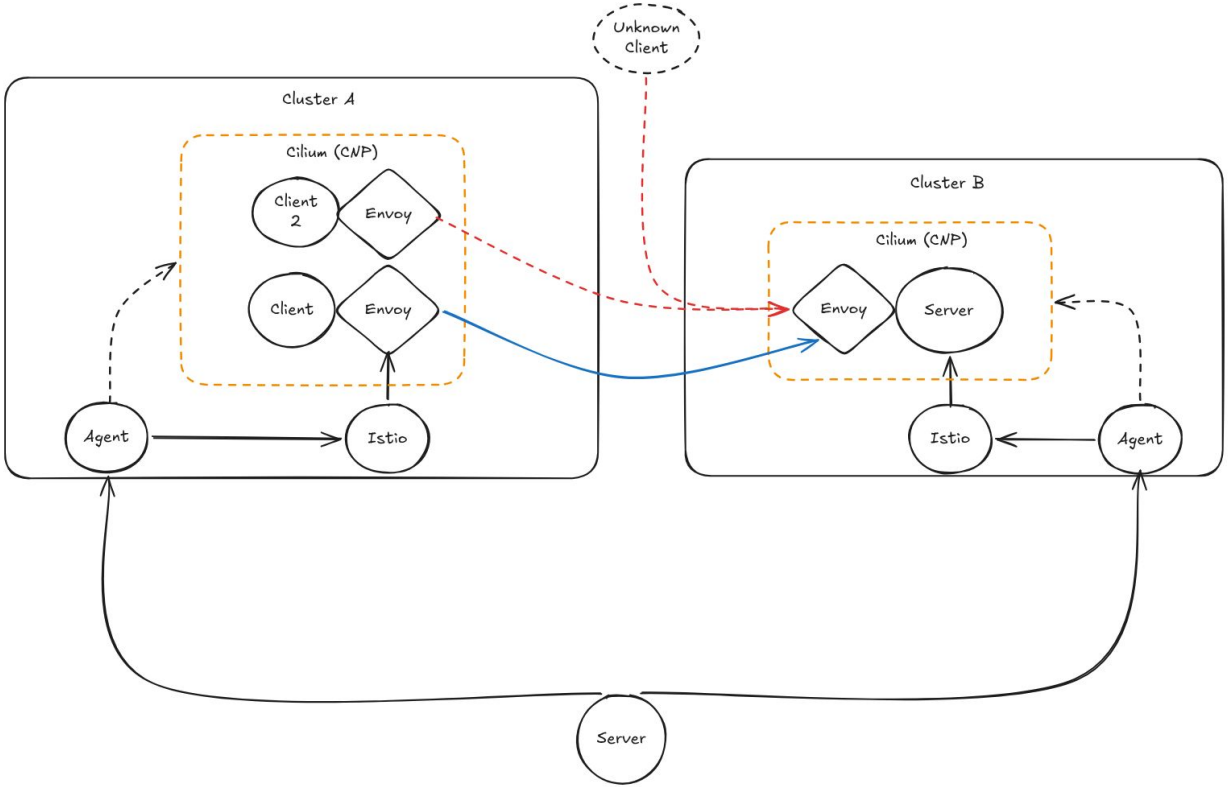
Zaira Shaikh - Senior Engineer
Michael Bolot - Senior Staff Engineer



Legacy At GEICO

- Mix of VMs (Windows and Linux) and AKS Clusters, mostly on Azure
- Many varied sources of traffic enforcement:
  - Cloud provider resources (Azure NSGs, Azure Firewall)
  - On-premises firewalls
  - In-cluster resources
- Problems:
  - No unified definition of policy
  - Manual configuration management
  - Inconsistent enforcement
  - Complex network topology

What is the New World?



How does the New World solve our problems?

- No unified definition of policy
  - All networking policy is defined in the API and enforced through abstracted dataplane drivers.
- Manual Configuration Management
  - API driven approach - Allows for API Tenancy/Self-Service and modern configuration management
- Inconsistent Enforcement
  - Dataplane agents own most of the potential enforcement methods
- Complex network topology
  - Everything has direct (no NAT) L3 connectivity by default.
  - There are very few enforcement methods not controlled by the API

Building a Secure Service Registry - Problems

- Discovery
  - How does a client find a server?
  - Harder than you think. Some applications need IP addressability.
- Security
  - How do we provide a stable, universal source of identity?
  - How do we ensure only allowed connections can complete?
  - How do we enforce mutual TLS on every connection?
- User Experience
  - Users need to configure this without being networking/security experts
  - Tenants should be able to define configuration for only the resources they own

Building a Secure Service Registry - Problems

- Two Main Solutions
- SPIRE
  - Provides stable source of identity based on ServiceAccount
  - Node Level attestation uses Cloud Provider Identity or TPM
- Istio
  - Provides the service registry and allows enforcement with AuthorizationPolicies
  - ServiceEntries and WorkloadEntries to identify cross-cluster endpoints
  - Selectively capturing DNS allows for custom hostnames
- End users don’t use either of these directly - they are implementation details

What about what Istio can’t cover?

- Envoy does not handle UDP
  - We want to block all UDP in our environment except DNS
  - Sidecar bypass is possible for outbound requests. How do we prevent data exfiltration?
- Not everything can run with an Istio sidecar. We need to apply policies to system components.
- **Solution: Cilium**
- Cilium acts as a software firewall for Kubernetes pods and offers rich policy control at L3/L4 with CiliumNetworkPolicies.
- Does not require pod mutation to enforce policy.

Why not use Cilium for everything?

- Redundant Envoy
  - Cilium supports L7 policy enforcement with an Envoy proxy per node
  - Cilium's Envoy is less configurable than Istio
- Identity Explosion
  - Cilium defines pod identity using labels
  - Cilium docs recommend limiting the set of labels used for identities
  - We want to allow our users to choose the right labels/selectors for their workloads
- Reduced Feature Set
  - L3/L4 Policies are a too primitive
  - Istio can provide mTLS and Identity based authorization

Using Cilium to hem the edges

1. Create a default-deny CiliumNetworkPolicy in each tenant namespace
   a. Traffic to system pods (istiod/kube-dns) is allowed
   b. Everything else is denied
2. Apply a known, constant label to selected workloads
   a. Prevents identity explosion, allows for the default-deny to block unknown pods
3. Create a CiliumClusterwideNetworkPolicy allowing internal connections
   a. Internal is defined by our known IP Ranges
   b. Policy applies to everything matching the known label
4. Policy Enforcement for Un-meshed resources
   a. Nodes and some pods don’t have sidecars, but still need networking policy