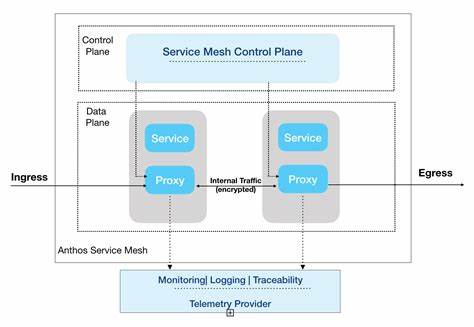
**Service Mesh:**

* A service mesh is a software layer that handles all communication between services in applications. This layer is composed of containerized microservices.
* Dedicated infrastructure layer for facilitating service-to-service communication between services and microservices.
* Software layer that communicates between services in application.
* Composed of containerized microservices.
* Application scales, microservices increases. So, this becomes challenging to monitor services between the containers.
* To manage connections, a service mesh provides new features like monitoring, logging, tracing, and traffic control.
* Independent of each service’s code, allows it to work across network boundaries with multiple service management systems.
* Benefits:
  + Load Balancing:
  + Traffic Management:
  + Traffic Splitting:
  + Service Discovery:
  + Security:

**Work process:**

* Consist of network proxies paired with each service in an application and a set of task-management processes.
* Proxies are called the data plane and the management processes are called the control plane.
* The data plane intercepts different calls between different services and processes them; the control plane is the brain of the mesh that coordinates the behavior of proxies and provides APIs for operations and maintenance personnel to manipulate and observe the entire network.
* It routes the requests from one service to another one, to optimize how all the parts of application are working together.

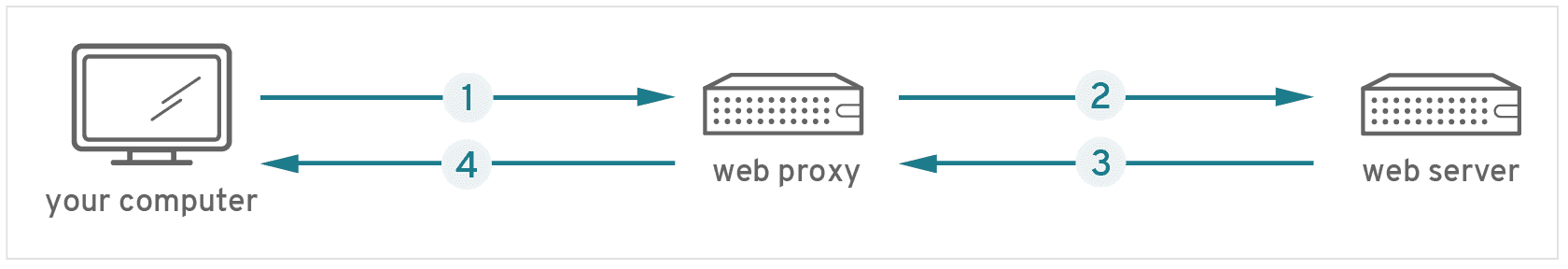


**Difference between it and microservices:**

* Service-to-service communication by microservices.
* The logic governing communication can be coded into each service without a service mesh layer—but as communication gets more complex, a service mesh becomes more valuable.
* For cloud-native apps built in a microservices architecture, a service mesh is a way to comprise a large number of discrete services into a functional application.

**How does it work?**

* a service mesh takes the logic governing service-to-service communication out of individual services and abstracts it to a layer of infrastructure.



* accessing a webpage from a work computer. Steps:
  + the request for the page first received by company’s web proxy.
  + Then proxy will measure the security concern and send the request to server that hosts the page for which you have requested.
  + Next, page ill again return to the proxy, and it again check the security measure.
  + Then proxy will finally send the page to the laptop.
* Without a service mesh, each microservice needs to be coded with logic to govern service-to-service communication, which means developers are less focused on business goals.
* Sidecar: The requests are routed between microservices through proxies in their OWN infrastructure layer. So, that make up a service mash are sometimes called “sidecars”, as they are running along each service, rather than within them.
* Performance metrics can suggest ways to optimize communication in the runtime environment.
* Main drivers of Service mesh:
  + Service level Observability
  + Service-level Control