Pod Security Standards (PSS) & Network Policies

# 1. Introduction

Kubernetes provides robust security features to ensure isolation and controlled communication between workloads. Two critical components of Kubernetes security are:

- Pod Security Standards (PSS): Defines baseline security requirements for pod configurations.

- Network Policies: Controls traffic flow at the IP address or port level between pods.

# 2. Pod Security Standards (PSS)

2.1 What is PSS?  
Pod Security Standards are a set of policies defined by Kubernetes to control security settings in pod specifications. They were introduced as a part of the Kubernetes PodSecurity Admission controller in Kubernetes v1.23+.

2.2 PSS Levels  
There are three predefined policy levels:  
- Privileged: No restrictions; allows all Kubernetes features. Useful for trusted workloads.  
- Baseline: Minimally restrictive; suitable for common workloads while reducing known risks.  
- Restricted: Highly restrictive; enforces current best practices for hardening pods.

2.3 Enabling PSS  
To enable PSS in a Kubernetes cluster, configure the PodSecurity admission plugin via namespace labels.

Example:

kubectl label namespace <namespace> \

pod-security.kubernetes.io/enforce=baseline \

pod-security.kubernetes.io/enforce-version=latest

# 3. Network Policies

3.1 What are Network Policies?  
Network Policies are Kubernetes resources used to control the communication between pods and services. They define rules for ingress (incoming) and egress (outgoing) traffic.

3.2 Key Concepts  
- Pod Selector: Selects pods the policy applies to.  
- Ingress Rules: Control incoming traffic to pods.  
- Egress Rules: Control outgoing traffic from pods.  
- Namespaces: Policies can span multiple namespaces if specified.

3.3 Example: Deny All Traffic

apiVersion: networking.k8s.io/v1  
kind: NetworkPolicy  
metadata:  
 name: deny-all  
 namespace: default  
spec:  
 podSelector: {}  
 policyTypes:  
 - Ingress  
 - Egress

# 3.4 Example: Allow Ingress from a Specific Pod

apiVersion: networking.k8s.io/v1  
kind: NetworkPolicy  
metadata:  
 name: allow-frontend-to-backend  
 namespace: default  
spec:  
 podSelector:  
 matchLabels:  
 app: backend  
 ingress:  
 - from:  
 - podSelector:  
 matchLabels:  
 app: frontend

# 3.5 Example: Allow Egress to Specific CIDR

apiVersion: networking.k8s.io/v1  
kind: NetworkPolicy  
metadata:  
 name: allow-egress-to-external  
 namespace: default  
spec:  
 podSelector:  
 matchLabels:  
 app: myapp  
 policyTypes:  
 - Egress  
 egress:  
 - to:  
 - ipBlock:  
 cidr: 8.8.8.8/32

# 4. Best Practices

4.1 Pod Security Standards  
- Use restricted mode in production namespaces.  
- Apply baseline for development or less sensitive workloads.  
- Always test policies using audit and warn before enforcing.

4.2 Network Policies  
- Apply a default deny all policy and explicitly allow required traffic.  
- Use namespace and pod selectors to scope policies accurately.  
- Keep rules granular and specific to reduce attack surfaces.

# 5. Tools & Observability

- kubewarden / kyverno / opa-gatekeeper: For advanced policy enforcement.

- NetworkPolicy Editor & Visualizer: https://editor.cilium.io/

- kubectl plugin (kubectl-np-viewer): Helps view effective network policies.

# 6. Summary Table

Feature | PSS | Network Policy  
--------|-----|----------------  
Purpose | Pod configuration hardening | Control traffic between pods  
Applied via | Namespace labels | YAML manifests  
Levels | Privileged, Baseline, Restricted | Not applicable  
Requires plugin | No | Yes (CNI plugin)  
Enforcement modes | Enforce, Audit, Warn | No built-in modes

# 7. Conclusion

Combining Pod Security Standards and Network Policies provides a layered security model in Kubernetes:

- PSS ensures pods don’t use insecure or excessive permissions.

- Network Policies ensure pods only communicate as needed.

Together, they form a foundational part of a Zero Trust security model in Kubernetes environments.