FORTINET | USA

XPERTS

SUMMIT 2024

Helping you create a digitally secure future.

K8S 201 WORKSHOP

CFOS

Srija Allam – Cloud Security Architect

Andy Wang – Consulting Systems Engineer

XPERTS SUMMIT 2024

Goals

K8s 201 Workshop

- Learn K8s Security Concepts
- Build a K8s Cluster
- Hands on Container FortiOS (cFOS) lab
- Ingress inspection with cFOS
- East west and Egress inspection with cFOS.

Disclaimer

Fortinet Confidential

This document contains confidential material proprietary to Fortinet, Inc.

This document and information and ideas herein may not be disclosed, copied, reproduced or distributed to anyone outside Fortinet, Inc. without prior written consent of Fortinet, Inc.

This information is pre-release and forward looking and therefore is subject to change without notice.

The purpose of this document is to provide a statement of the current direction of Fortinet's product strategy and product marketing efforts.

Please note that this Product Roadmap is neither intended to bind Fortinet to any particular course of product marketing and development nor to constitute a part of the license agreement or any contractual agreement with Fortinet or its subsidiaries or affiliates.

Agenda

1. Getting K8s Ready

2. K8s Security

3,4,5,6:K8s concepts and basics

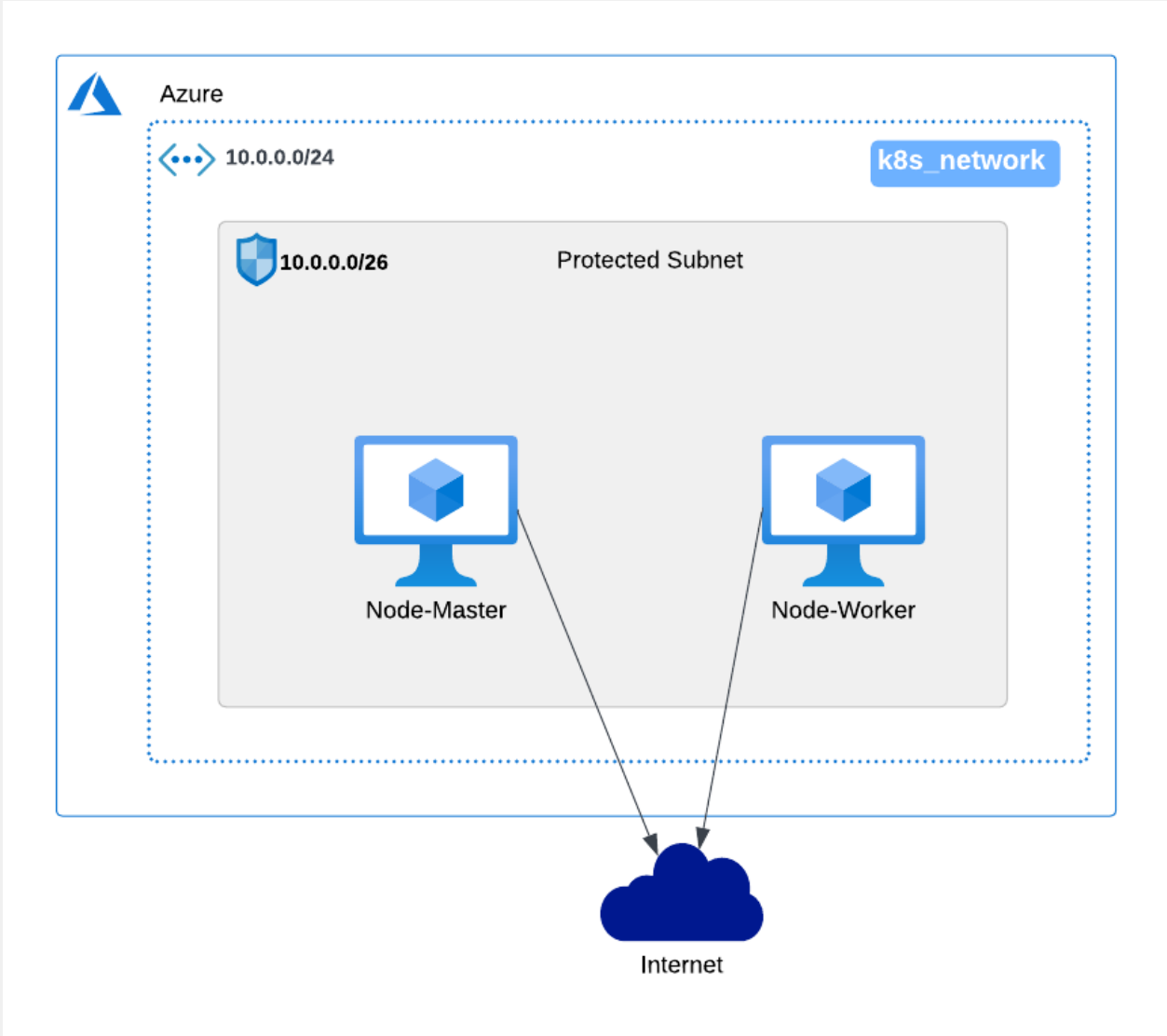
7. Ingress

8. Multus and 9. Egress Traffic

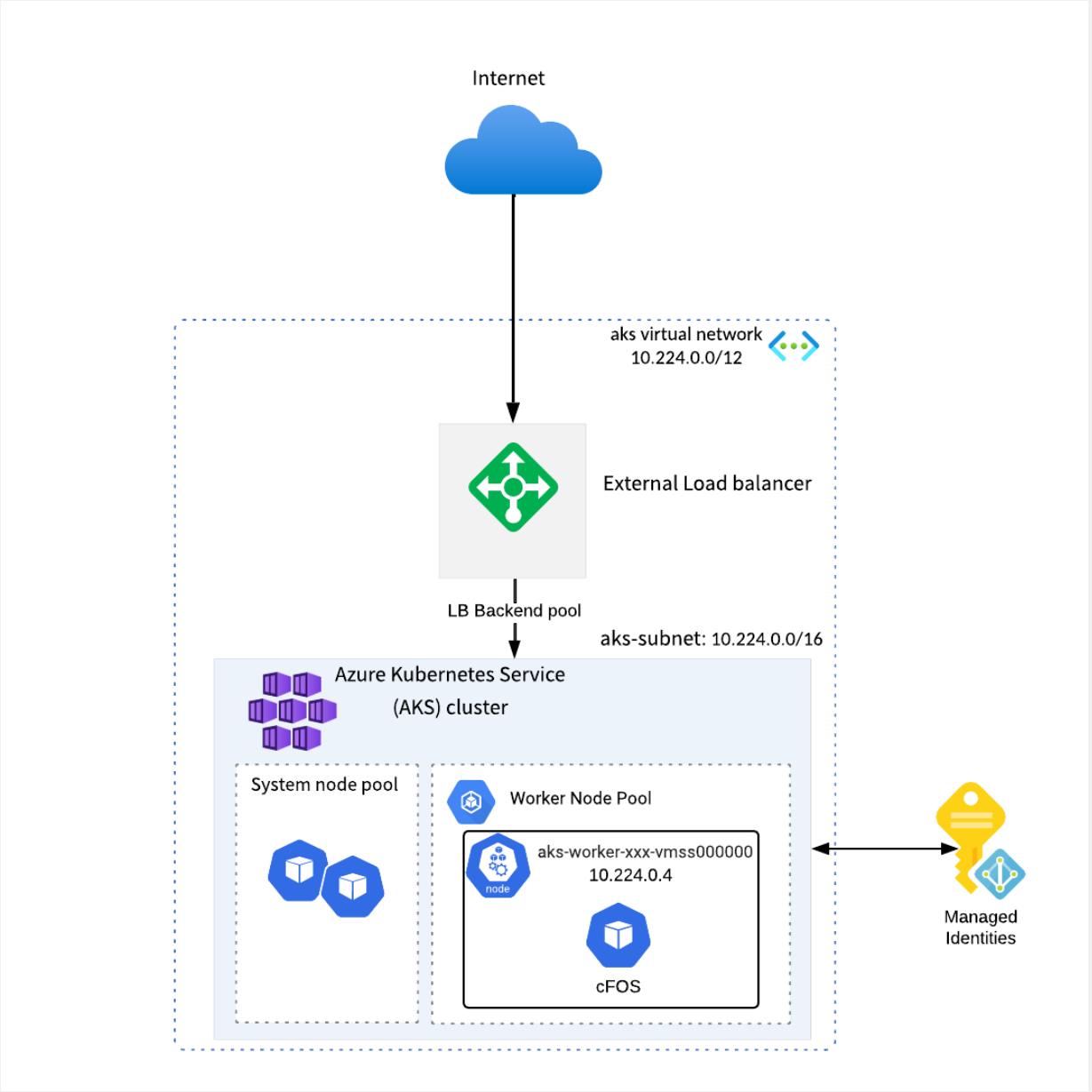
XPERTS SUMMIT 2024

Lab Architecture

Option1 and Option 2 – Self Managed K8s



Option 3 - AKS



XPERTS SUMMIT 2024

K8s Security

K8s Security- Authentication vs Authorization

Authentication Examples:

- **Username and Password**: When you log into your email account, you enter a username (or email address) and a password. The system checks whether the credentials match what is stored in the database to verify your identity.

- **Biometric Scanning**: Using a fingerprint, facial recognition, or iris scan to unlock your smartphone or access a secure area.

Authorization Examples:

- **Role-Based Access Control (RBAC):** In a company's internal system, an employee's role (e.g., HR, IT, Finance) determines what they can access. For example, an HR employee might have access to employee records, but not to financial systems.

- **File Permissions:** On your computer, you might have a folder where you are the only one who can read and write files, while other users might only be able to read the files but not edit them.

RBAC in K8s

Role-Based Access Control (RBAC) in Kubernetes is a method of regulating access to resources within a Kubernetes cluster based on the roles assigned to individual users or groups.

RBAC allows you to define what actions a user or a group of users can perform within the cluster, ensuring that they have the appropriate level of access to perform their tasks without compromising security.

Key Concepts of RBAC:

- **Roles:** A Role in Kubernetes contains a set of rules that define the allowed operations (such as get, list, create, delete) on certain resources (like Pods, ConfigMaps, etc.) within a specific namespace.
- **RoleBinding:** A RoleBinding associates a Role with a user, group, or service account within a specific namespace. It defines who can perform what actions within that namespace.
    - **ClusterRoleBinding:** Like RoleBinding, but it links a ClusterRole to users, groups, or service accounts at the cluster level, granting permissions across all namespaces.
- **Subjects:** Subjects refer to the entities (users, groups, or service accounts) that the roles are assigned to via RoleBinding or ClusterRoleBinding.

ConfigMaps and Secrets

ConfigMap and **Secret** are both Kubernetes resources used to manage configuration data and sensitive information in a Kubernetes cluster. They allow you to decouple configuration artifacts from image content to keep containerized applications portable.

ConfigMap	Secret
store non-sensitive configuration data in key-value pairs	specifically designed to store sensitive data, such as passwords, OAuth tokens, SSH keys
• Storing environment variables. • configuration files in a more structured way. • application-specific configuration available to pods	• Storing sensitive information like database passwords, API keys, or TLS certificates. • Providing credentials to applications without embedding them in the container images.

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: example-configmap
data:
  app.properties: |
    APP_ENV=production
    DB_HOST=database.example.com
```

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: example-secret
type: Opaque
data:
  username: YWRtaW4=  # base64 encoded "admin"
  password: MWYyZDFlMmU2N2Rm  # base64 encoded "1f2d1e2e67df"
```

Storage

- **emptyDir:** A temporary directory that is created when a pod is assigned to a node and exists as long as the pod runs on that node.
- **hostPath:** Maps a directory or file on the host node’s filesystem to a pod.
- **persistentVolumeClaim (PVC):** Represents a user's request for storage, which is dynamically provisioned based on the request.
- **network-attached storage (NAS):** Examples include NFS, GlusterFS, CephFS, etc.
- **Cloud provider volumes:** Cloud-specific storage solutions like Amazon EBS, Google Persistent Disk, Azure Disk, etc.

Summary:

Volumes provide a way to persist data at the pod level but are tied to the pod's lifecycle.

PersistentVolumes (PVs) and **PersistentVolumeClaims (PVCs)** provide a more durable storage solution, abstracting the underlying infrastructure and allowing dynamic or static provisioning of storage.

K8s Networking

CNI (Container Network Interface) is a crucial component in Kubernetes, responsible for configuring the network interfaces in containers and enabling communication between them.

How CNI Works in Kubernetes:

•When a pod is created, the kubelet (the node agent in Kubernetes) calls the CNI plugin to set up the network for the pod.

•The CNI plugin assigns an IP address to the pod's network interface and configures the necessary routes for communication.

•The plugin may also set up network policies, firewall rules, or other networking-related settings based on the Kubernetes configuration.

XPERTS SUMMIT 2024

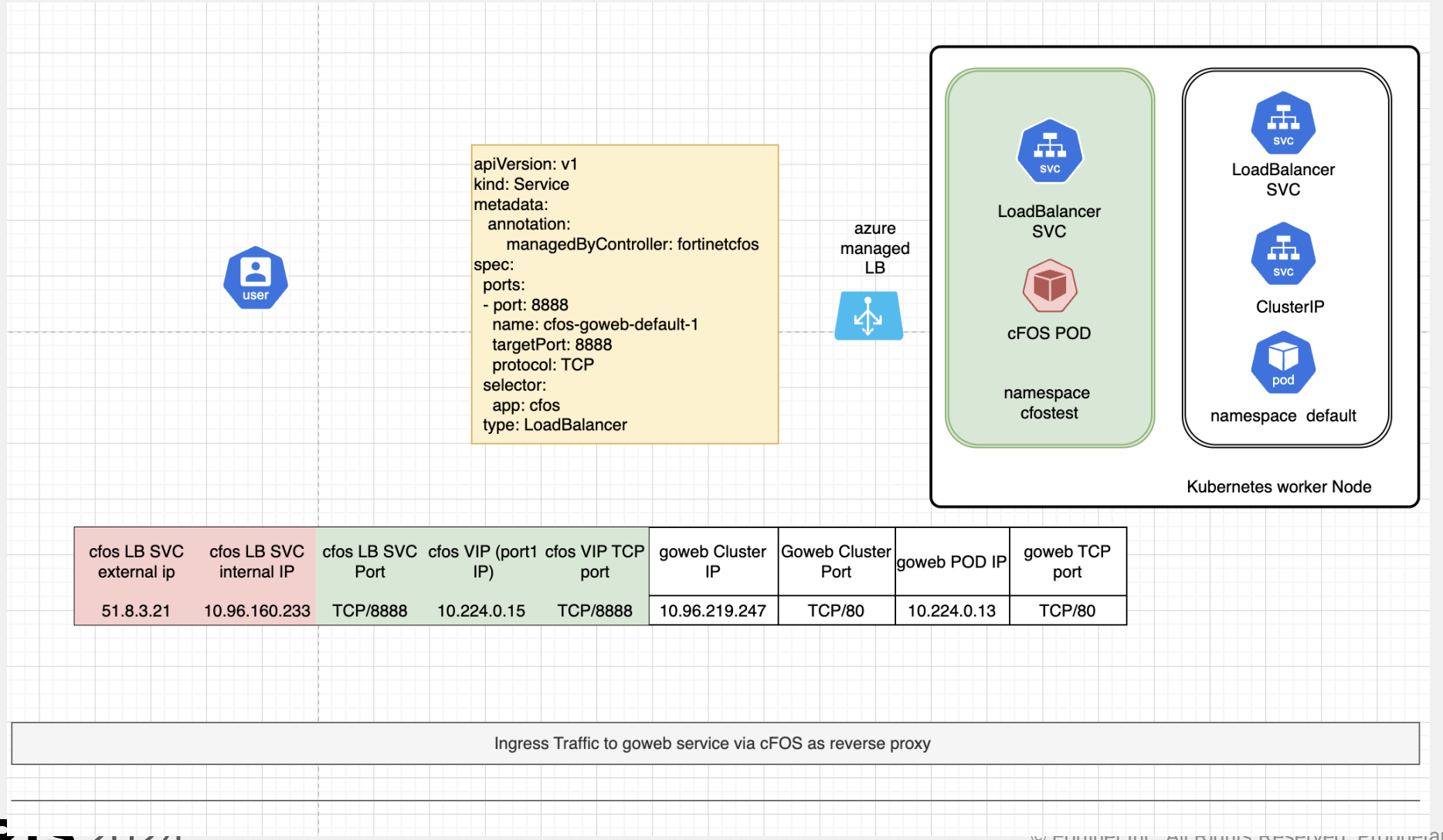
cFOS

cFOS

Container FortiOS, the operating system that powers Fortinet's security appliance as a container, can be integrated with Kubernetes to enhance the security of inbound traffic to your containers.

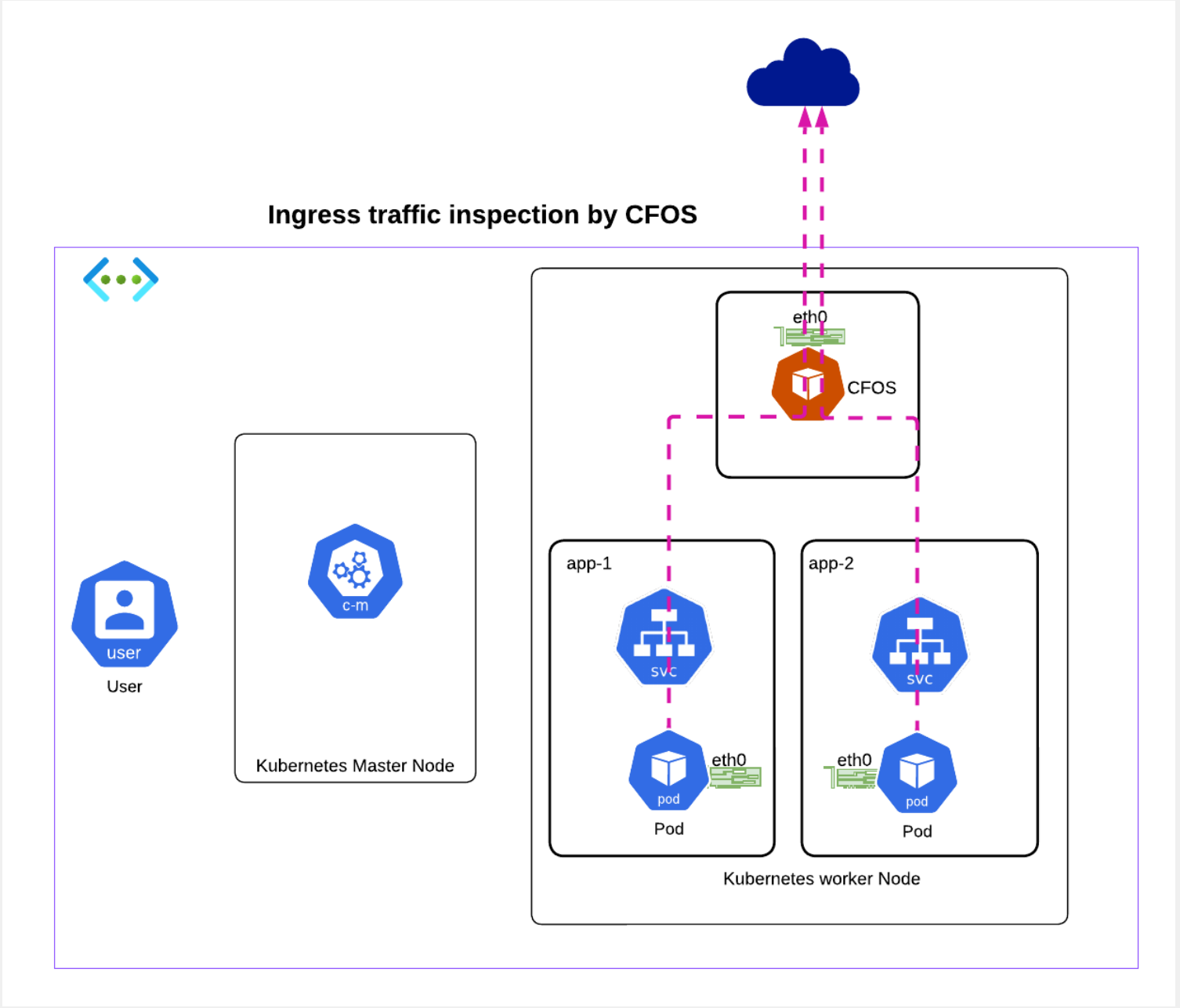
Before cFOS

After cFOS



cfos LB SVC external ip	cfos LB SVC internal IP	cfos LB SVC Port	cfos VIP (port1 IP)	cfos VIP TCP port	goweb Cluster IP	Goweb Cluster Port	goweb POD IP	goweb TCP port
51.8.3.21	10.96.160.233	TCP/8888	10.224.0.15	TCP/8888	10.96.219.247	TCP/80	10.224.0.13	TCP/80

Ingress traffic flow



XPERTS SUMMIT 2024

Multus

Multus

Multus is a CNI (Container Network Interface) plugin for Kubernetes that enables attaching multiple network interfaces to a single pod.

Key Features of Multus:

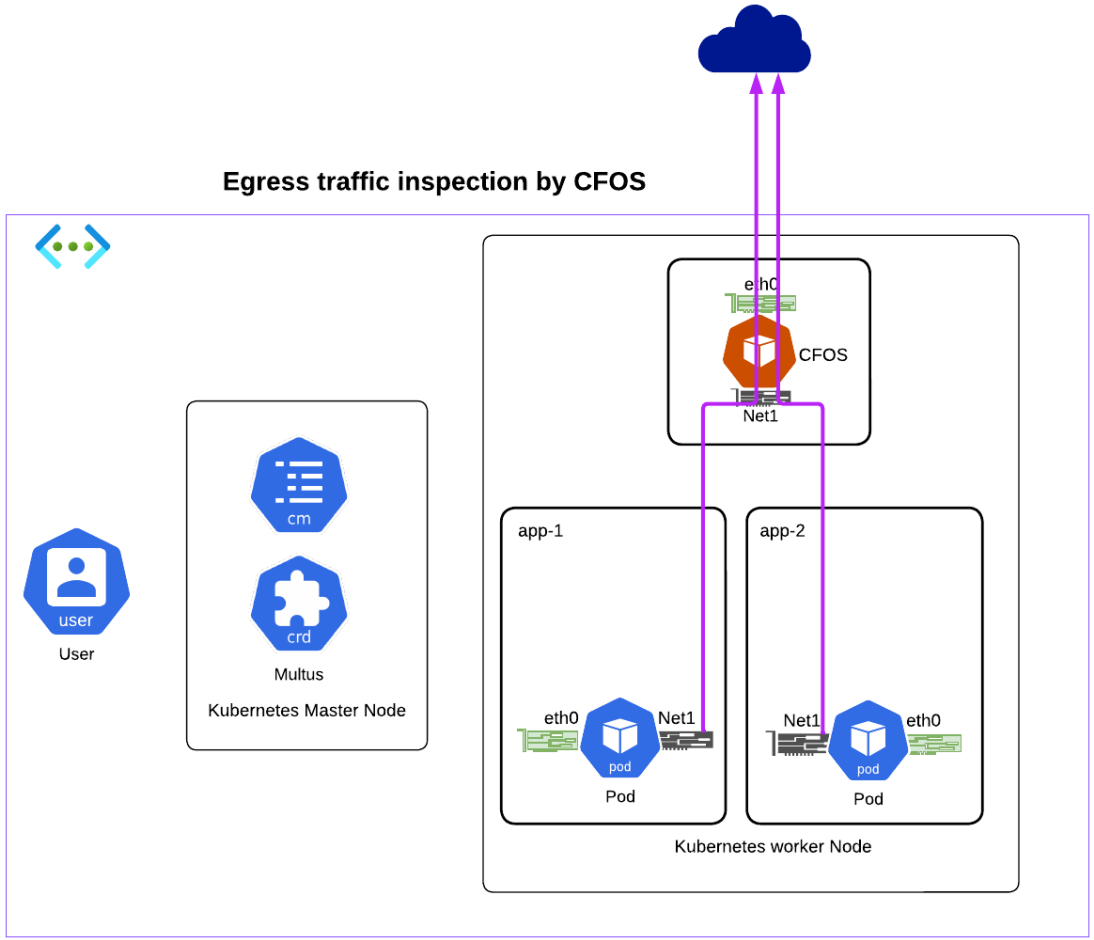
1.Multiple Network Interfaces: Multus allows pods to have more than one network interface, each connected to different networks. This is useful in scenarios where pods need to interact with different network environments simultaneously.

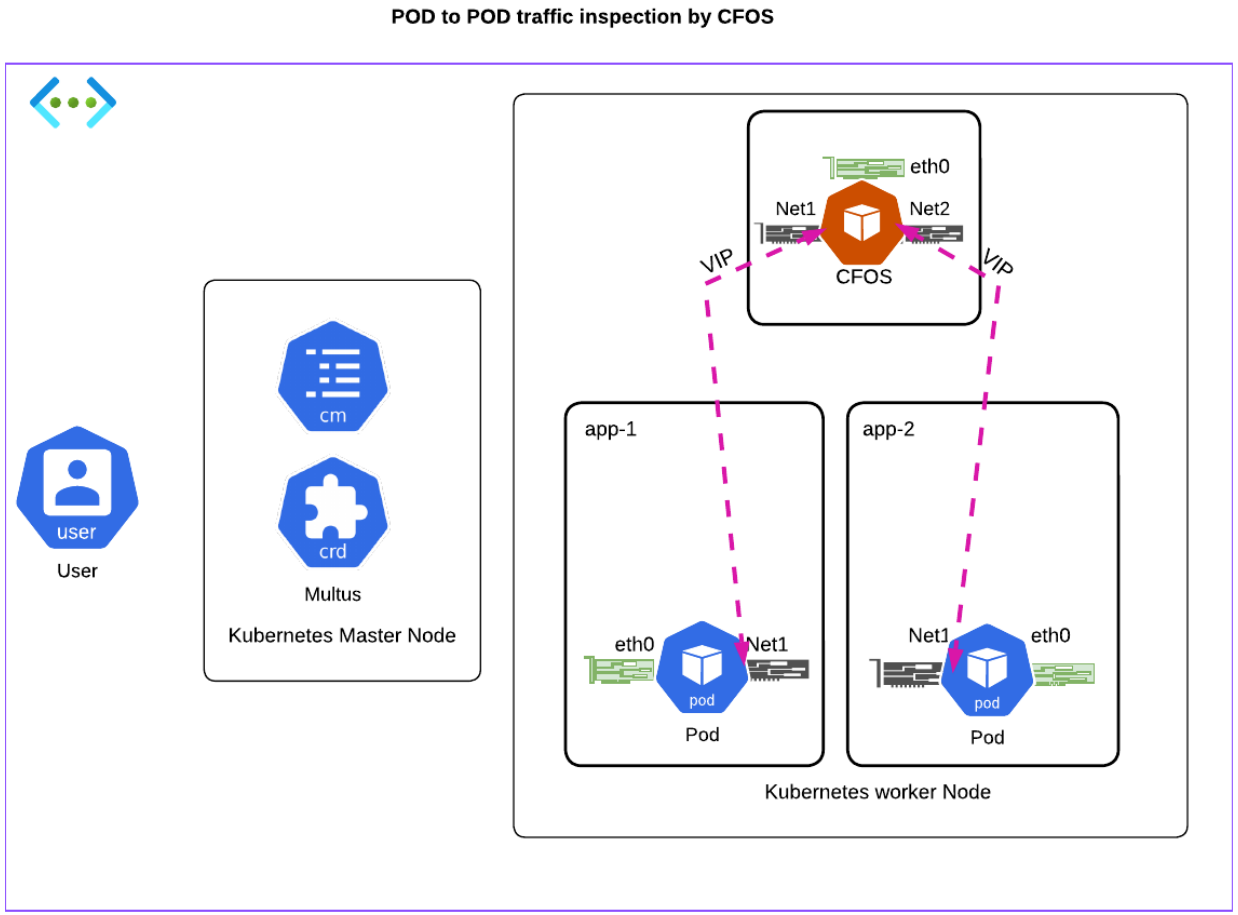
2.Primary and Secondary Networks: Multus works by delegating the primary network to a default CNI plugin (like Flannel or Calico), while enabling additional interfaces (secondary networks) to be configured using other CNI plugins or configurations.

3.Custom Network Annotations: Multus uses Kubernetes pod annotations to specify which additional networks to attach to the pod. This means that network configurations can be customized on a per-pod basis, giving fine-grained control over pod networking.

4. NetworkAttachmentDefinition CRD: Multus introduces the NetworkAttachmentDefinition custom resource definition (CRD), which is used to define the configuration of additional networks that can be attached to pods.

Egress and E-W traffic flow





Chapter1 Task3: Access Token

Username:

fortinetwandy

Token:

Uc2NB8ze8j3NUKCFLZNV0PRg4pQarKOB1M2EsIviKs+ACRCNVWDH