The Complete Guide to Service Mesh

A REGULARLY UPDATED GUIDE TO ALL THINGS SERVICE MESH
# Table of Contents

**YOUR GUIDE TO THE GUIDE**

## 01 The Path To Service Mesh
1. The origin story

## 02 Understanding Service Mesh
1. What is a service mesh?  
2. Why developers love it  
3. Why operations loves it  
4. How it enables DevOps  
5. Problems a service mesh solves  
6. The service mesh landscape

## 03 What A Service Mesh Provides
1. Observability  
   - Deep system insights  
   - An intuitive UI  
2. Security  
   - mTLS  
   - Ingress/Egress  
3. Operational control  
   - RBAC  
   - CI/CD  
4. A powerful developer toolbox  
   - Chaos testing  
   - Tracing
Understanding Service Mesh

Easing the management of containerized applications at runtime
In the beginning, we had packets and packet-switched networks.

Everyone on the Internet — all 30 of them — used packets to build addressing, session establishment/teardown. And then, they’d need a retransmission scheme. Then, they’d build an ordered byte stream out of it.

Eventually, they realized they had all built the same thing. The RFCs for IP and TCP standardized this, operating systems provided a TCP/IP stack, so no application ever had to turn a best-effort packet network into a reliable byte stream.
We took our reliable byte streams and used them to make applications. Turns out that a lot of those applications had common patterns again — they requested things from servers, and then got responses. So, we separated these request/responses into metadata (headers) and body.

HTTP standardized the most widely deployed request/response protocol. Same story. App developers don't have to implement the mechanics of requests and responses. They can focus on the app on top.
There's a newer set of functionality that you need to build a reliable microservices application. Service discovery, versioning, zero-trust... all the stuff popularized by the Netflix architecture, by 12-factor apps, etc. We see the same thing happening again - an emerging set of best practices that you have to build into each microservice to be successful.

So, service mesh is about putting all that functionality again into a layer, just like HTTP, TCP, packets, that's underneath your code, but creating a network for services rather than bytes.

Questions? Let's start with the most basic one.
What exactly is a service mesh?
It’s designed to handle a high volume of service-to-service communications using application programming interfaces (APIs). A service mesh ensures that communication among containerized application services is fast, reliable and secure. The mesh provides critical capabilities including service discovery, load balancing, encryption, observability, traceability, authentication and authorization, and the ability to control policy and configuration in your Kubernetes clusters.

**What is a Service Mesh?**

A service mesh is a transparent infrastructure layer that sits between your network and application.
What is a Service Mesh?

Service mesh helps address many of the challenges that arise when your application is being consumed by the end user. Being able to monitor what services are communicating with each other, if those communications are secure, and being able to control the service-to-service communication in your clusters are key to ensuring applications are running securely and resiliently.

**service mesh** noun

A service mesh is a transparent infrastructure layer that sits between your network and application.
SERIOUSLY, WHAT THE HELL ARE THEY UP TO?

The self-contained, ephemeral nature of microservices comes with some serious upside, but keeping track of every single one is a challenge — especially when trying to figure out how the rest are affected when a single microservice goes down. The end result is that if you’re operating or developing in a microservices architecture, there’s a good chance part of your days are spent wondering what the hell your services are up to.

With the adoption of microservices, problems also emerge due to the sheer number of services that exist in large systems. Problems like security, load balancing, monitoring and rate limiting that had to be solved once for a monolith, now have to be handled separately for each service.

Service mesh helps address many of these challenges so engineering teams, and businesses, can deliver applications more quickly and securely.
If you're reading this, you're probably responsible for making sure that you and your end users get the most out of your applications and services. In order to do that, you need to have the right kind of access, security and support. That’s probably why you started down the microservices path.

If that's true, then you've probably realized that microservices come with their own unique challenges, such as:

1. Increased surface area that can be attacked
2. Polyglot challenges
3. Controlling access for distributed teams developing on a single application

That's where a service mesh comes in.
A service mesh is an infrastructure layer for microservices applications that can help reduce the complexity of managing microservices and deployments by handling infrastructure service communication quickly, securely and reliably. (More on that later.)

Service meshes are great at solving operational challenges and issues when running containers and microservices because they provide a uniform way to secure, connect and monitor microservices. (More on that later, too.)

Here’s the point: a good service mesh keeps your company’s services running the way they should. A service mesh designed for the enterprise, like Aspen Mesh, gives you all the observability, security and traffic management you need — plus access to engineering and support, so you can focus on adding the most value to your business.

And that is good news for DevOps.
It’s happening, and it’s happening fast.

Companies are transforming internal orgs and product architectures along a new axis of performance. They’re finding more value in iterations, efficiency and incremental scaling, forcing them to adopt DevOps methodologies. This focus on time-to-market is driving some of the most cutting-edge infrastructure technology that we have ever seen. Technologies like containers and Kubernetes, and a focus on stable, consistent and open APIs allow small teams to make amazing progress and move at the speeds they require. These technologies have reduced the friction and time to market.

The adoption of these technologies isn’t perfect, and as companies deploy them at scale, they realize that they have inadvertently increased complexity and de-centralized ownership and control. In many cases, it’s challenging to understand the entire system.
A service mesh enables DevOps teams by helping manage this complexity. It provides autonomy and freedom for development teams through a stable and scalable platform, while simultaneously providing a way for platform teams to enforce security, policy and compliance standards.

**This empowers your development teams to make choices based on the problems they are solving rather than being concerned with the underlying infrastructure.** Dev teams now have the freedom to deploy code without the fear of violating compliance or regulatory guidelines, and platform teams can put guardrails in place to ensure applications are secure and resilient.
Why Developers Love It

Having a service mesh means developers no longer have to worry about managing infrastructure. Now, they can focus on creating features that add value, not spend their time managing existing ones. On top of that, access to features like canary testing and fault injection enable easy testing and debugging.
Why Operators Love It

A service mesh provides platform owners with observability, baked-in security and the ability to easily manage policy and configuration. This means they can help the business more quickly deliver applications and increase the uptime of those applications.
Why Business Loves It

Service mesh enables **progressive delivery** - decoupling deployment from service activation, so developers can keep developing without restrictions, but the business decides how new features are rolled out.
Service Mesh Landscape

A service mesh overlaps, complements, and in some cases, replaces many tools that are commonly used to manage microservices. Let’s explore.
Many technologies are involved in the service mesh landscape. This is a view of how service mesh fits with other commonly used container tools.
What a Service Mesh Provides
What a Service Mesh Provides

There’s a lot of value a service mesh can provide. But let’s spend some time diving into four key areas.

**Observability**
A service mesh takes system monitoring a step further by providing observability. Monitoring reports overall system health, while observability focuses on highly granular insights into the behavior of systems along with rich context.

**Security**
A service mesh provides security features aimed at securing the services inside your network and quickly identifying any compromising traffic entering your cluster.

**Operational Control**
A service mesh allows security and platform teams to set the right macro controls to enforce access controls, while allowing developers to make customizations they need to move quickly within these guardrails.

**A Powerful Developer Toolbox**
A service mesh removes the burden of managing infrastructure from the developer, and provides developer-friendly features such as distributed tracing and easy canary deploys.
Why it matters
Kubernetes seemed like the way to rapid iteration and quick development sprints, but the promise and the reality of managing containerized applications at scale are two very different things.

Docker and Kubernetes enable you to more easily build and deploy apps. But it's often difficult to understand how those apps are behaving once deployed.

Service mesh provides tracing and telemetry metrics that make it easy to understand your system and quickly root cause any problems.
Try it out with Aspen Mesh
Aspen Mesh provides distributed tracing capabilities through Jaeger, and metrics through Prometheus. Our platform provides hosted Jaeger and Prometheus so we manage all of the datastore for you. If you love the idea of getting fine-grained observability through tracing and metrics, but don’t love the idea of managing all that data, give the Aspen Mesh platform a look.

TRY VISUALIZING TRAFFIC & TRACING WITH ASPEN MESH IN 10 MINUTES - FREE CLICK HERE
Why it matters
A service mesh is uniquely positioned to gather a trove of important data from your services. The sidecar approach places an Envoy next to every pod in your cluster, which then surfaces telemetry data up to the Istio control plane. This is great, and it also means a mesh will gather more data than is useful. The key is surfacing only the data you need to confirm the health and security status of your services. A good UI solves this problem, and it also lowers the bar on the engineering team, making it easier for more members of the team to understand and control the services in your organization’s architecture.
About our UI
The Aspen Mesh dashboard provides at-a-glance views of performance and security posture, as well as the ability to see service details.
Go from macro to micro view – zoom from service view into namespace and workload details

- View service details to quickly identify failures or bottlenecks – view and sort services by latency, error rate and health scores
- Understand the health of your mesh and be alerted to any mesh configuration errors by our Istio Vet tool
Why it matters
Securing ephemeral microservices is hard. There are a multitude of tools that address microservices security, but service mesh is the most elegant solution for addressing encryption of on-the-wire traffic within the network.

Service mesh provides defense with mutual TLS encryption of the traffic between your services. The mesh can automatically encrypt and decrypt requests and responses, removing that burden from the application developer. It can also improve performance by prioritizing the reuse of existing, persistent connections, reducing the need for the computationally expensive creation of new ones. With service mesh, you can secure traffic over the wire and also make strong identity-based authentication and authorizations for each microservice.
How Aspen Mesh can help

Aspen Mesh makes it easy to configure and manage mTLS. You can see whether mTLS is enabled and working between each of your services and get immediate alerts if security status changes.
Why it matters
Service mesh adds a layer of security that allows you to monitor and address compromising traffic as it enters the mesh. Istio integrates with Kubernetes as an ingress controller and takes care of load balancing for ingress. This allows you to add a level of security at the perimeter with ingress rules. Egress control allows you to see and manage external services and control how your services interact with them.
Get Aspen Mesh expertise and support
We can help you with your integration — and the other stuff that comes after that. Our team is fanatical about doing the hard work so you don’t have to, and our expert engineers (meshperts) are here to support you all the way through integration and into production.
Why it matters
A strong Role Based Access Control (RBAC) system is arguably one of the most critical requirements in large engineering organizations, since even the most secure system can be easily circumvented by overprivileged employees. Restricting privileged users to least privileges necessary to perform job responsibilities, ensuring access to systems are set to “deny all” by default, and ensuring proper documentation detailing roles and responsibilities are in place is one of the most critical security concerns in the enterprise.
Aspen Mesh solutions
Aspen Mesh provides Istio Vet, which is designed to warn you of incorrect or incomplete configuration of your service mesh, and provide guidance to fix it. Istio Vet prevents misconfigurations by refusing to allow them in the first place. Global Istio configuration resources require a different solution, which is addressed by the Traffic Claim Enforcer solution.
Why it matters
Modern Enterprises manage their applications via an agile, iterative lifecycle model. Continuous Integration and Continuous Deployment systems automate the build, test, deploy and upgrade stages. Service mesh adds power to your CI/CD systems, allowing operators to build fine-grained deployment models like canary, A/B, automated dev/stage/prod promotion and rollback. Doing this in the service mesh layer means the same models are available to every app in the enterprise without app modification. You can also up-level your CI testing using techniques like traffic mirroring and fault injection to expose every app to complicated, hard-to-simulate fault patterns before you encounter them with real users.
Aspen Mesh solutions
Aspen Mesh provides an open framework that can integrate with your CI system of choice.

You can use Aspen Mesh canary testing features that show the health of your services to help decide if you should deploy changes to production.
Why it matters

Distributed tracing lets you track activities resulting from requests to an application, so you can trace request paths, find latency in those pathways, and identify components creating issues or bottlenecks. It’s becoming an increasingly key component for debugging and understanding application dependencies in microservices architectures.
Aspen Mesh integrations
Aspen Mesh provides hosted Jaeger as part of the Aspen Mesh platform. This means you get the flexibility of OpenTracing standards and the Aspen Mesh Platform manages the datastore so you get all the advantages without the costs.
Why it matters
Chaos testing is regularly exposing your applications and platforms to failures that exercise and demonstrate their resilience capabilities. Organizations that use chaos testing know where they stand and what kinds of disruptions they can tolerate before they happen. A service mesh allows you to inject chaos into your system, starting with localized and isolated chaos and expanding to encompass the entire architecture. Fallbacks and circuit breakers provided by a service mesh can provide resilience to chaos, whether it originates from chaos testing or the real world.
Why use the Aspen Mesh toolbox
Without Aspen Mesh, it can be hard to determine the impact of any chaos you introduce - is it only affecting this local microservice or is it affecting the entire app? The Service Graph lets you see at-a-glance how your services are interacting to rapidly identify defective behavior. You can use Aspen Mesh to drill into multiple clusters so you can compare results across clusters, whether those clusters are geographically separated or specialized by purpose like dev vs. prod.
Dev or Ops?

Doesn’t matter, a service mesh makes your job easier.

Whether you own the platform and need to provide resiliency and security for your applications, or you’re a developer who needs a better way to achieve application testing and debugging - a service mesh makes it easier to control and secure microservices.
About

We’re the ones who made this for you — the deck and the service mesh. We provide critical enterprise features in a platform built on top of Istio, so organizations enjoy the benefits of an open source approach without forgoing the features, support and guarantees needed to power enterprise applications.

Aspen Mesh builds on the Istio feature set by providing:

- Advanced policy and configuration options
- Analytics and alerting
- Multi-cluster/multi-cloud capabilities
- An intuitive UI
- A fully hosted SaaS platform
- Complete support from our team of service mesh experts

AND YOU CAN GET ALL OF THESE WITH (YOU GUessed IT) OUR FREE BETA.

CLICK HERE
Thank you

DON'T FORGET TO CHECK BACK FOR UPDATES
Not done yet? Cool.

WE'VE GOT SOME ADDITIONAL CONTENT FOR YOU