AKS Security Best Practices

Alessandro Vozza
Principal Software Engineer @Microsoft, CNCF Ambassador,
Cloud Pirate Captain

Content

Azure Kubernetes Service 101

Security best practices

Recent CVEs

OSS security

Q&A // Discussion

The Pirates

Who are the Cloud Pirates

Meet the pirates


The Pirate Cove
cloudpirates.nl/signup
Opening mid-October
@cloudpiratesnl

Introduction

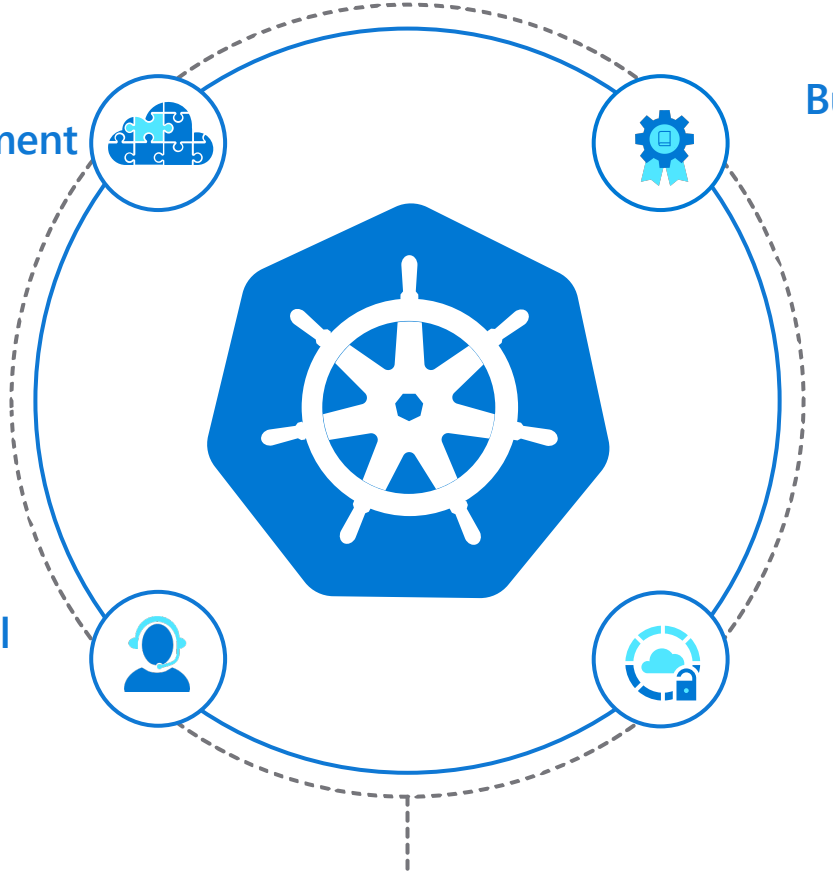
Kubernetes on Azure investment areas

Unified management

Consistent configuration and governance across environments

Built-in best practices

Proactive and actionable recommendations, based on knowledge from thousands of enterprise engagements



Operational efficiency

Support from certified Kubernetes experts or an interactive portal experience for faster diagnostics and troubleshooting

Multi-layer security

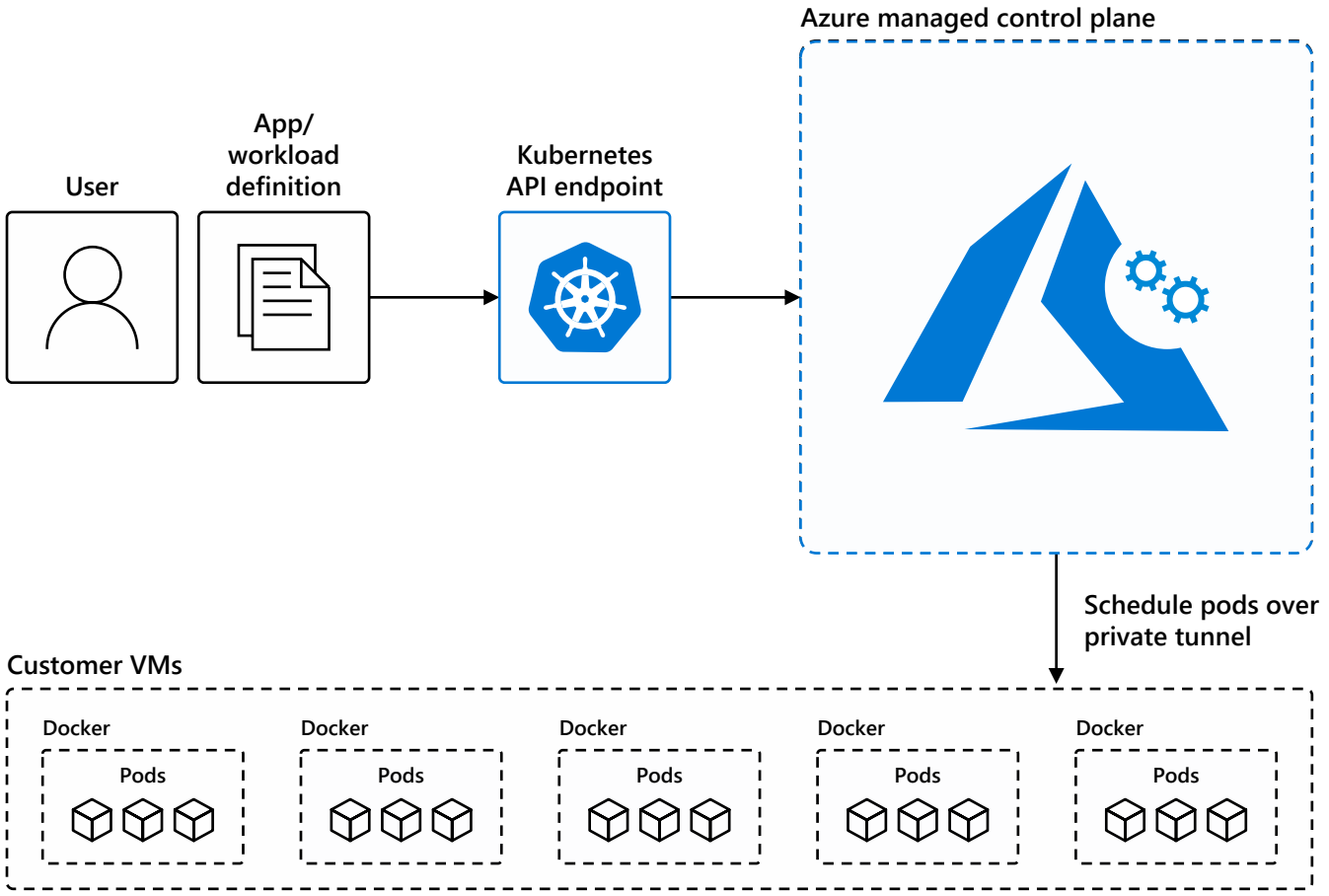
Hardened security and layers of isolation across compute resources, data, and networking

Community-led innovation

Kubernetes innovations built together with the community

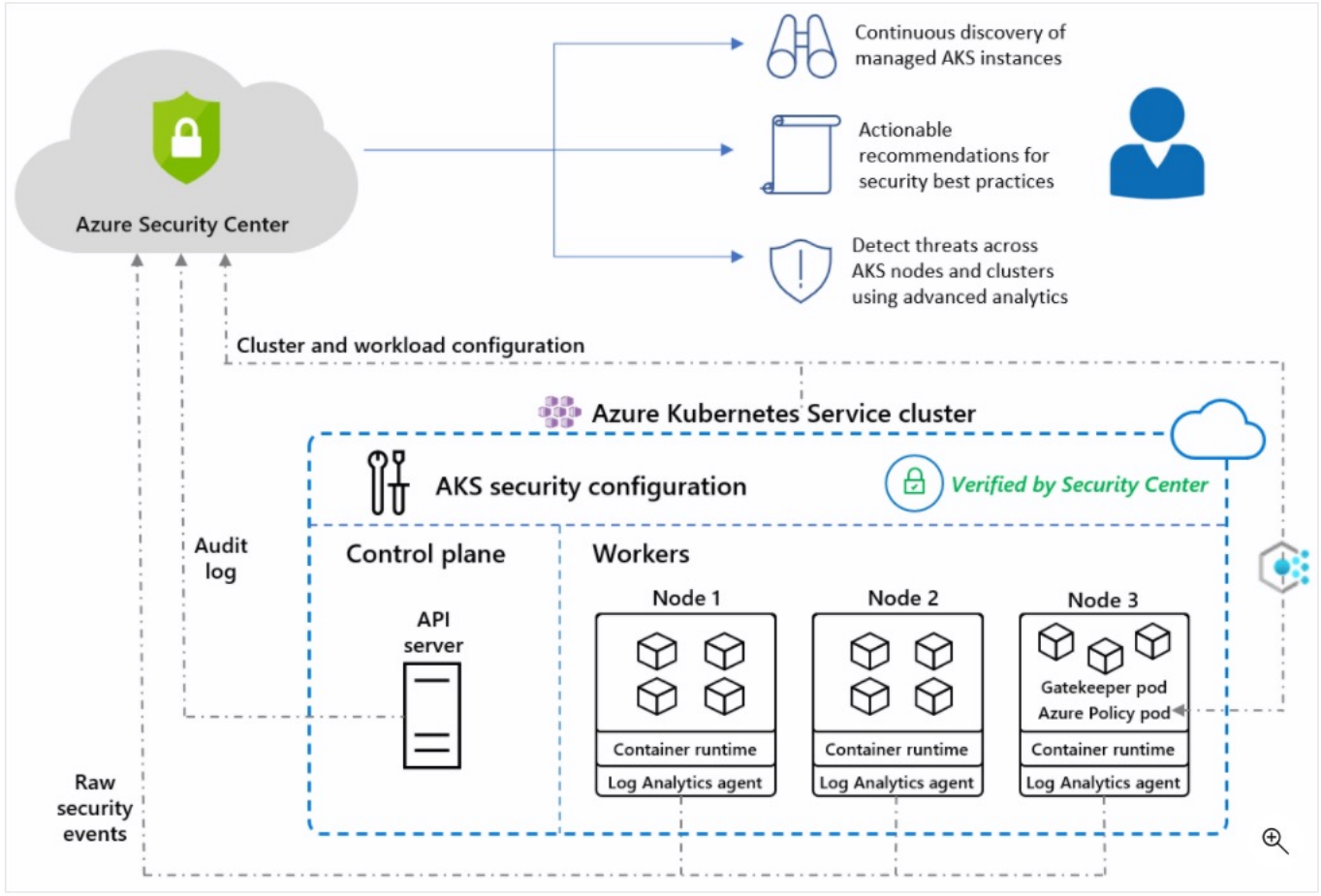
AKS 101

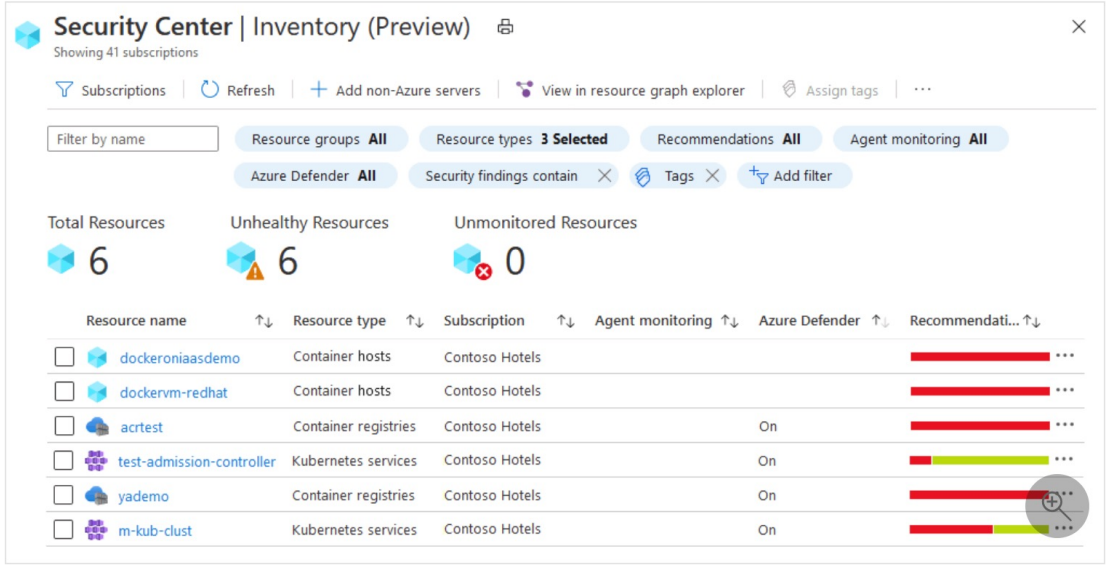
- Automated upgrades, patches
- High reliability, availability
- Easy, secure cluster scaling
- Self-healing
- API server monitoring



Azure Security

Azure Security Center





Azure Security Benchmark

Azure security benchmark is used here to organize these AKS security features.

- Network Security
- Identity Management
- Data Protection
- Logging & Monitoring
- Vulnerability

ref: https://docs.microsoft.com/en-us/azure/security/benchmarks/overview

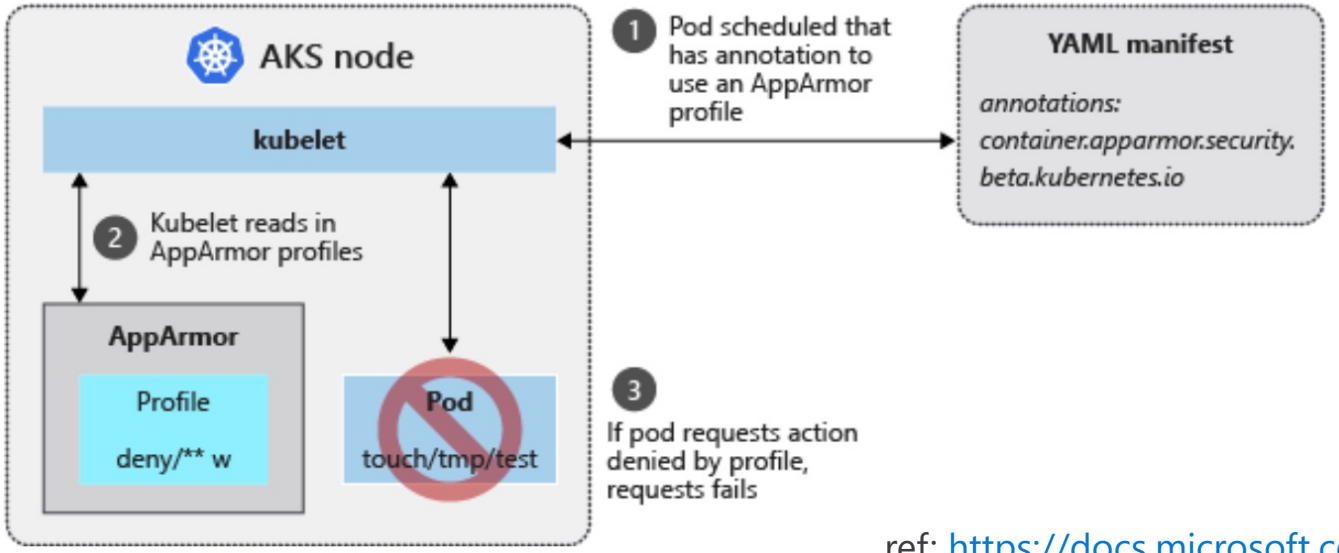
Application Security

AppArmor in AKS

```
#include <tunables/global>
profile k8s-apparmor-example-deny-write flags=(attach_disconnected) {
  #include <abstractions/base>
  file,
  # Deny all file writes.
  deny /** w,
}
```

- Linux Kernel Security Module (like SELinux)
- *Enforcement* or *Complain* mode

```
annotations:
  container.apparmor.security.beta.kubernetes.io/hello: localhost/k8s-apparmor-example-deny-write
```



ref: https://docs.microsoft.com/en-us/azure/security/benchmarks/overview

Seccomp in AKS

SECCOMP
Linux Privilege Escalation

- Linux Kernel Security Module (like AppArmor)
- Works at the process level

```
annotations:
  seccomp.security.alpha.kubernetes.io/pod: localhost/prevent-chmod
```

```
{
  "defaultAction": "SCMP_ACT_ALLOW",
  "syscalls": [
    {
      "name": "chmod",
      "action": "SCMP_ACT_ERRNO"
    },
    {
      "name": "fchmodat",
      "action": "SCMP_ACT_ERRNO"
    },
    {
      "name": "chmodat",
      "action": "SCMP_ACT_ERRNO"
    }
  ]
}
```

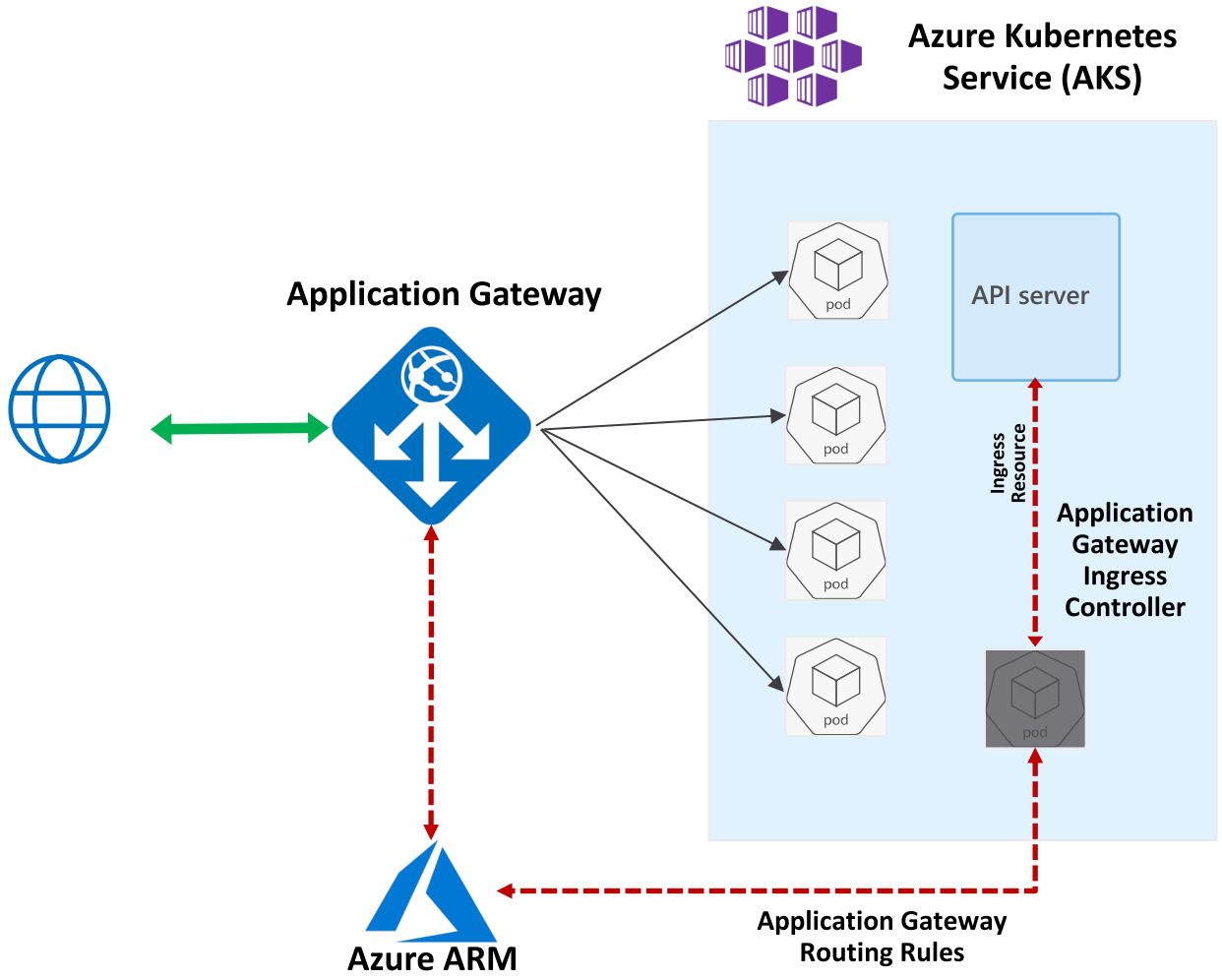
Network security

Preview

Application Gateway Ingress Controller

- Attach Application Gateways to AKS Clusters
- Load Balance from the Internet to pods (CNI and kubenet supported)
- Supports features of k8s ingress resource – TLS, multi-site and path-based routing
- Pod-AAD for ARM authentication

https://github.com/Azure/application-gateway-kubernetes-ingress



North/South Traffic in AKS with Azure Firewall

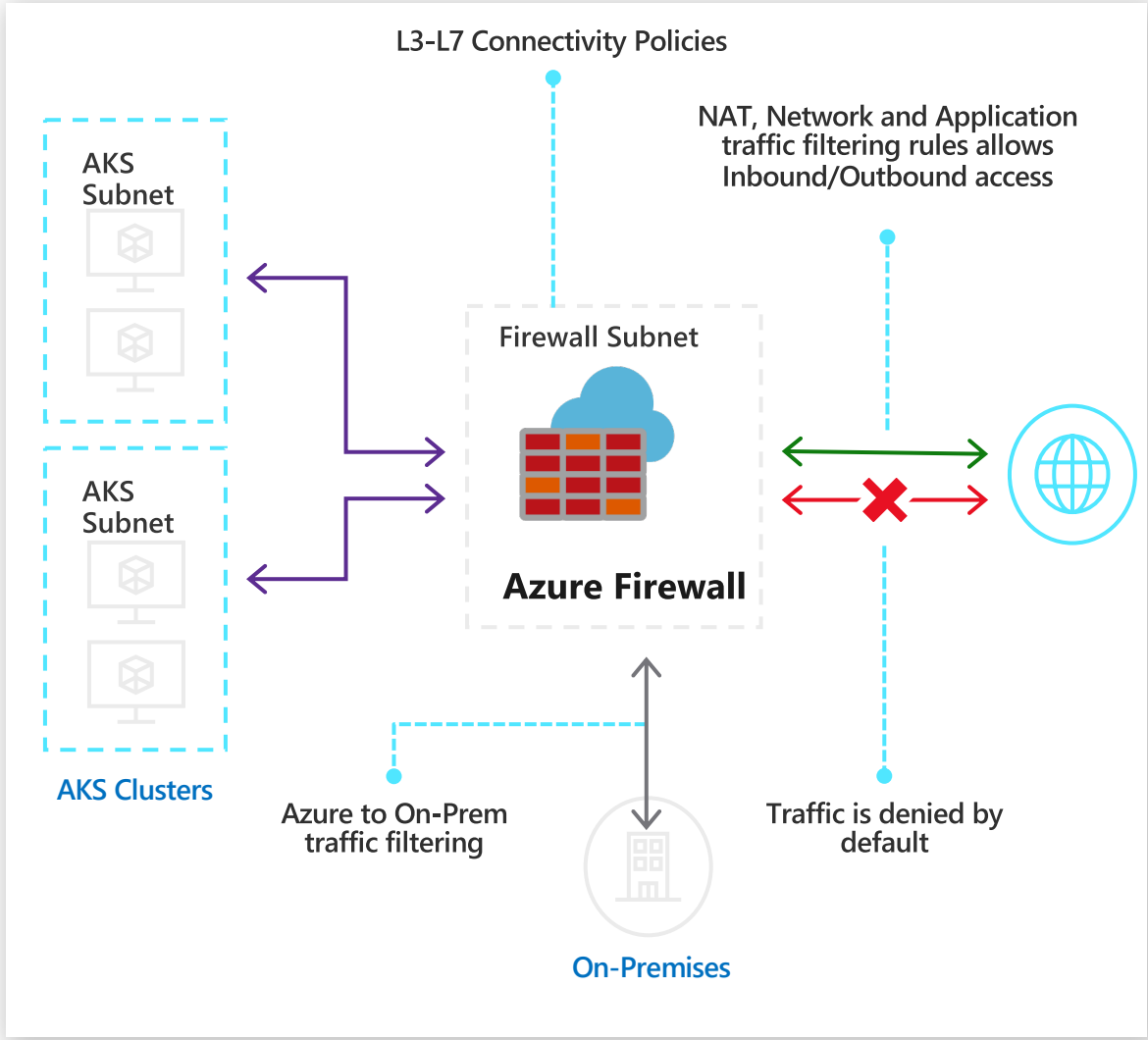
Cloud native stateful Firewall as a service

A first among public cloud providers

Target FQDN

*.azmk8s.io – http, https (Eg. *eastus.azmk8s.io)
k8s.gcr.io – http, https
storage.googleapis.com - http, https
*auth.docker.io – http, https
*cloudflare.docker.io – http, https
*registry-1.docker.io – http, https

- NAT Rules
  - TCP Port 22 to AKS Destination Addresses for the Region

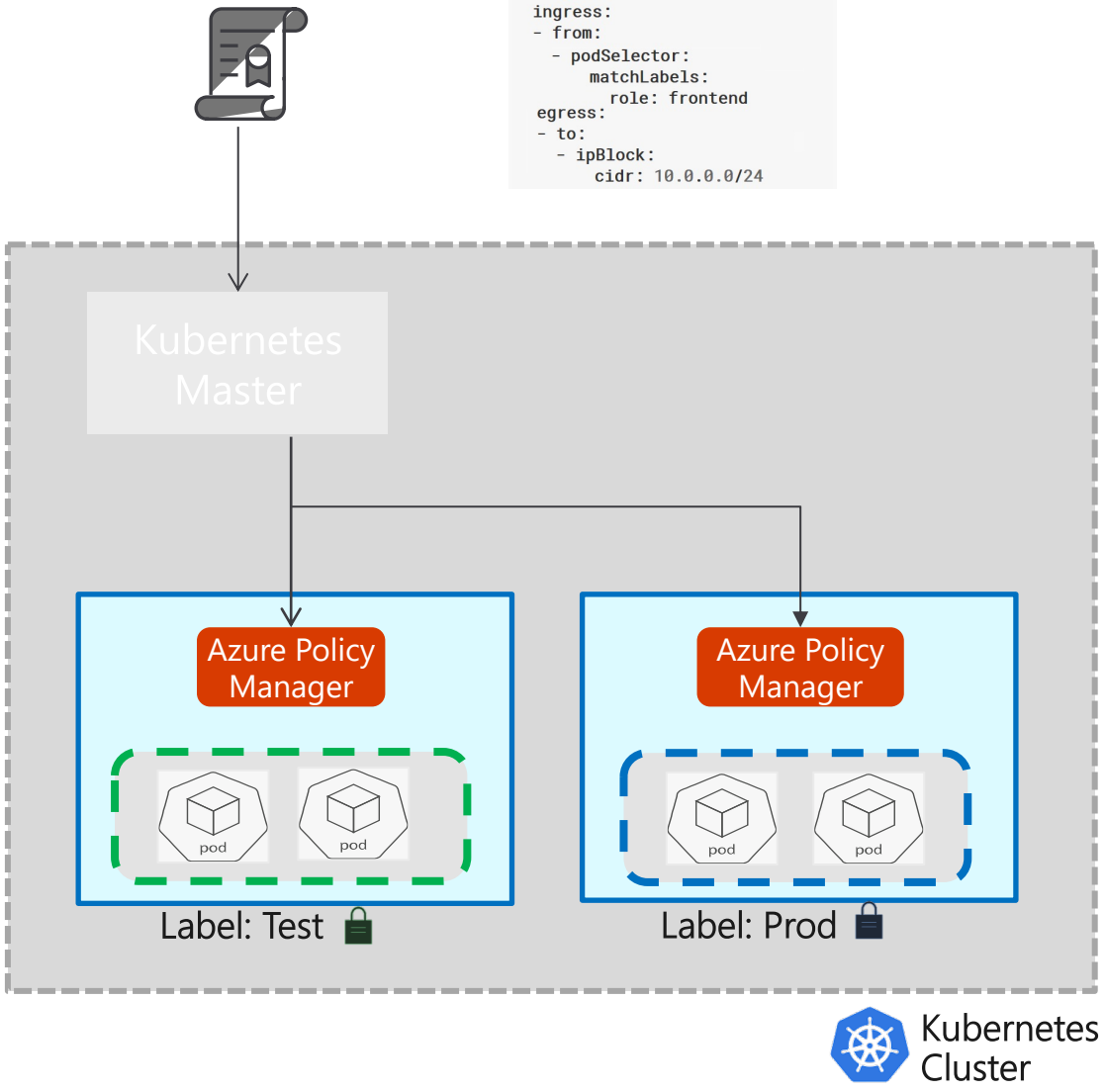


Network Policies

Provides micro-segmentation for containers
Label-based selection of Pods
Policy resource yaml file specifies Ingress and Egress policies
Works in conjunction with Azure CNI

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: test-network-policy
spec:
  podSelector:
    matchLabels:
      role: db
  policyTypes:
  - Ingress
  - Egress
  ingress:
  - from:
    - podSelector:
        matchLabels:
          role: frontend
  egress:
  - to:
    - ipBlock:
        cidr: 10.0.0.0/24
```

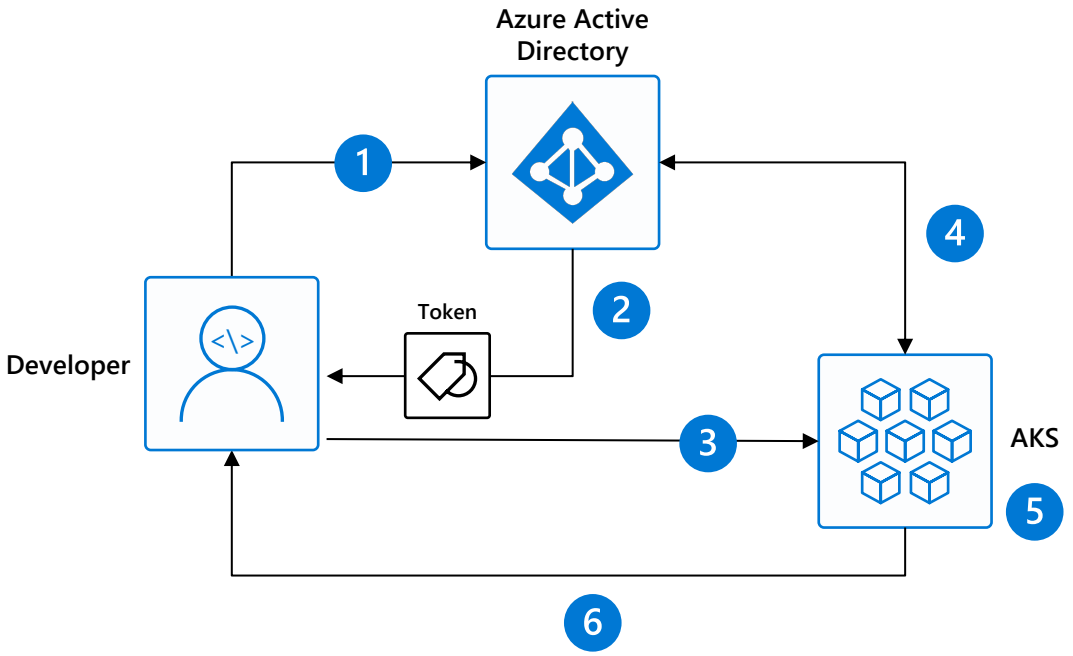
```
kubectl apply -f
policy.yaml
```



Cluster access

Cluster Level – Kubernetes RBAC and Azure AD Authentication

1. Kubernetes Developer authenticates with AAD
2. The AAD token issuance endpoint issues the access token
3. Developer performs action w/ AAD token.
   Eg. *kubectl create pod*
4. Kubernetes validates token with AAD and fetches the Developer's AAD Groups
   Eg. Dev Team A, App Group B
5. Kubernetes RBAC and cluster policies are applied
6. Request is successful or not based on the previous validation



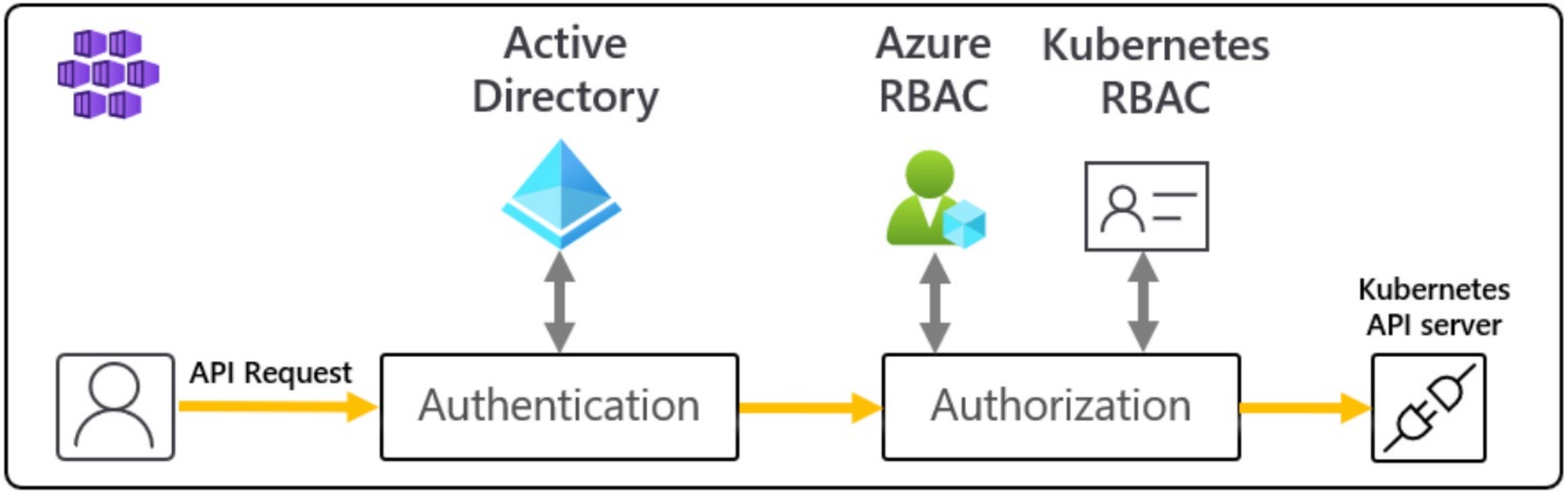
AAD with Azure RBAC Authorization

Usage: az aks create -g <rg> -n <name> --enable-aad --enable-azure-rbac

Roles:

- Azure Kubernetes Service RBAC Viewer,
- Azure Kubernetes Service RBAC Writer,
- Azure Kubernetes Service RBAC Admin,
- Azure Kubernetes Service RBAC Cluster Admin

```
az role assignment create --role "Azure Kubernetes Service RBAC Viewer" --assignee <aad-id> --scope
<AKS_ID/namespaces/namespace-name>
```



AAD with Azure RBAC Authorization – custom roles

```
{
    "Name": "AKS Deployment Reader and DS writer",
    "Description": "Lets you view all deployments in cluster/namespace.",
    "Actions": [],
    "NotActions": [],
    "DataActions": [
        "Microsoft.ContainerService/managedClusters/apps/deployments/read”,
        "Microsoft.ContainerService/managedClusters/apps/statefulsets/write”,
    ],
    "NotDataActions": [],
    "assignableScopes": [
        "/subscriptions/<YOUR SUBSCRIPTION ID>"
    ]
}
```

https://docs.microsoft.com/en-us/azure/role-based-access-control/resource-provider-operations#microsoftcontainerservice

Upgrades

Node Vulnerability

Node update: Security updates are automatically applied to Linux nodes to protect customer's Azure Kubernetes Service (AKS) clusters. These updates include OS security fixes or kernel updates. AKS don't do reboot to complete the update.
Node image upgrade: AKS built VHD including OS and runtime update. (including windows)

Usage: az aks upgrade -g <rg> -n <name> --node-image-only

Cluster Level – Nodes, Upgrade and Patches

Regular maintenance, security and cleanup tasks

- Maintain, update and upgrade hosts and kubernetes - Monthly ideal, 3 months minimum
- Security patches
  - **AKS automatically applies security patches to the nodes on a nightly schedule**
  - **You're responsible to reboot as required, leveraging tools like Kured DaemonSet**
    https://github.com/weaveworks/kured

Upgrade to new versions

```
$ az aks upgrade --name myAKSCluster \
--resource-group myResourceGroup \
--kubernetes-version 1.10.6
```

- **SSH Access**
  - DenyEscalatingExec
- **Running benchmarks and tests to validate cluster setup**
  - Kube-bench
  - Aqua Hunter
  - Others

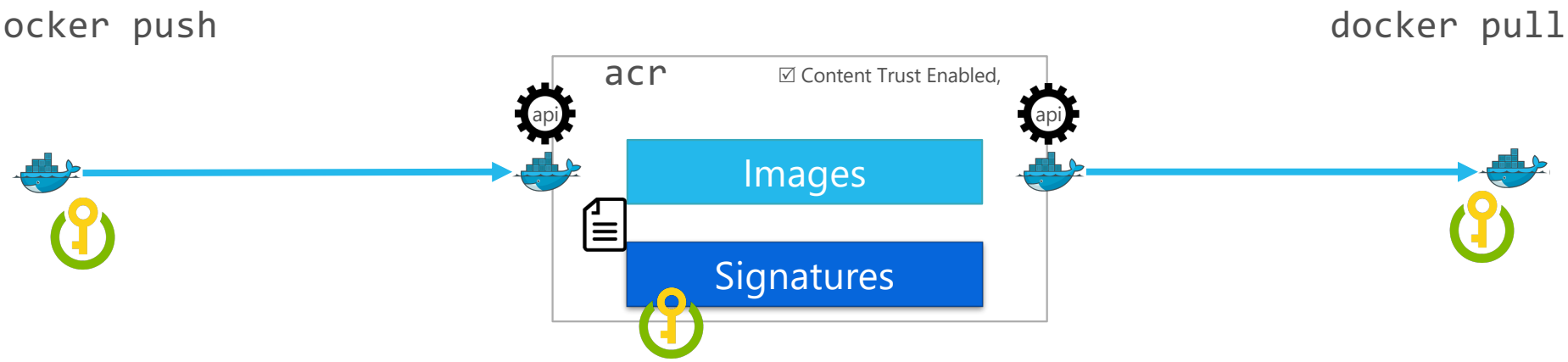
CVE-2021-25741


```
# az aks upgrade --resource-group rg-cluster --name cluster --node-image-only
```

AKS/issues#2547

Secure Supply Chain

ACR - Content Trust



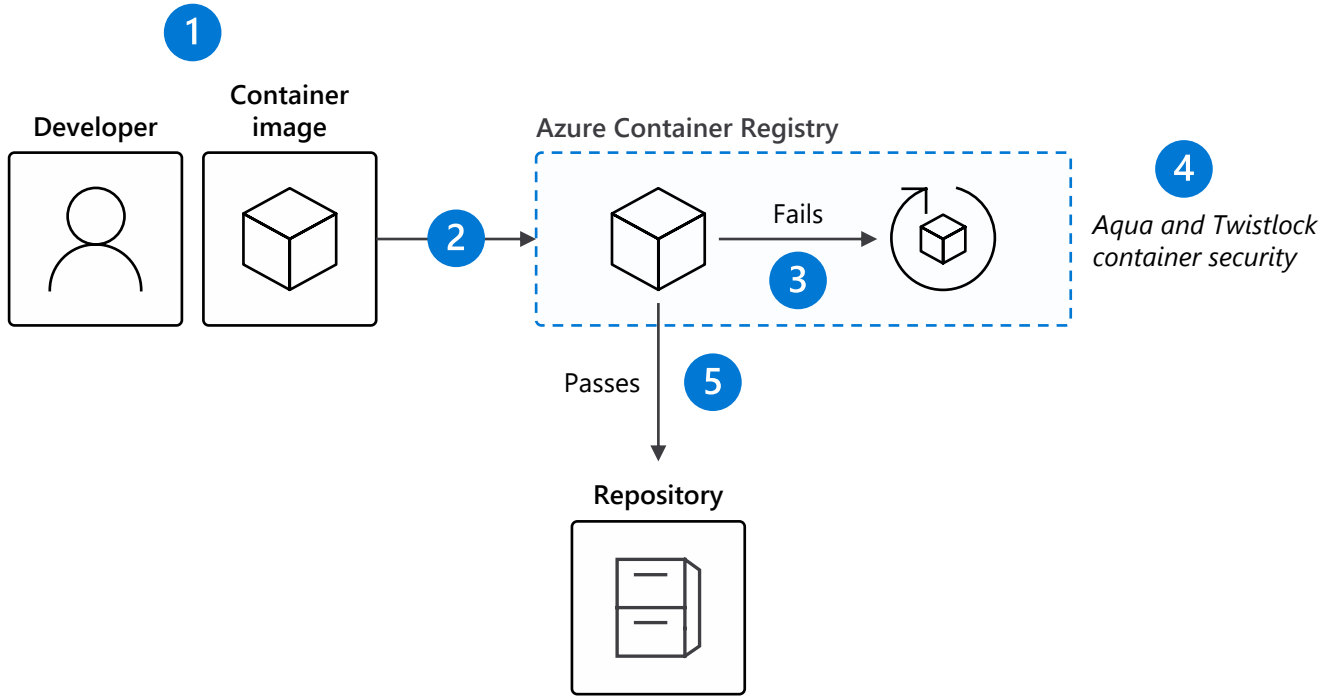
az role assignment create --scope <registry ID> --role AcrImageSigner --assignee <user name>

To enforce on AKS: https://github.com/sse-secure-systems/connaisseur

Note: notaryproject/notation is the evolution of CT (AKS/#688)

ACR – vulnerability scanning

1. Developer/CI system builds container image
2. Image pushed to Azure Container Registry
3. Azure Container Registry quarantines image until scanning passes
4. Azure Container Registry scans content leveraging Aqua, Twistlock
5. Azure Container Registry publishes the image to the repository



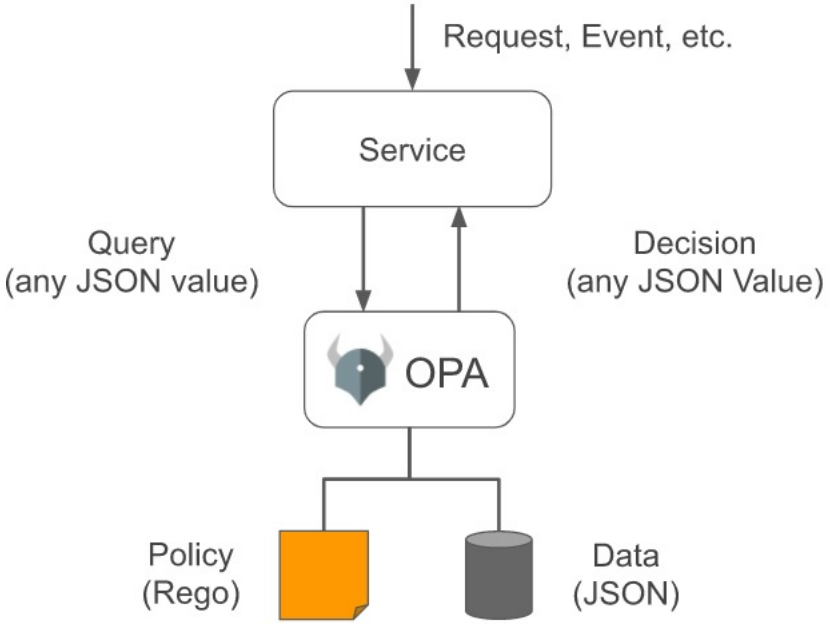
Azure Policy

Azure Policy (open policy agent/gatekeeper)

The Azure Policy Add-on for AKS installs a managed instance of Gatekeeper, a validating/mutating admission controller.
Azure Policy for Kubernetes is built on the open-source Open Policy Agent.

The add-on enacts the following functions:

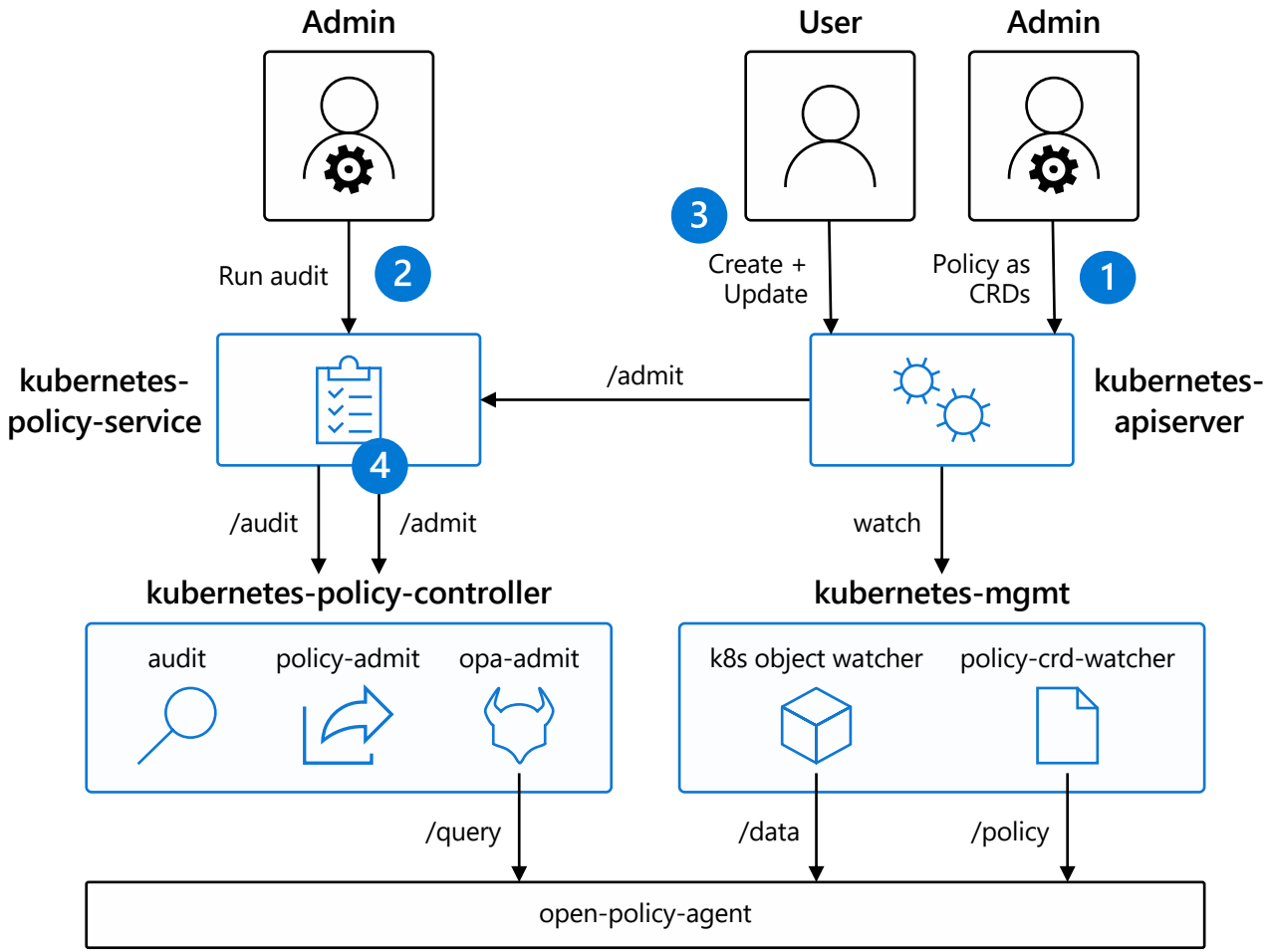
- Checks with Azure Policy service for policy assignments to the cluster.
- Deploys policy definitions into the cluster as constraint template and constraint custom resources.
- Reports auditing and compliance details back to Azure Policy service.



Note: PSP are deprecated! Translation: github.com/open-policy-agent/gatekeeper-library/tree/master/src/pod-security-policy

Policy controller

1. Admin adds policy for the cluster
2. Admin audits compliance of the cluster using /audit endpoint
3. User uses standard Kubernetes API to operate the cluster and the create actions are guarded by policy
4. Kubernetes-policy-controller provides an admission controller webhook that performs evaluations by calling open-policy-agent (OPA) service



Azure Policy

```yaml
apiVersion: templates.gatekeeper.sh/v1beta1
kind: ConstraintTemplate
metadata:
  name: k8sazurecontainernoprivilege
spec:
  crd:
    spec:
      names:
        kind: K8sAzureContainerNoPrivilege
        listKind: K8sAzureContainerNoPrivilegeList
        plural: k8sazurecontainernoprivilege
        singular: k8sazurecontainernoprivilege
  targets:
    - target: admission.k8s.gatekeeper.sh
      rego: |
        package k8sazurecontainernoprivilege

        violation[{"msg": msg, "details": {}}] {
            c := input_containers[_]
            c.securityContext.privileged
            msg := sprintf("Privileged container is not allowed: %v, securityContext: %v", [c.name, c.securityContext])
        }

        input_containers[c] {
            c := input.review.object.spec.containers[_]
        }

        input_containers[c] {
            c := input.review.object.spec.initContainers[_]
        }
```

```yaml
apiVersion: constraints.gatekeeper.sh/v1beta1
kind: K8sAzureContainerNoPrivilege
metadata:
  name: container-no-privilege
spec:
  match:
    excludedNamespaces: {{ .Values.excludedNamespaces }}
    kinds:
      - apiGroups: [""]
        kinds: ["Pod"]
```

Azure Policy

Now export your definitions and assignments to GitHub and manage them using actions! Click on 'Export definition' menu option. Learn more here

Name	Definition loc...	Policies	Type	Definition type	Category
Kubernetes cluster pod security restricted standards for Linux-based workloads		8	Built-in	Initiative	Kubernetes
Kubernetes cluster pod security baseline standards for Linux-based workloads		5	Built-in	Initiative	Kubernetes
[Preview]: Azure Policy Add-on for Kubernetes service (AKS) should be installed and enab...			Built-in	Policy	Kubernetes
[Preview]: Deploy GitOps to Kubernetes cluster			Built-in	Policy	Kubernetes
Both operating systems and data disks in Azure Kubernetes Service clusters should be en...			Built-in	Policy	Kubernetes
Deploy Azure Policy Add-on to Azure Kubernetes Service clusters			Built-in	Policy	Kubernetes
Kubernetes cluster pod hostPath volumes should only use allowed host paths			Built-in	Policy	Kubernetes
Kubernetes cluster pods should only use allowed volume types			Built-in	Policy	Kubernetes
Enforce HTTPS ingress in Kubernetes cluster			Built-in	Policy	Kubernetes
Kubernetes clusters should not allow container privilege escalation			Built-in	Policy	Kubernetes

Secret Store CSI Driver

Azure Key Vault provider for Secret Store CSI Driver allows you to get secret contents stored in an Azure Key Vault instance and use the Secrets Store CSI driver interface to mount them into Kubernetes pods.
https://github.com/Azure/secrets-store-csi-driver-provider-azure

Usage: az aks create –g <rg> -n <name>
install secret store csi driver into cluster
create your own secretProviderClass object
assign kubelet identity access to keyvault get access

More info

https://aka.ms/aks/releases
https://aka.ms/aks/roadmap
https://github.com/mspnp/aks-secure-baseline
https://www.the-aks-checklist.com/
https://azure.github.io/Aks-Construction/

Kubernetes Hardening Guide – NSA/CISA