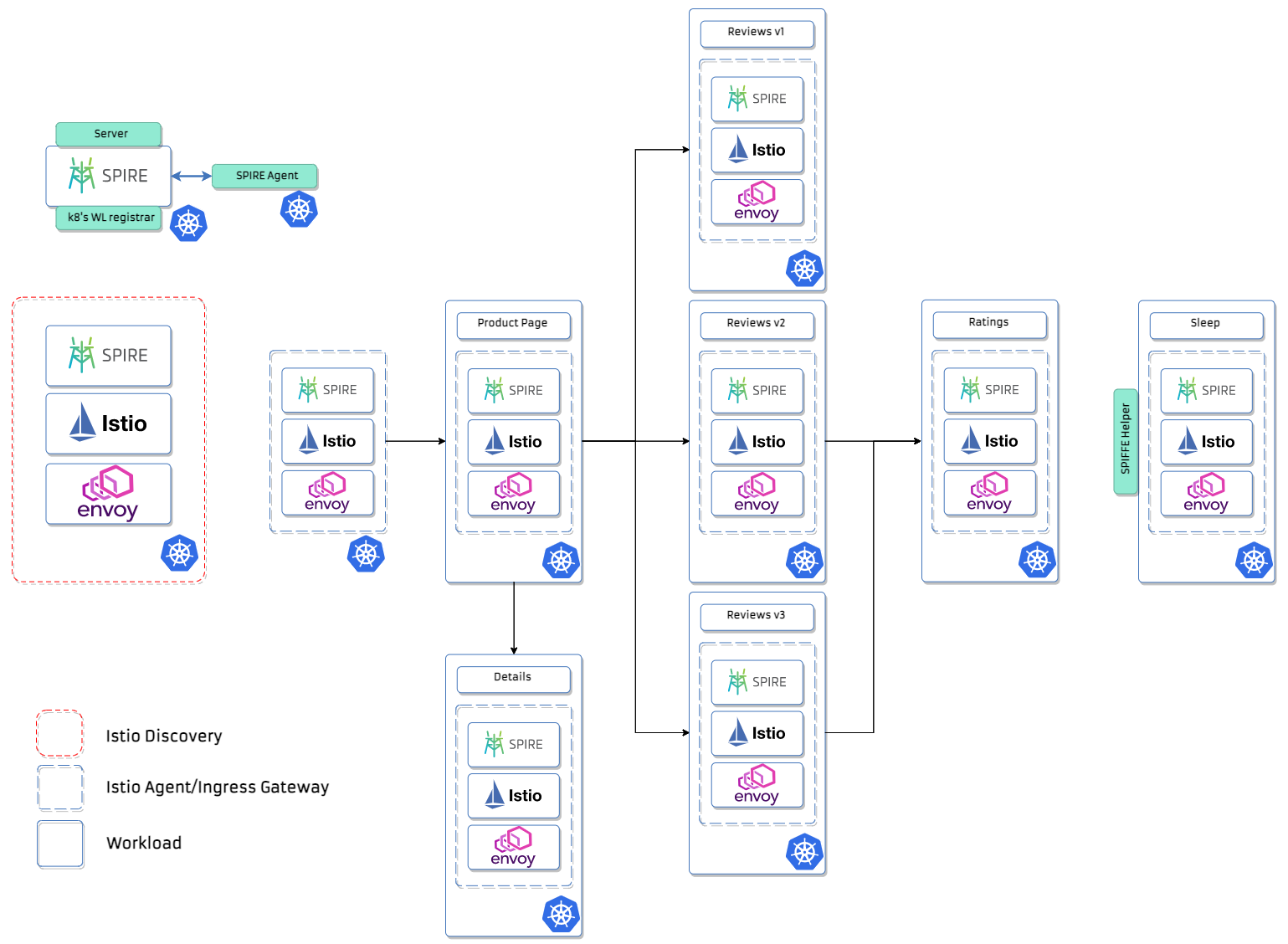
**Connecting two workloads from different Mithril clusters using external disk Spire CA**

This section showcases a ***mTLS*** between two workloads from different **Mithril** clusters within the same *trust domain.* On this use case both clusters have a **SPIRE Server** using the ‘disk’ ***UpstreamAuthority*** plugin that loads ***CA*** credentials from disk, more information about this **SPIRE** setup on the [UpstreamAuthority disk example.](https://github.com/spiffe/spire/blob/v1.0.1/doc/plugin_server_upstreamauthority_disk.md)

**Overview**

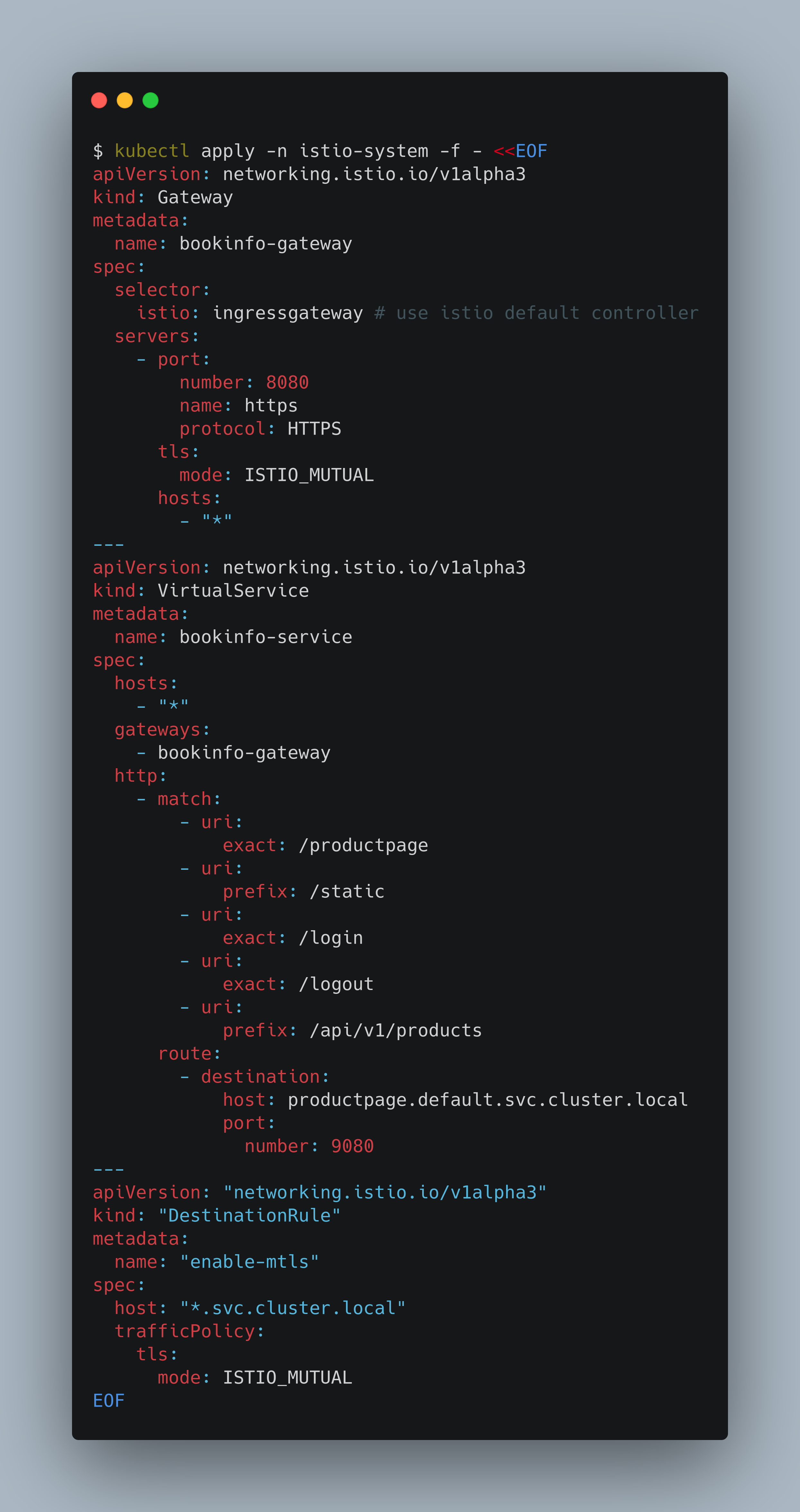


Structure for each Mithril cluster with the bookinfo example.

The Mithril changes in [Istiod](https://istio.io/latest/blog/2020/istiod/) and [Istio Agent](https://istio.io/latest/docs/reference/commands/pilot-agent/) make possible the use of the SPIRE Workload API to fetch identities for every entity within the mesh. The identities for **Istiod**, **Istio Ingress,** **Egress**, and for the workloads are all generated by **SPIRE** by leveraging the k8s workload registrar, and then the **Istio Agent** sidecars of each workload fetch and push their credentials materials to be used by the **Envoy** proxy to communicate and perform ***mTLS*.**

**Configuring Istio Ingress Gateway for *mTLS***

To showcase the ***mTLS*** communication between the workloads we will need to configure a **Gateway,** a **VirtualService** and a **DestinationRule** on the server-side ***Ingressgateway*** for it to require HTTPS on incoming requests from outside the cluster.

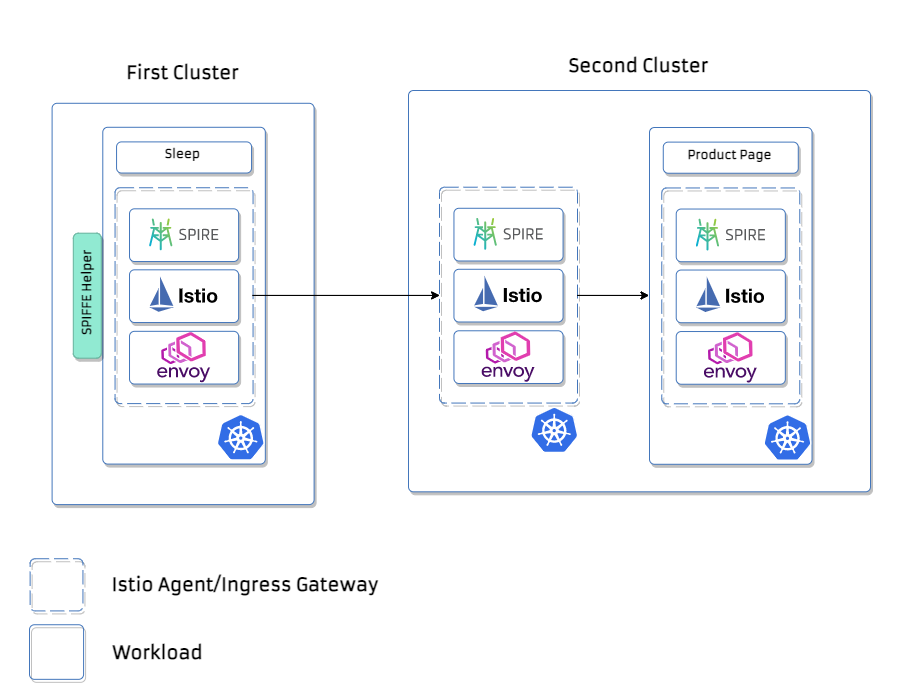
Ingress Configuration for the server-side Mithril cluster

Using this configuration, we route requests from outside the cluster to internal services like the **/*productpage***service from the [bookinfo example](https://istio.io/latest/docs/examples/bookinfo/).

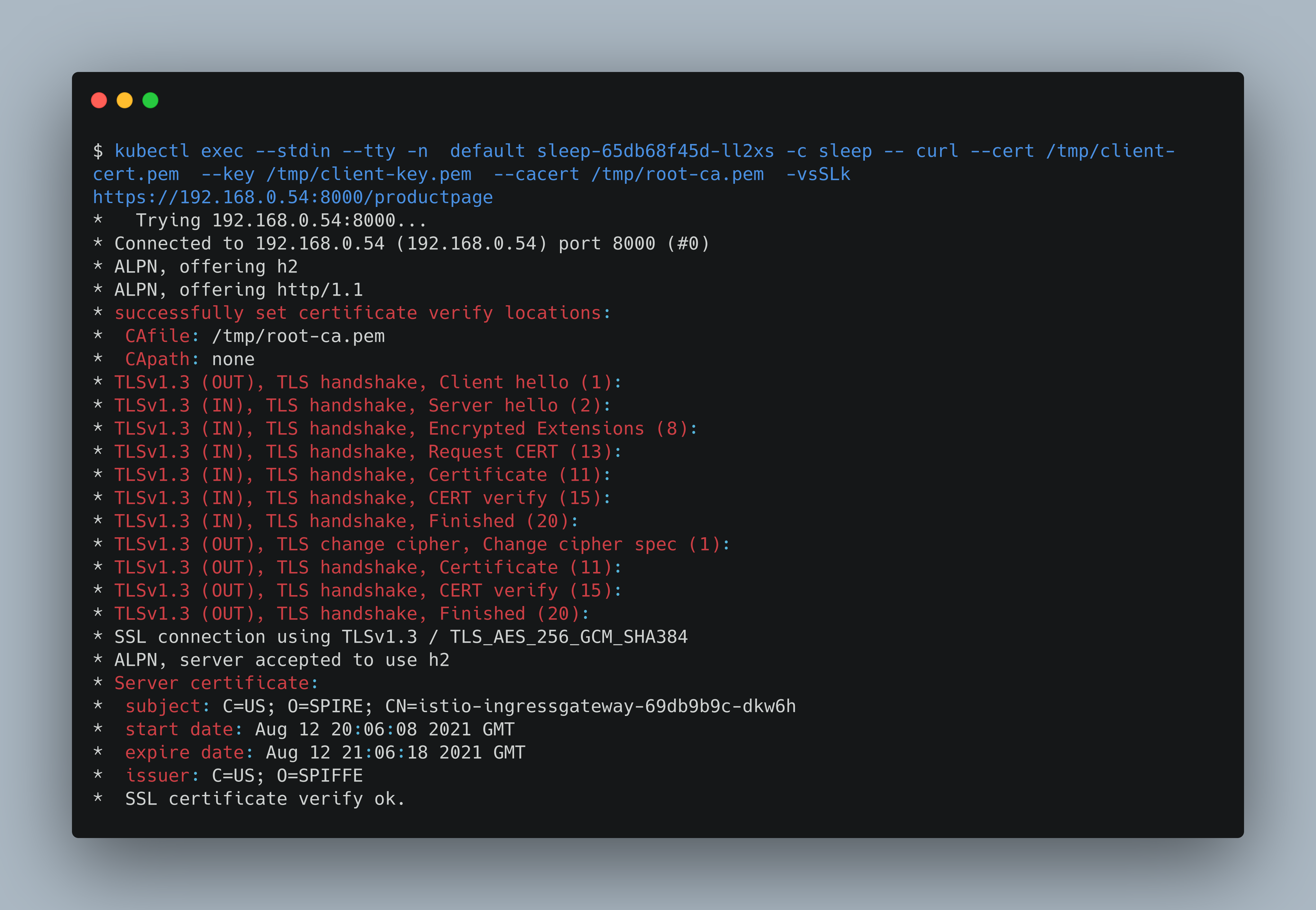
**Perform a curl between the workloads**

We can use the [sleep sa](https://raw.githubusercontent.com/istio/istio/release-1.11/samples/sleep/sleep.yaml)mple to use **curl** and test the connection between the ***sleep*** app from the first **Mithril** cluster and the **Product Page** workload from the server-side cluster. If you have [automatic sidecar injection](https://istio.io/latest/docs/setup/additional-setup/sidecar-injection/#automatic-sidecar-injection) enabled:

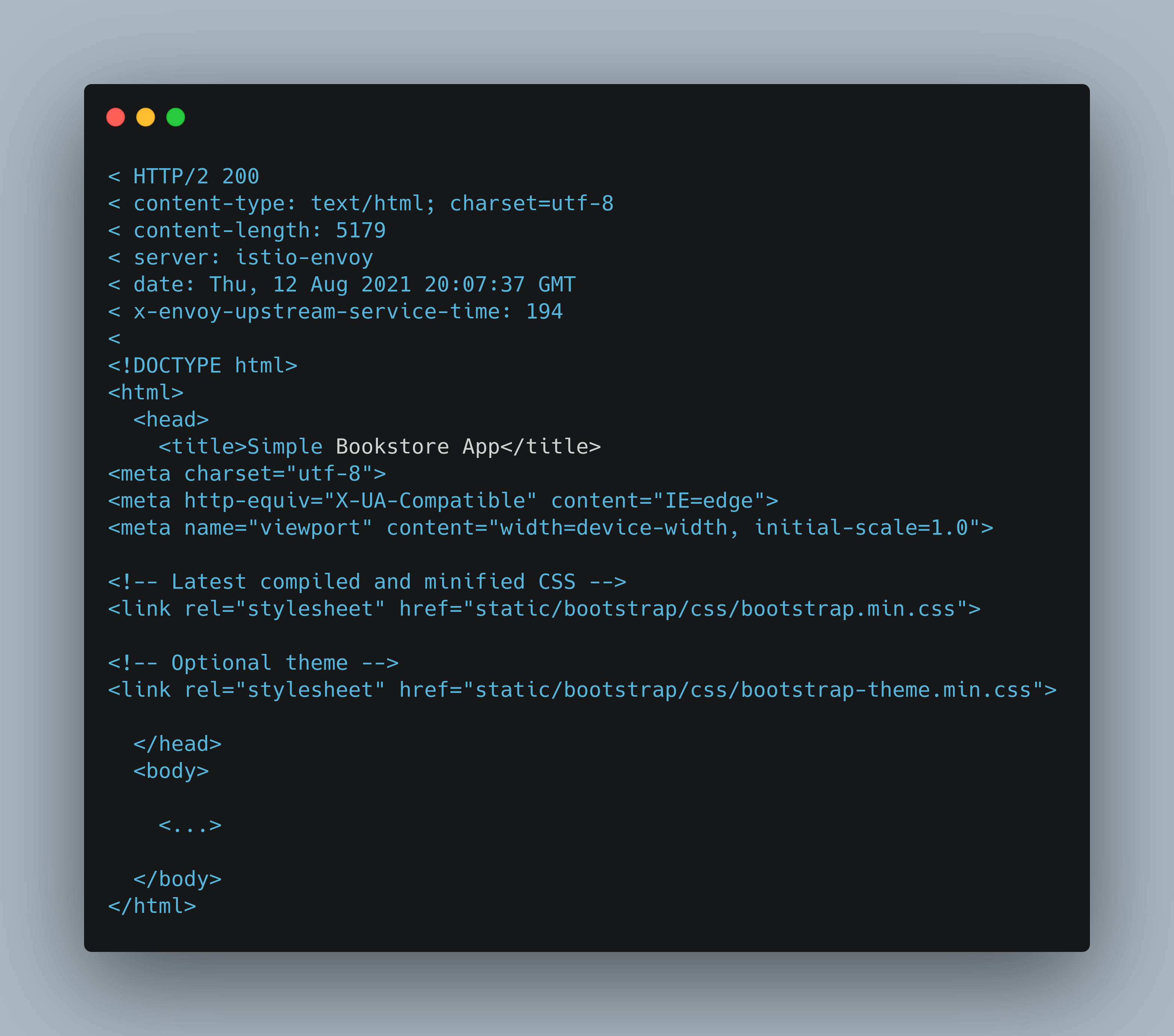




Making usage of the [SPIFFE Helper Utility](https://github.com/spiffe/spiffe-helper) we can fetch the identity for the ***sleep***workload in the client cluster (first Mithril cluster) and use it to ***curl*** using ***mTLS*** tothe **Ingressgateway** of the second cluster (server Mithril cluster) and follow the destination rule configured for the **Product Page**service*.* On this particular example, we have the **Ingressgateway** of the server cluster exposed on the endpoint **192.168.0.54:8000.**



Checking the response from the server:



**Inspecting the credential material of the workloads**

