Kubernetes Networking

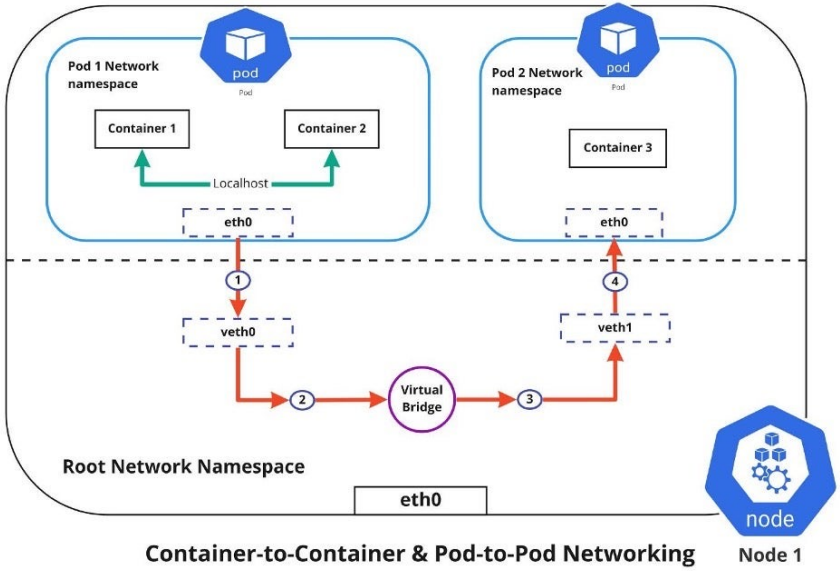
Learn The Ins And Outs Of Networking In K8s

Michael Levan

Trainer, Course Creator, and coach

Pearson

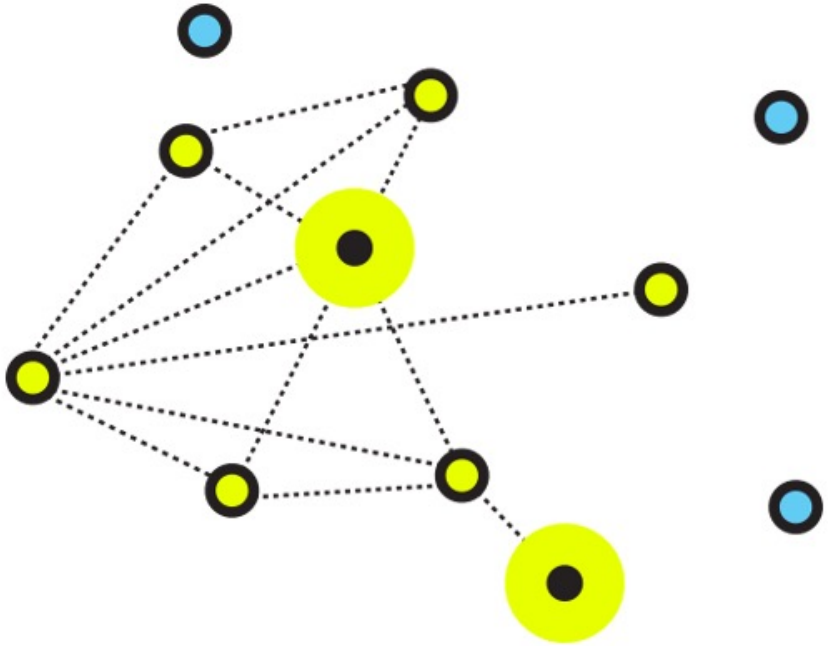
Section 1: Kubernetes Host Networking



- Cluster Networking
- Cloud Networking
- Static Pods

Networking Crash Lesson

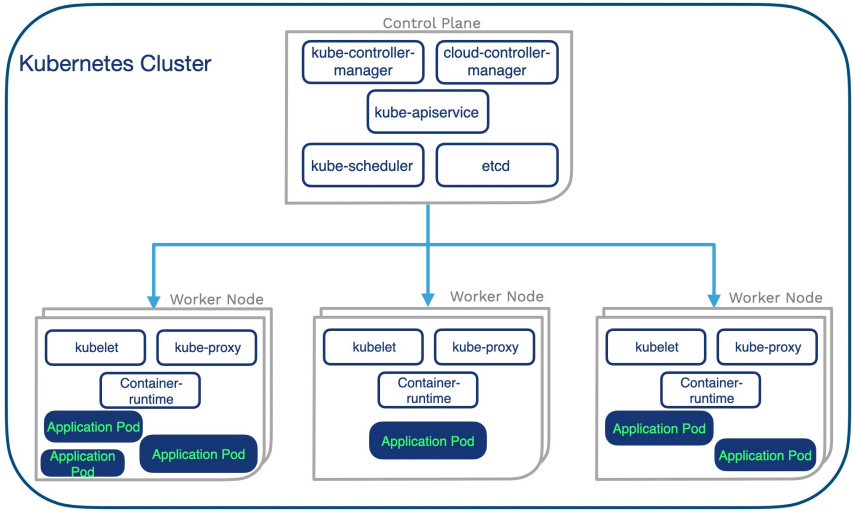
- CIDRs
- IP’s
- Subnets
- OSI model



Cluster Networking

- The cluster itself
- IP addresses and CIDR’s for the Worker Nodes
- IP addresses and CIDR’s for the Control Planes

The Cluster Itself



- Control Planes
- Worker Nodes
- Subnets and how it's all connected

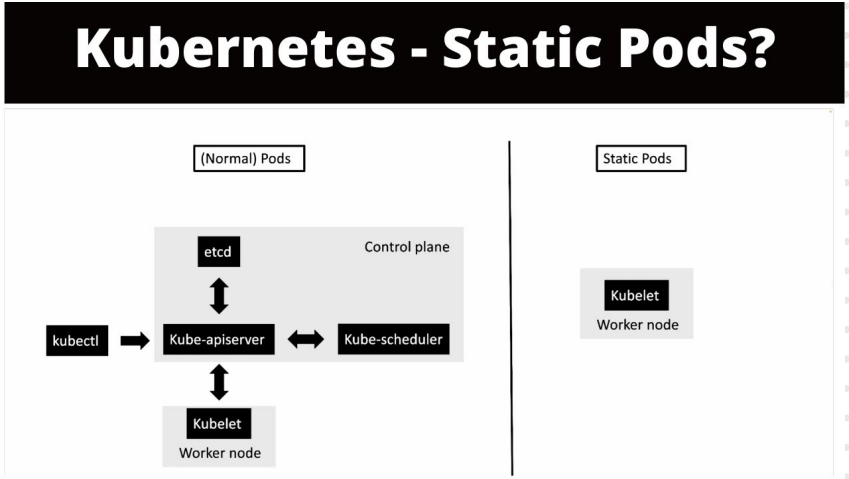
Quick Note On Kube-Proxy and CoreDNS

- kube-proxy: Runs on every node and is responsible for local cluster networking.

- CoreDNS: The cluster's DNS service has a static IP address that is hard-coded into every Pod on the cluster

Static Pods When The Cluster Is Created

- Bound to the Control Plane



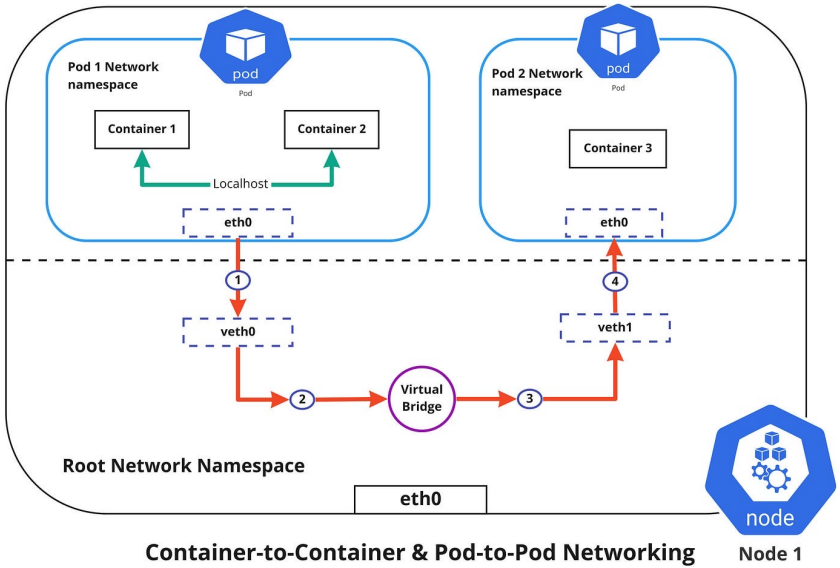
Bare Metal vs VM (Cloud Later)

- Everything is managed by you

Cloud Networking

- Switches, routers, firewalls, etc are all virtual

Networking With Managed Kubernetes



Container-to-Container & Pod-to-Pod Networking Node 1

- Managed by you

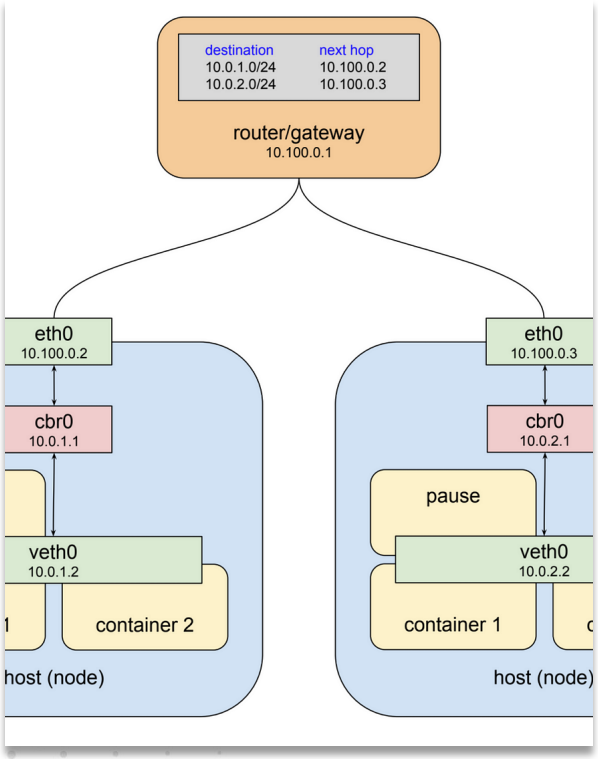
Secure Cluster Networking

- Proper firewall rules
- Private IPs for the Control Plane

Lab

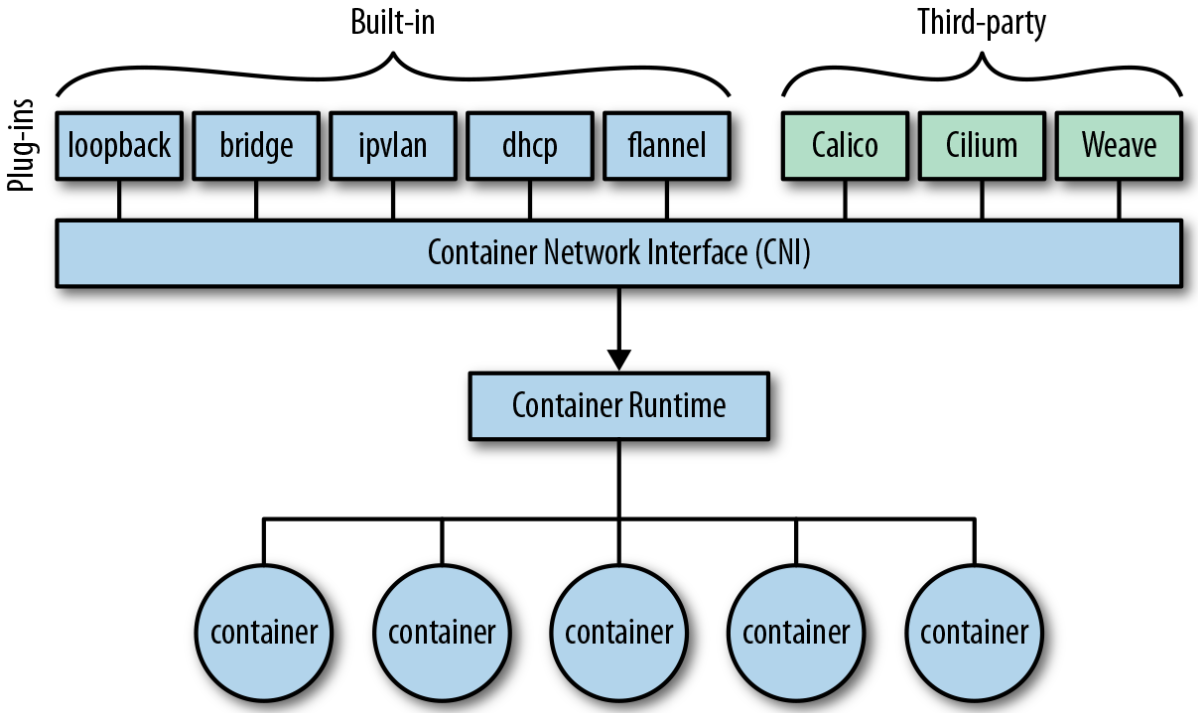
- Cluster networking in the cloud (Azure or AWS)

Section 2: Kubernetes Internal Networking

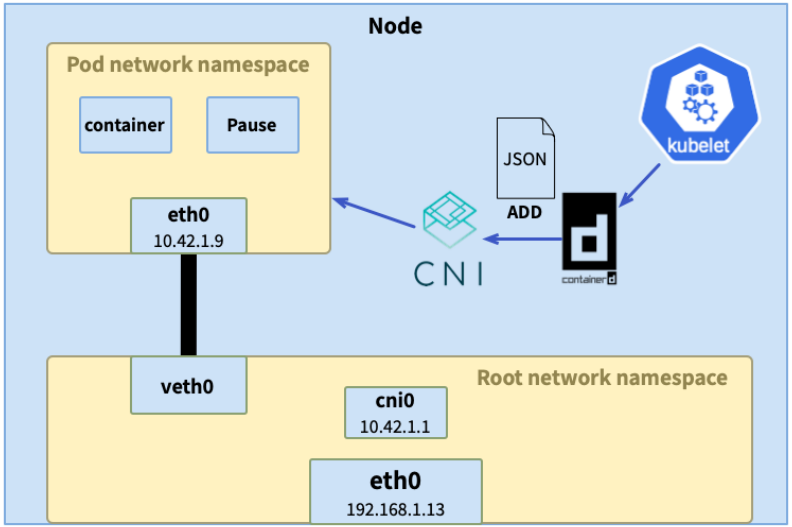


- CNI
- kube-proxy
- Pod networking
- Pod and Service communication
- Ingress
- eBPF

Container Network Interface (CNI)



Implementing a CNI



- Options (available CNIs)
- Installation methods

CoreDNS

CoreDNS

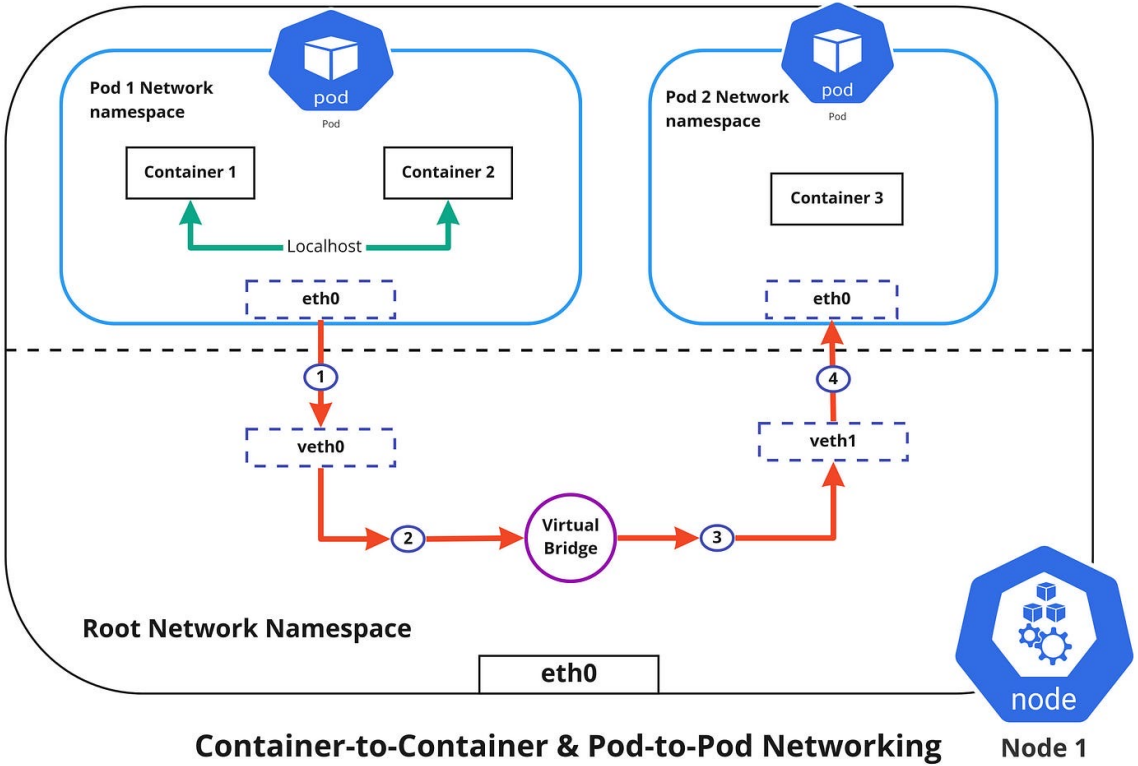
Pod Networking

- How Pods get IP addresses
- Lease times
- Container IPs
- Sidecar IPs

Service Networking

- How services get Ips
- How services get DNS
- How services talk to Pods

Pod To Pod Communication

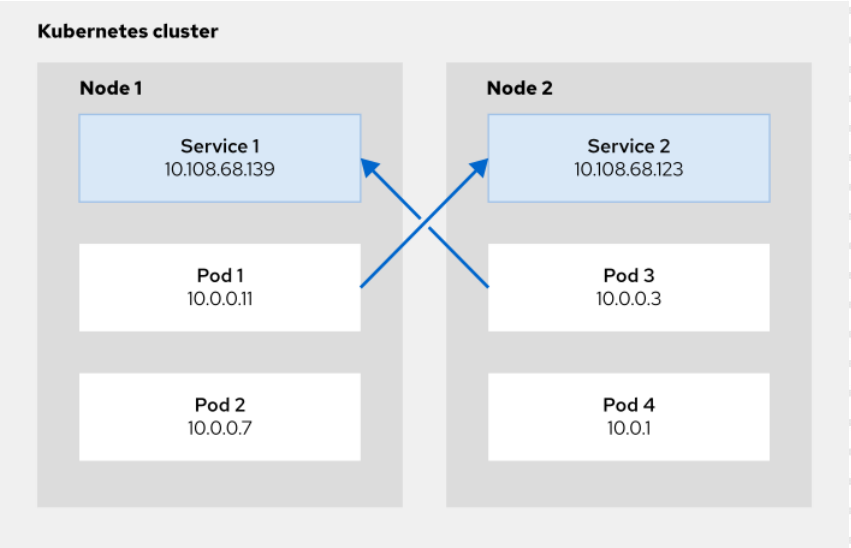


Container-to-Container & Pod-to-Pod Networking

Node 1

Service To Service Communication

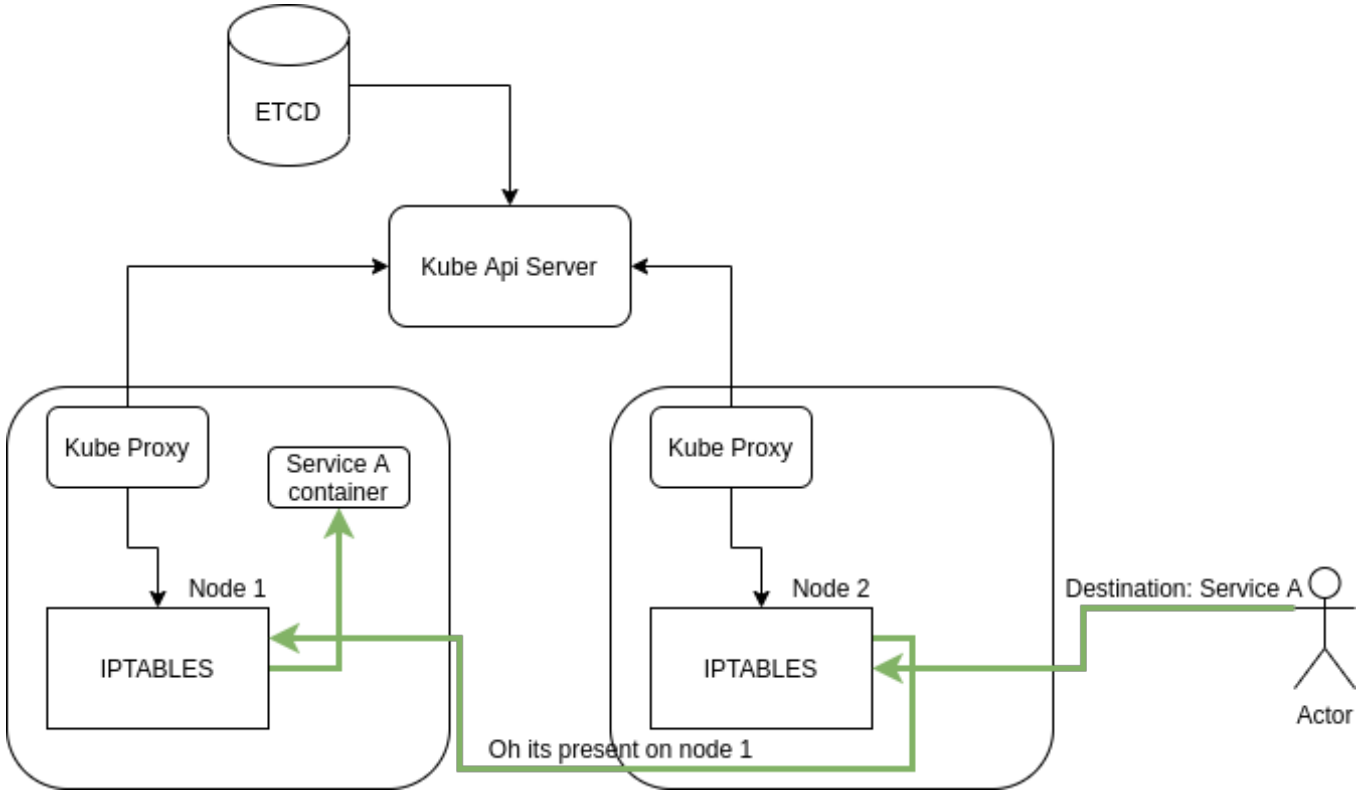
- Via IP or DNS



How To Secure It (It’s all Unencrypted)

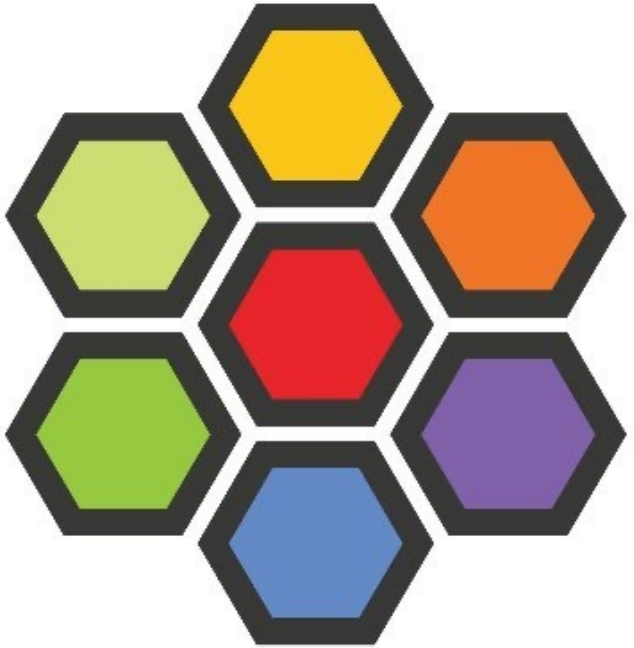
- Service Mesh for Services
- eBPF for Pods

Kube-Proxy

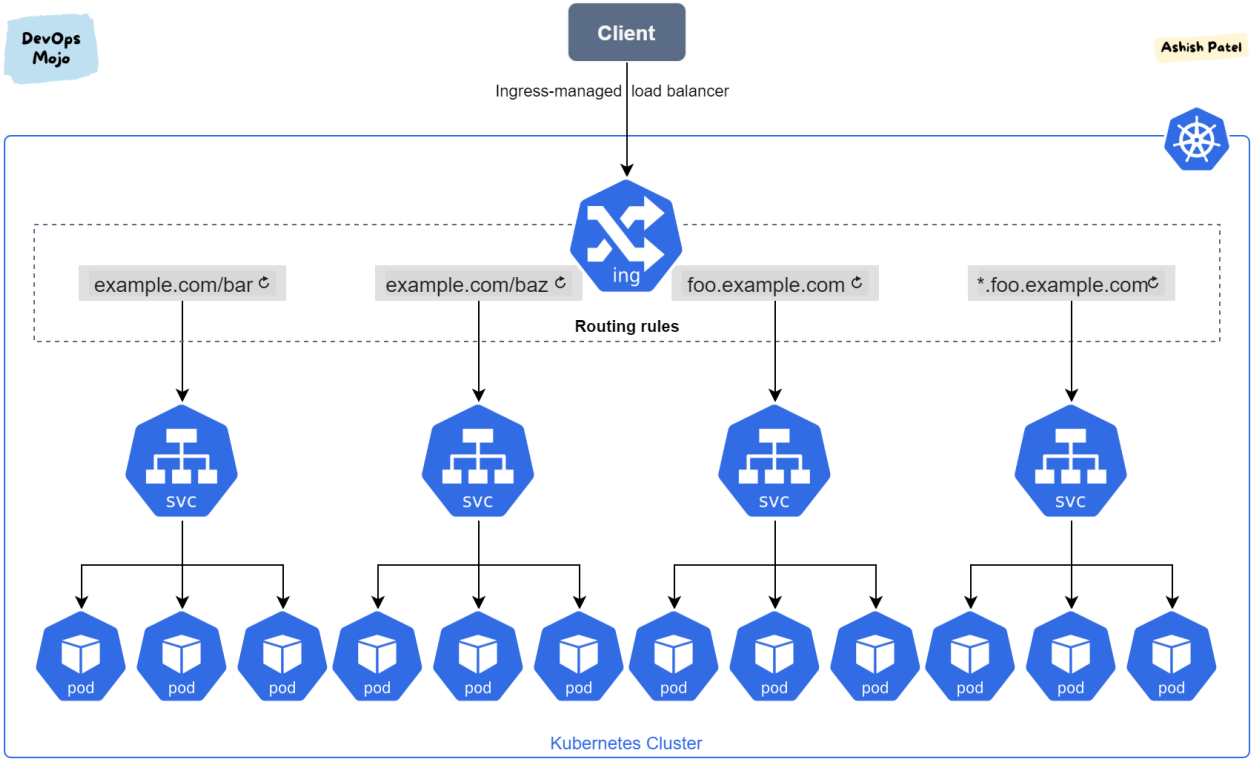

ETCD
Kube Api Server
Kube Proxy
Service A container
Node 1
IPTABLES
Kube Proxy
Node 2
IPTABLES
Destination: Service A
Actor
Oh its present on node 1

eBPF

- Secure Pod communication with a security-centric CNI



Ingress



Pearson

Ingress Implementation

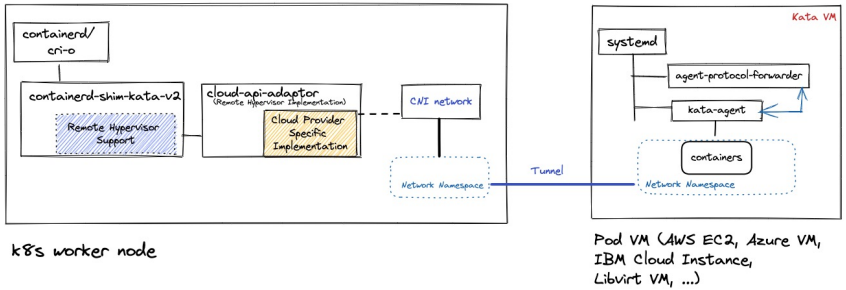
https://kubernetes.io/docs/concepts/services-networking/ingress-controllers/

- AKS Application Gateway Ingress Controller is an ingress controller that configures the Azure Application Gateway.
- Alibaba Cloud MSE Ingress is an ingress controller that configures the Alibaba Cloud Native Gateway, which is also the commercial version of Higress.
- Apache APISIX ingress controller is an Apache APISIX-based ingress controller.
- Avi Kubernetes Operator provides L4-L7 load-balancing using VMware NSX Advanced Load Balancer.
- BFE Ingress Controller is a BFE-based ingress controller.
- Cilium Ingress Controller is an ingress controller powered by Cilium.
- The Citrix ingress controller works with Citrix Application Delivery Controller.
- Contour is an Envoy based ingress controller.
- Emissary-Ingress API Gateway is an Envoy-based ingress controller.
- EnRoute is an Envoy based API gateway that can run as an ingress controller.
- Easegress IngressController is an Easegress based API gateway that can run as an ingress controller.
- F5 BIG-IP Container Ingress Services for Kubernetes lets you use an Ingress to configure F5 BIG-IP virtual servers.
- FortiADC Ingress Controller support the Kubernetes Ingress resources and allows you to manage FortiADC objects from Kubernetes
- Gloo is an open-source ingress controller based on Envoy, which offers API gateway functionality.
- HAProxy Ingress is an ingress controller for HAProxy.
- Higress is an Envoy based API gateway that can run as an ingress controller.
- The HAProxy Ingress Controller for Kubernetes is also an ingress controller for HAProxy.
- Istio Ingress is an Istio based ingress controller.
- The Kong Ingress Controller for Kubernetes is an ingress controller driving Kong Gateway.
- Kusk Gateway is an OpenAPI-driven ingress controller based on Envoy.
- The NGINX Ingress Controller for Kubernetes works with the NGINX webserver (as a proxy).
- The ngrok Kubernetes Ingress Controller is an open source controller for adding secure public access to your K8s services using the ngrok platform.
- The OCI Native Ingress Controller is an Ingress controller for Oracle Cloud Infrastructure which allows you to manage the OCI Load Balancer.
- The Pomerium Ingress Controller is based on Pomerium, which offers context-aware access policy.
- Skipper HTTP router and reverse proxy for service composition, including use cases like Kubernetes Ingress, designed as a library to build your custom proxy.
- The Traefik Kubernetes Ingress provider is an ingress controller for the Traefik proxy.
- Tyk Operator extends Ingress with Custom Resources to bring API Management capabilities to Ingress. Tyk Operator works with the Open Source Tyk Gateway & Tyk Cloud control plane.
- Voyager is an ingress controller for HAProxy.
- Wallarm Ingress Controller is an Ingress Controller that provides WAAP (WAF) and API Security capabilities.

Lab

- Configuring eBPF With Cilium

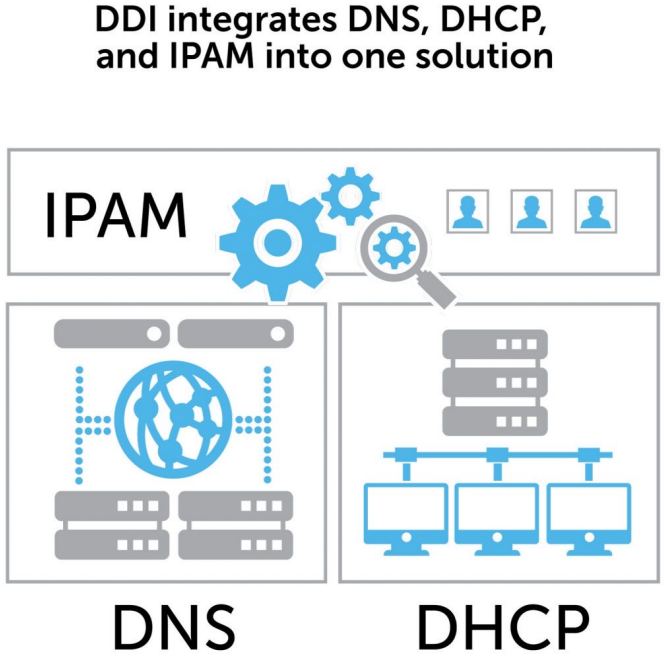
Section 3: Network Troubleshooting



- Host troubleshooting
- Internal k8s network troubleshooting

IPAM For Kubernetes

- Management and assign of IP addresses



DNS Breakdown

- Service: `my-svc.my-namespace.svc.cluster-domain.example`
- Pod: `pod-ip-address.my-namespace.pod.cluster-domain.example`
- Pod created by a deployment exposed as a service:

```
pod-ip-address.deployment-name.my-namespace.svc.cluster-domain.example
```

Host Network Troubleshooting

- Check subnets
- Check private/public IP addresses for Control Plane
- Check firewall rules

Check Overall Network

- Ensure that the subnets and CIDRs are correct

Check Private/Public IPs

- Ensure that you're running private or public IPs
- If you're running private, ensure that you have a way to access them

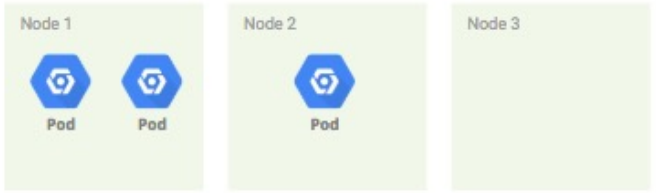
Check Firewall Rules

- Are the proper firewall rules open?
- Ensure that not ALL firewall rules are open

CNI Troubleshooting

- Check that the Pods for the CNI are running
- Check Pod logs
- Check Pod events

Are Pods Running?



Node 1 Could Be Out Of Resources, Overflowing To Node 2

Could Be A Daemon Pod, Could Be Luck

All Pods Could Be In One Node If Resources Exist

No Reason Node 1 Is Where We Start Putting Pods

Pod Port Troubleshooting

- Check which containers are using which Ports
- Ensure that there aren’t any Port conflicts

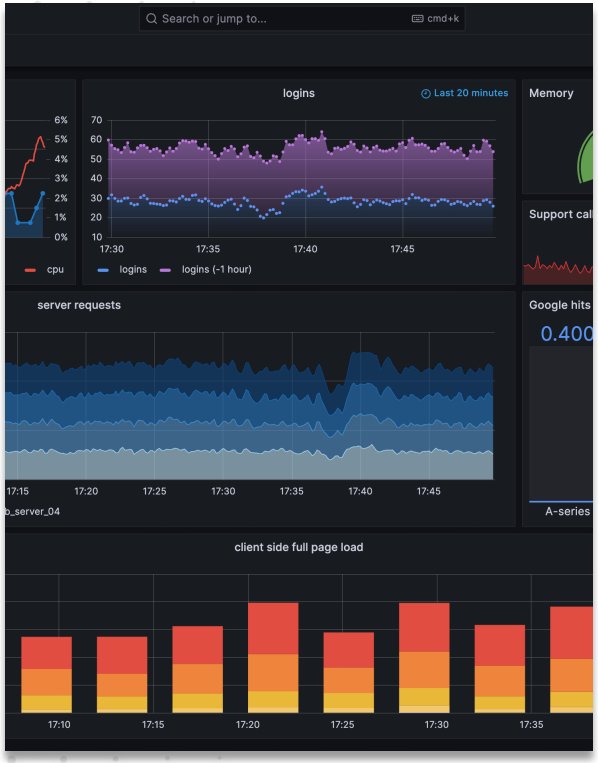
Pod Troubleshooting

- Describe
- Log
- Top
- Events
- debug

Lab

- Troubleshooting Pods

Section 4: Network Observability

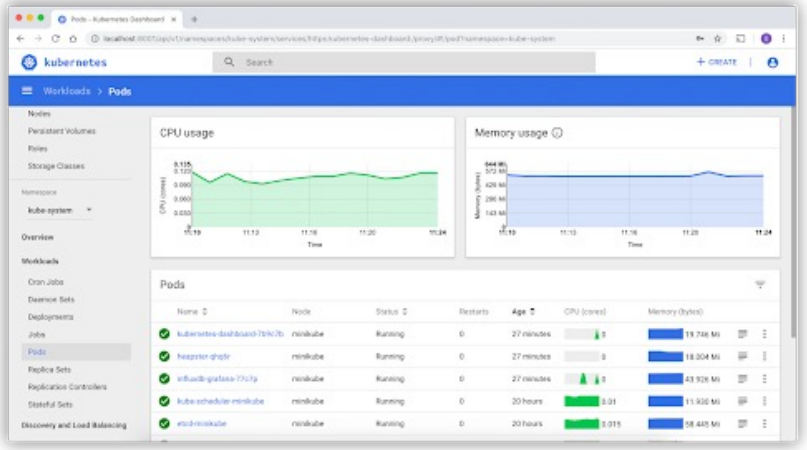


- Why monitoring and observability
- Network observability at the host and Pod level

Pearson

Why Monitoring

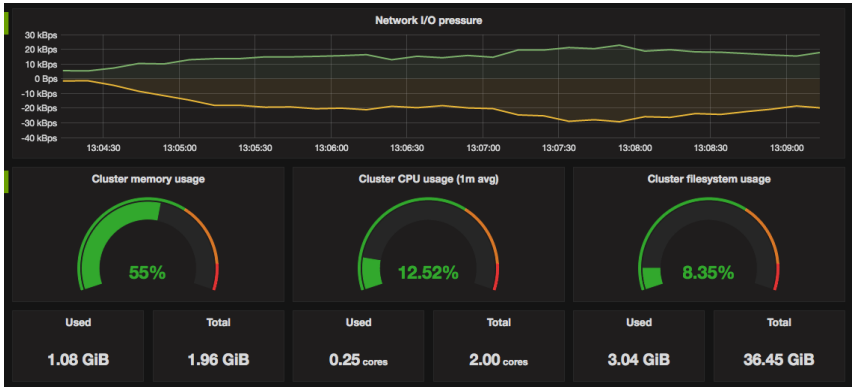
- See what's happening in real-time and take action



Why Observability

- Automate the action taking

Monitoring and Observability At The Host Level



- See the overall throughput for the host
- Watch for network blips and any downtime
- High response/MS
- Resource utilization
- VM health

Network Issues

- Ingress and Egress
- Local VM Networking
- Bandwidth

Resource Utilization and Scaling

Pearson

- Proper amount of CPU and Memory available.
- Scalable Worker Nodes

VM Health

- IF your Worker Nodes aren’t Serverless

Monitoring And Observability At The Pod Level

- Check for Pod networking health
- Ensure No CNI Issues
- Resource Optimization
- Metrics
- Events and Logs

Pod Networking

- CNI Health

Resource Optimization

- Proper amount of memory and CPU available for Pods
- Vertical and Horizontal Pod Autoscaling

Events and Logs

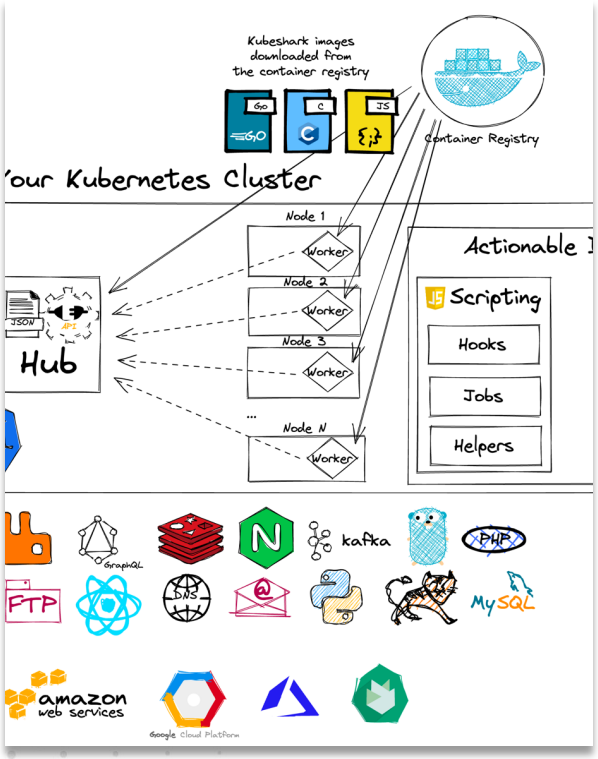
Pearson

- What we already talked about in Section 3:
  - `kubectl logs`
  - `kubectl events`
  - `kubectl top`
  - `kubectl describe`
  - `kubectl debug`

Lab

- Configuring Grafana
- Setting up resource optimization for Pods

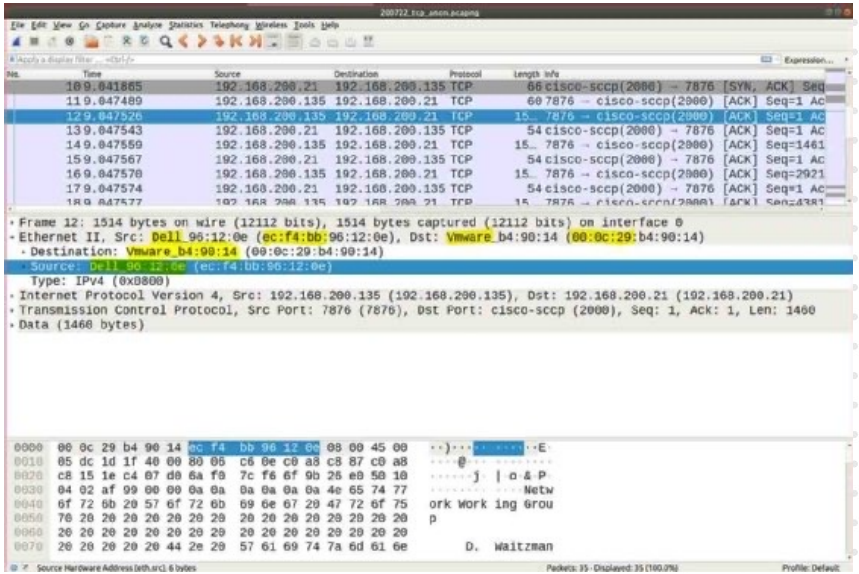
Section 5: Observability With Kubeshark



- What and why Kubeshark
- Monitoring and tracing API calls
- Filtering

What Is Kubeshark

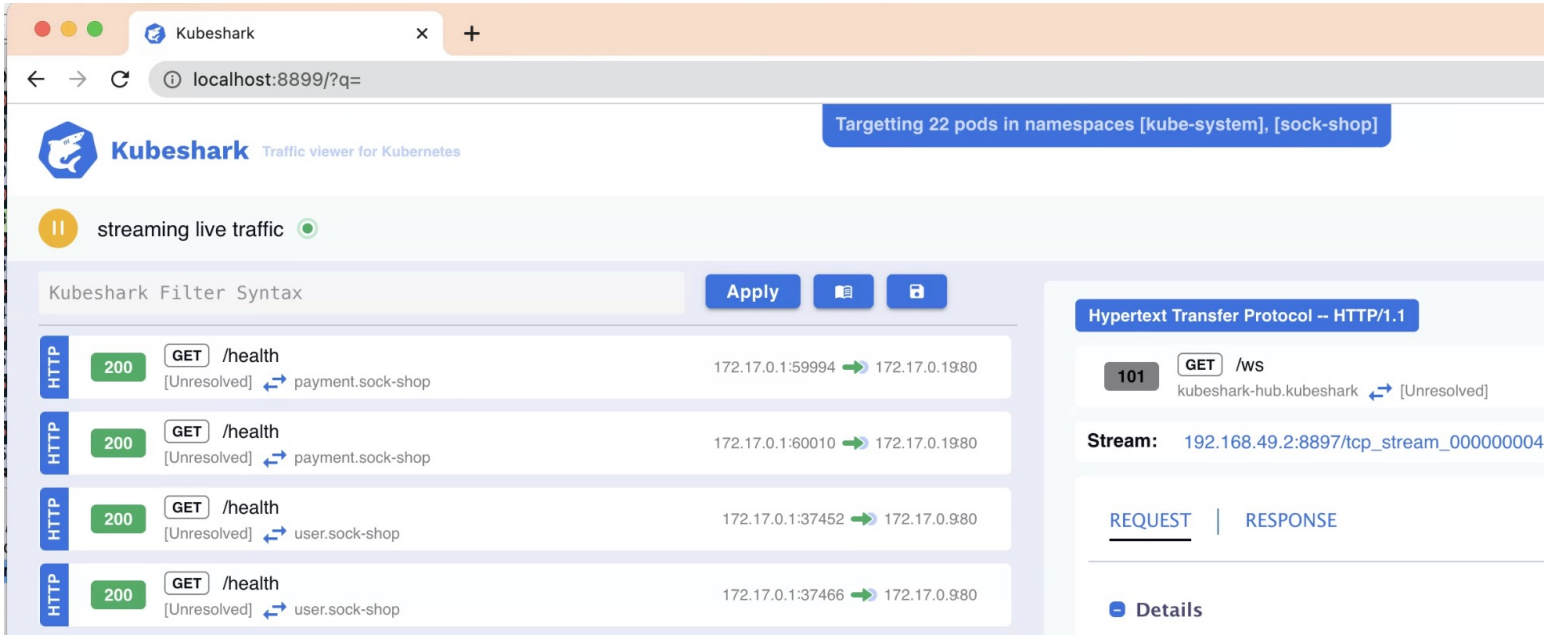
- Like Wireshark, but for Kubernetes



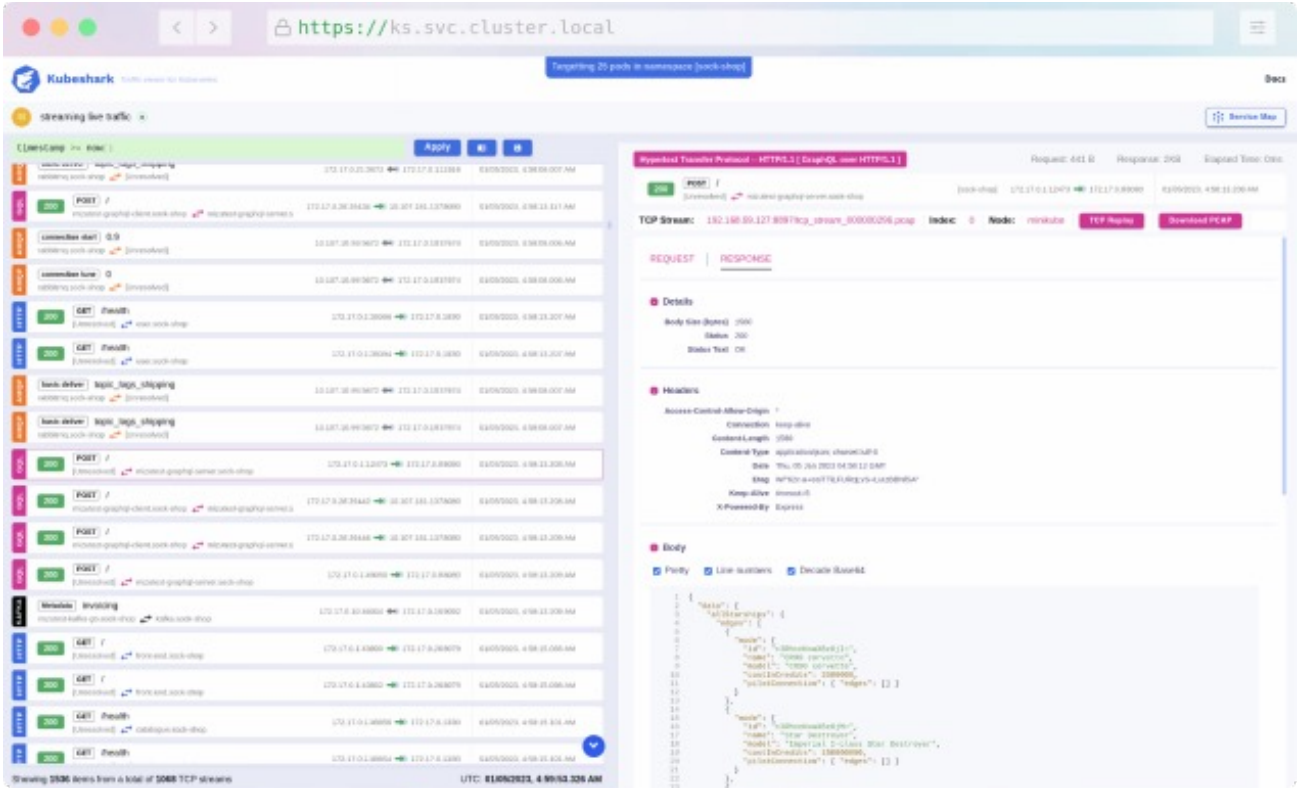
Why Use It?

- Tracing API calls
- Monitoring API calls
- Filtering network traffic

Installing Kubeshark



Monitoring API Calls



Pearson

Tracing API Calls

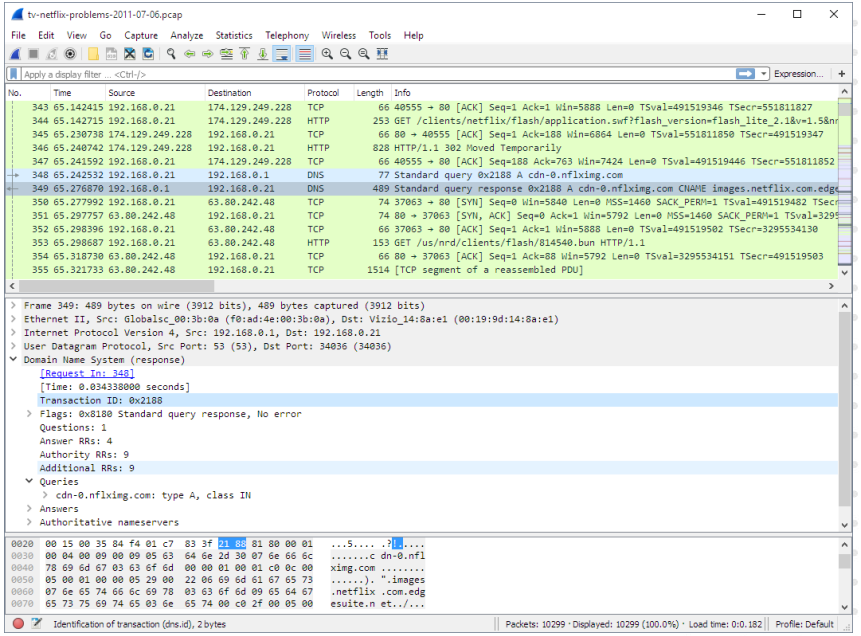
- Having the ability to see who and what is interacting with Kubernetes API endpoints

Filtering

- Seeing in real-time and past-time down to the second what and who is interacting with your environment

Wireshark

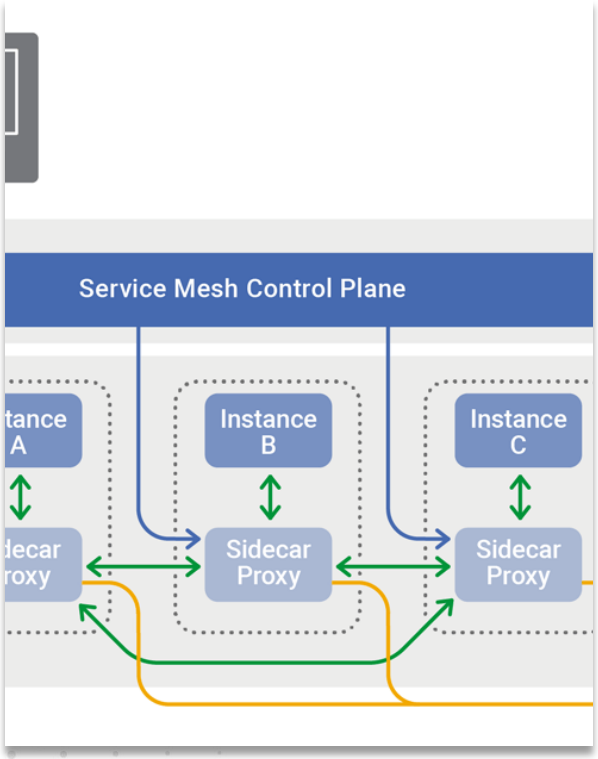
- Monitor all network traffic



Lab

- Install and configure Kubeshark
- Deploy a workload and view API calls

Section 6: Service Mesh



- What's a service mesh
- Service mesh options

What Is A Service Mesh

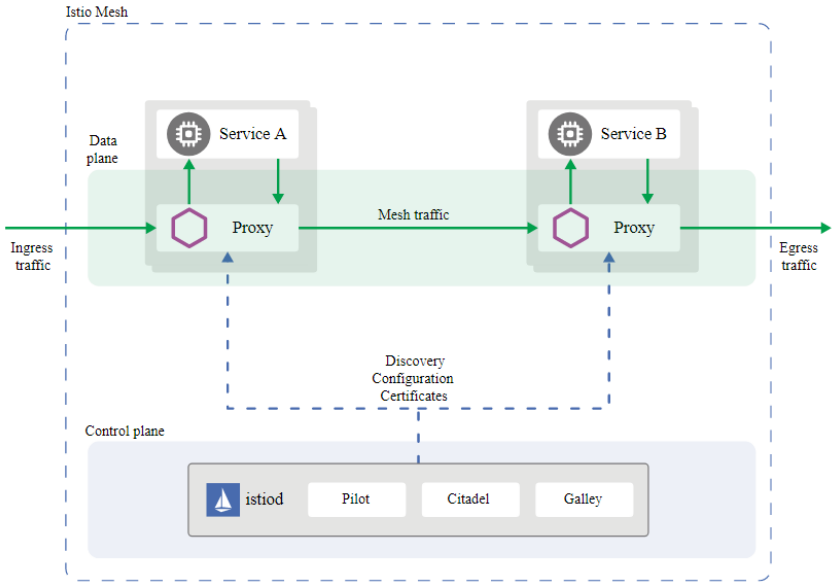
- End-to-end service encryption
- Network traffic management

Service Mesh Options

- Istio
- Linkerd
- Consul Connect

Istio

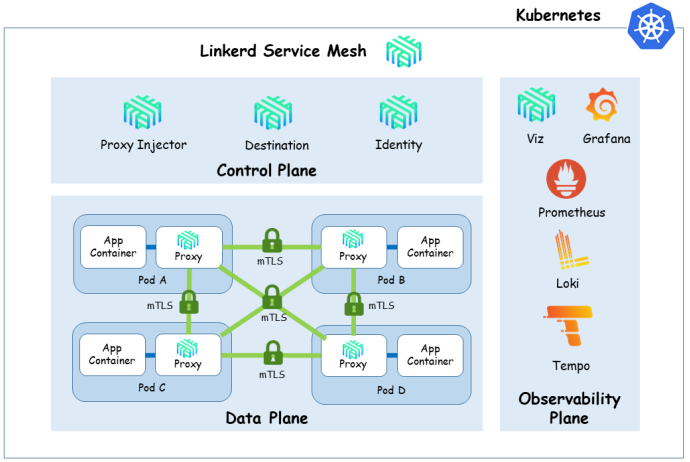
- Most popular Service Mesh



Istio Architecture

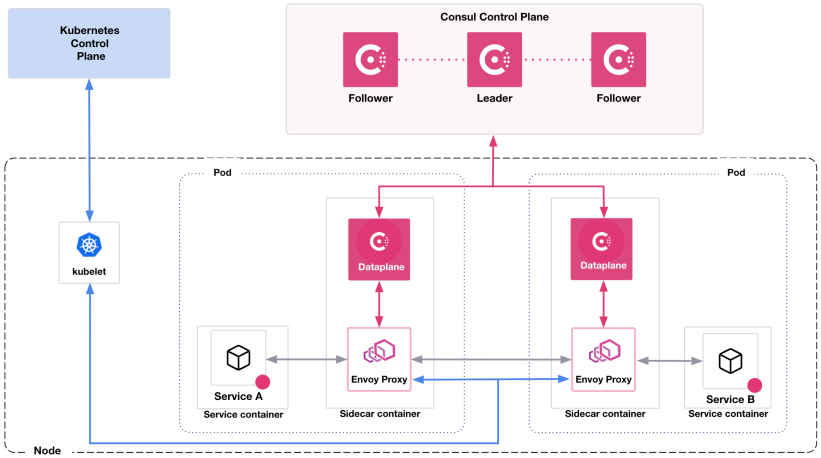
Linkerd

- Easy to use service mesh (service mesh out of the box)



Consul Connect

- HashiCorps service mesh



Pearson

Service Mesh Latency Management

- Manage and capture network latency

Service Mesh Encryption

- Utilization of mTLS

Network Policies

- Provide security for ingress/egress
- Block internal and external traffic

Lab

- Configuring Istio for network latency
- Network Policies