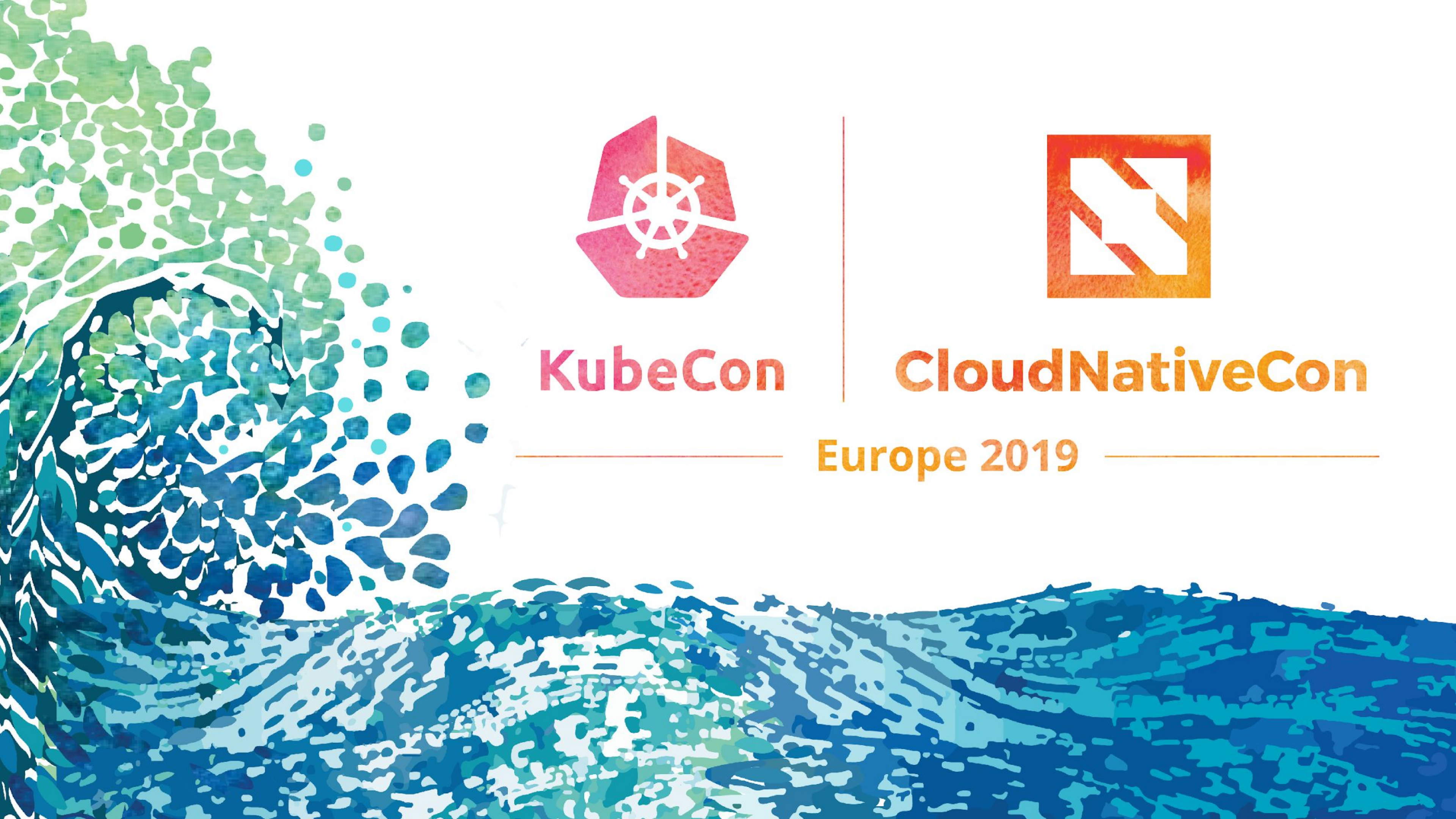


KubeCon | CloudNativeCon

Europe 2019

KubeCon | CloudNativeCon

Europe 2019

Lifecycle Of A kubectl Command:
Harden Kubernetes Setup With Automation

Sanjary Rahman, Booking.com

$ whoami

Sanjary Rahman

Site Reliability Engineer, Booking.com

sanjary.rahman@booking.com
link.sanjary.dev
twitter.sanjary.dev
instagram.sanjary.dev

Agenda

Agenda

- Multi-tenant cluster architecture in Booking.com

Agenda

- Multi-tenant cluster architecture in Booking.com
- Deployment workflow on Kubernetes in Booking.com

Agenda

- Multi-tenant cluster architecture in Booking.com
- Deployment workflow on Kubernetes in Booking.com
- Challenges faced managing those clusters

Agenda

- Multi-tenant cluster architecture in Booking.com
- Deployment workflow on Kubernetes in Booking.com
- Challenges faced managing those clusters
- Workspace provisioning automation

Agenda

- Multi-tenant cluster architecture in Booking.com
- Deployment workflow on Kubernetes in Booking.com
- Challenges faced managing those clusters
- Workspace provisioning automation
- Lifecycle of a kubectl command

Agenda

- Multi-tenant cluster architecture in Booking.com
- Deployment workflow on Kubernetes in Booking.com
- Challenges faced managing those clusters
- Workspace provisioning automation
- Lifecycle of a kubectl command
    - exec-credential plugin
    - Custom auth webhook

Agenda

- Multi-tenant cluster architecture in Booking.com
- Deployment workflow on Kubernetes in Booking.com
- Challenges faced managing those clusters
- Workspace provisioning automation
- Lifecycle of a kubectl command
  - exec-credential plugin
  - Custom auth webhook
  - Custom mutating admission webhook
  - Custom validating admission webhook

Agenda

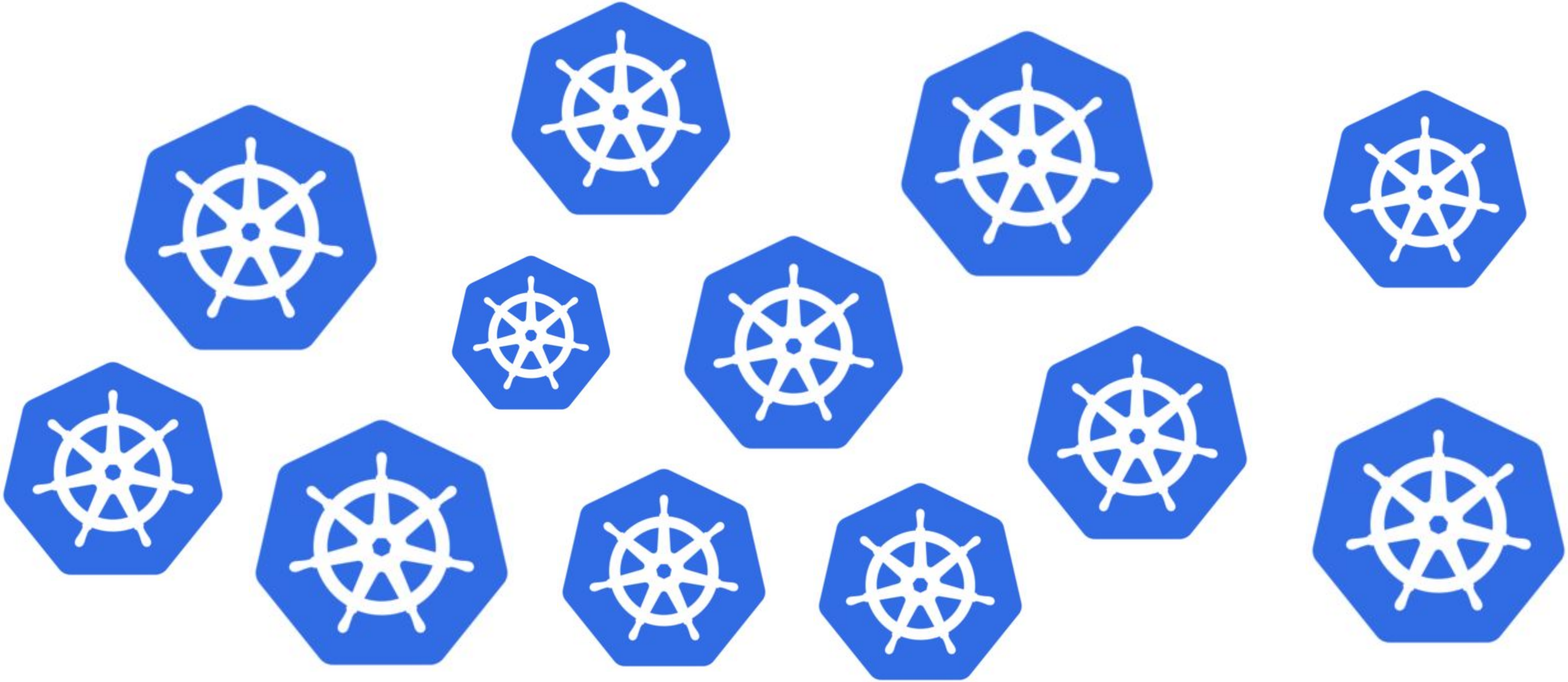
- Multi-tenant cluster architecture in Booking.com
- Deployment workflow on Kubernetes in Booking.com
- Challenges faced managing those clusters
- Workspace provisioning automation
- Lifecycle of a kubectl command
  - exec-credential plugin
  - Custom auth webhook
  - Custom mutating admission webhook
  - Custom validating admission webhook
  - Pod Security Policy

Agenda

- Multi-tenant cluster architecture in Booking.com
- Deployment workflow on Kubernetes in Booking.com
- Challenges faced managing those clusters
- Workspace provisioning automation
- Lifecycle of a kubectl command
    - exec-credential plugin
    - Custom auth webhook
    - Custom mutating admission webhook
    - Custom validating admission webhook
    - Pod Security Policy
- Q/A

Kubernetes Clusters in Booking.com

Kubernetes Clusters in Booking.com

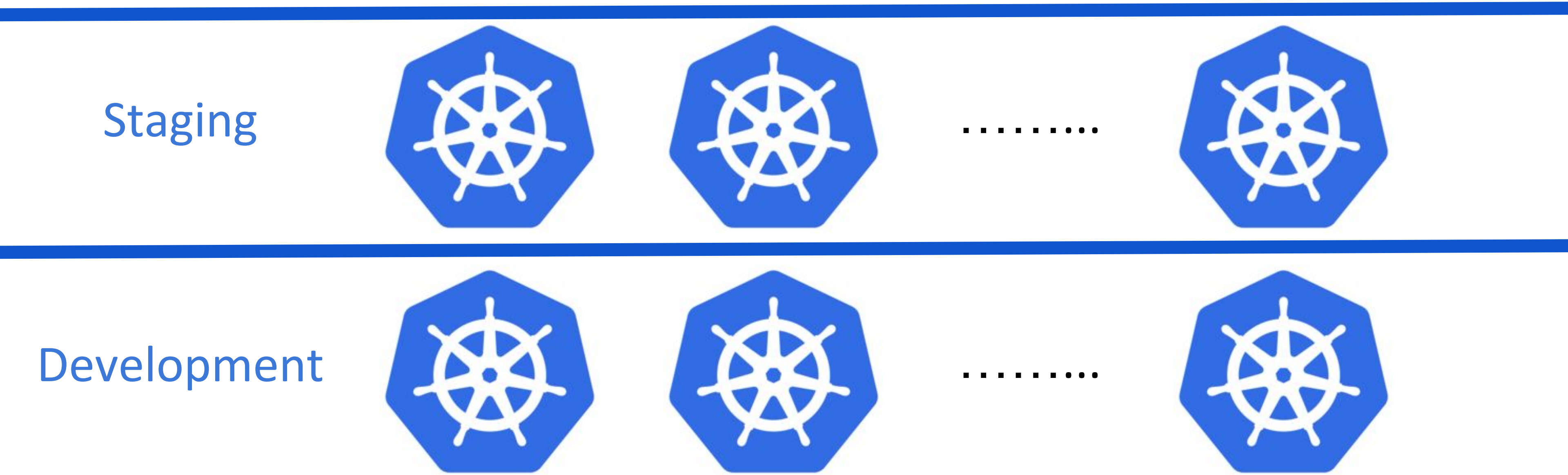


Kubernetes Clusters in Booking.com

Kubernetes Clusters in Booking.com

Development

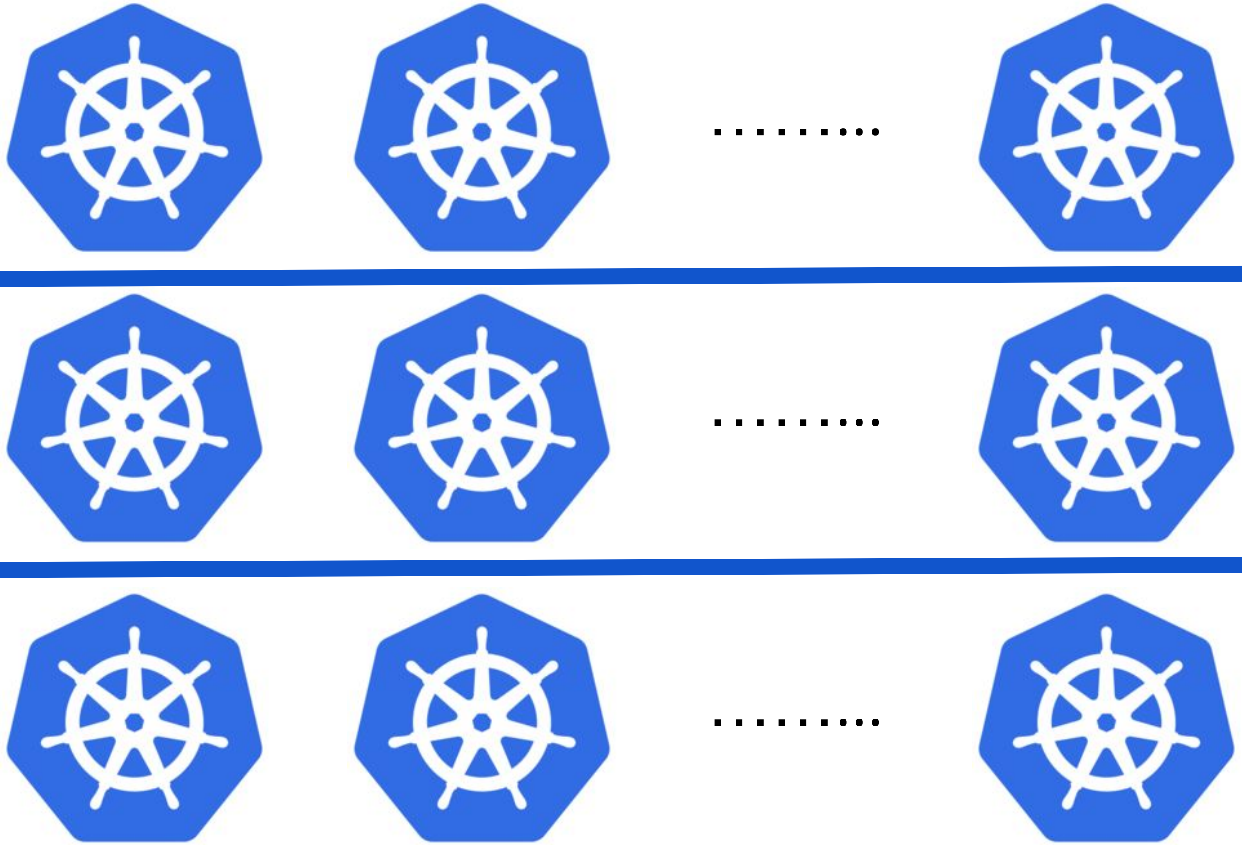
Kubernetes Clusters in Booking.com



Staging

Development

Kubernetes Clusters in Booking.com



Production

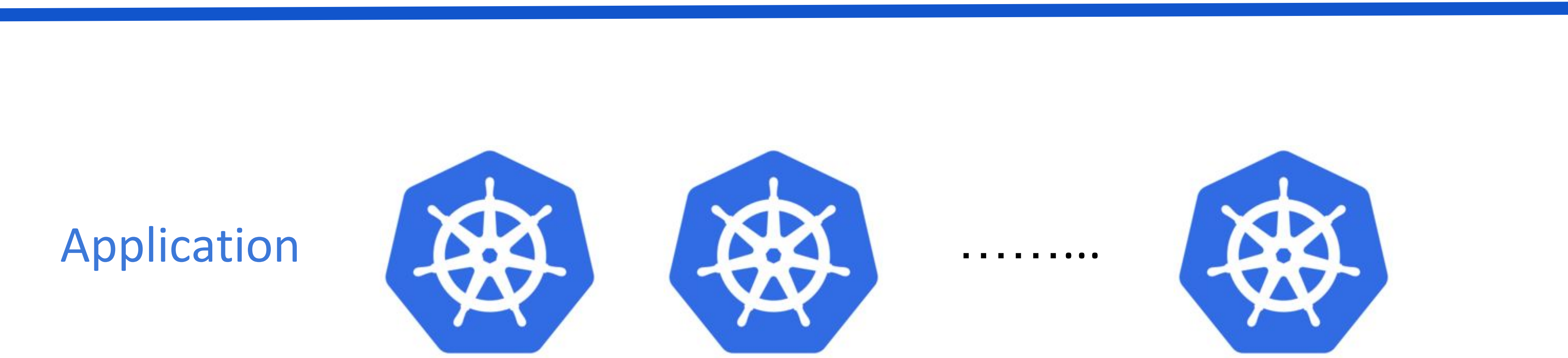
Staging

Development

Deployment Workflow

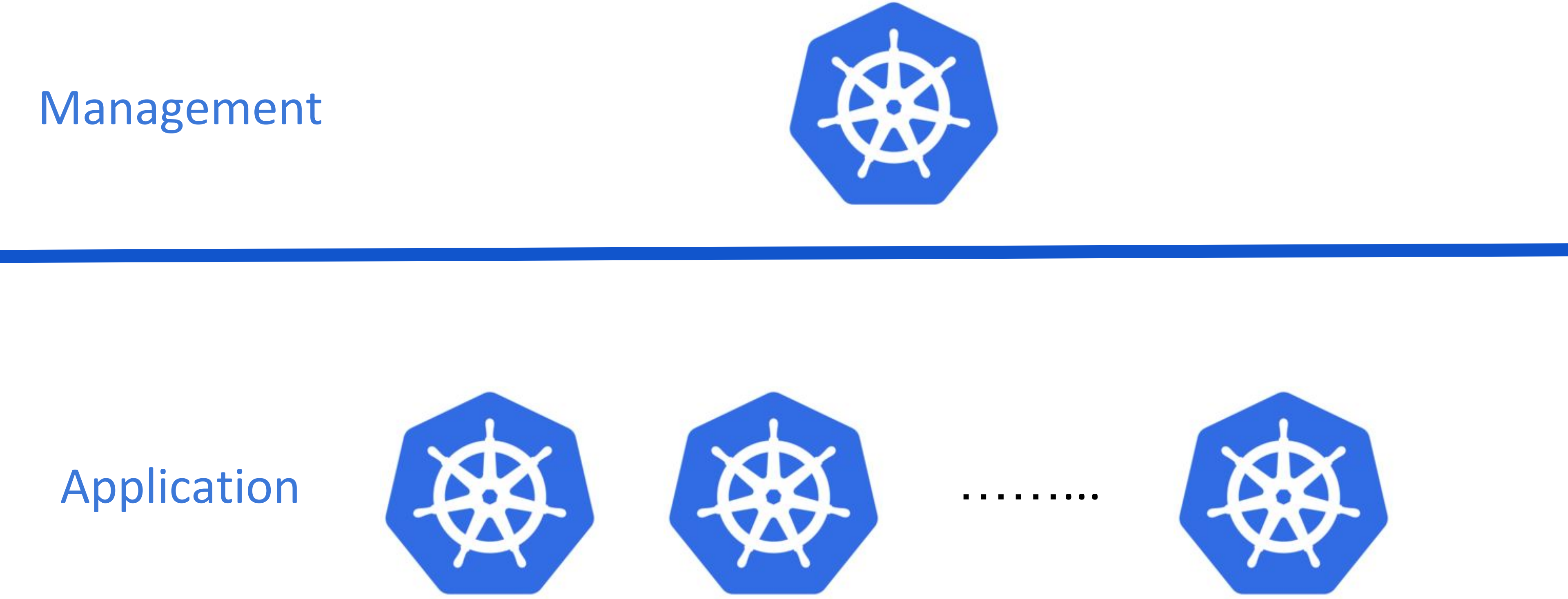
Deployment Workflow

Deployment Workflow



Application

Deployment Workflow

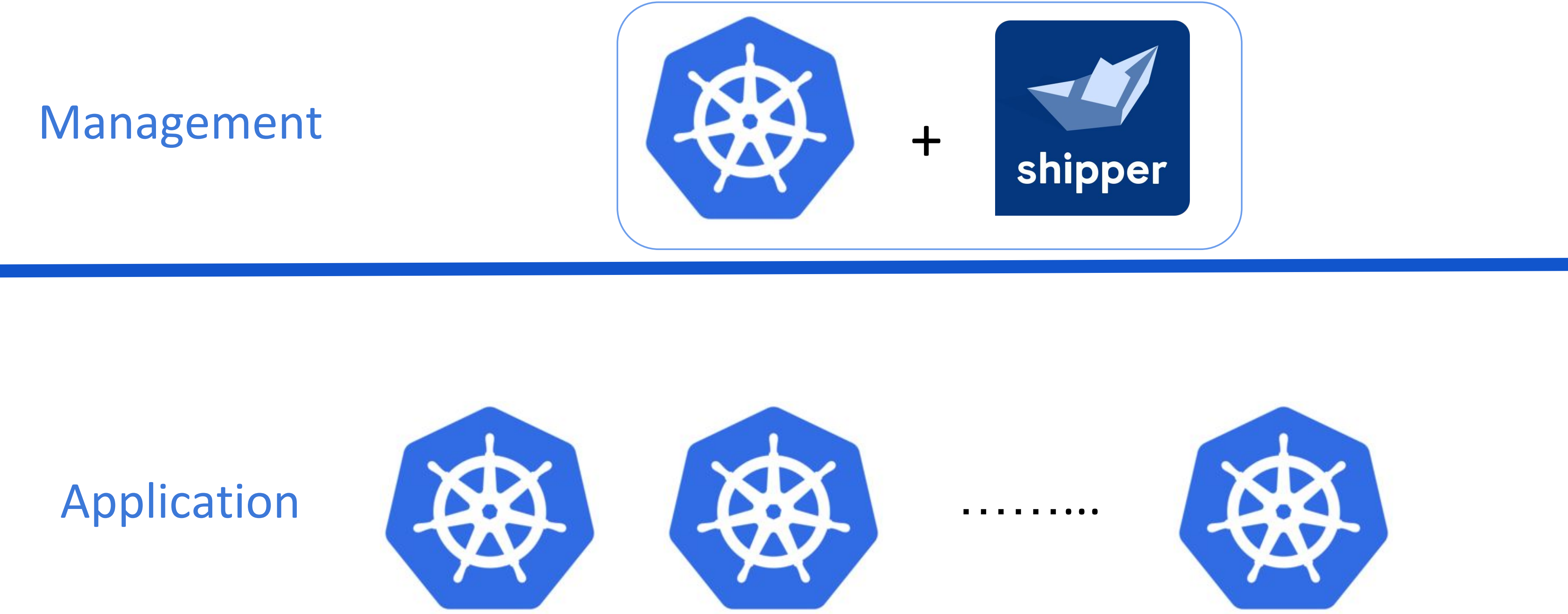


Management

Application

....

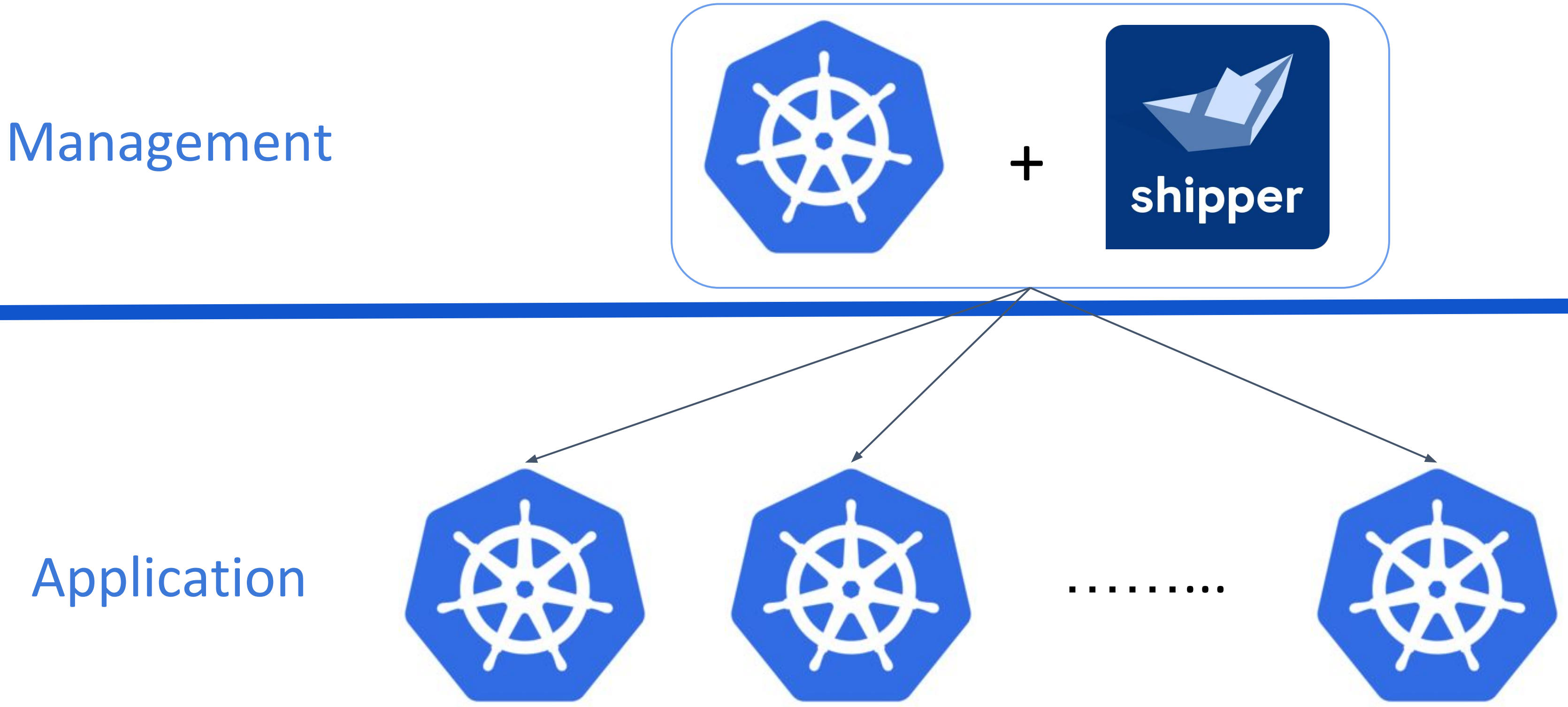
Deployment Workflow



Management

Application

Deployment Workflow



Management

Application

More info on shipper

Read more about shipper here:

https://shipper-k8s.io
https://docs.shipper-k8s.io

Challenges In Multitenant Clusters

Challenges

Challenges

Solution

Challenges

Challenges

- Project management

Solution

- Kubernetes namespaces

Challenges

Challenges	Solution
- Project management	- Kubernetes namespaces
- Resource management	- ResourceQuotas

Challenges

Challenges	Solution
- Project management	- Kubernetes namespaces
- Resource management	- ResourceQuotas
- Auth management	- Rolebindings + Auth Webhook

Challenges

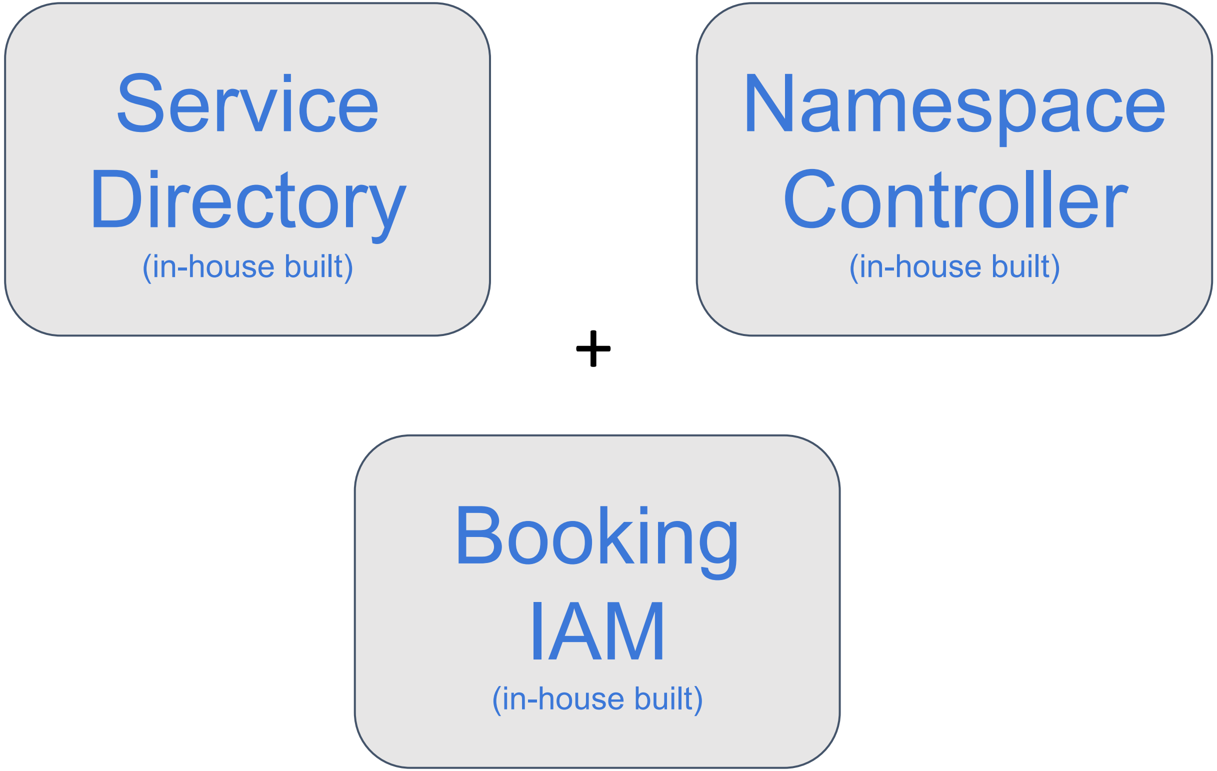
Challenges	Solution
- Project management	- Kubernetes namespaces
- Resource management	- ResourceQuotas
- Auth management	- Rolebindings + Auth Webhook
- Config management	- Configmaps

Challenges

Challenges	Solution
- Project management	- Kubernetes namespaces
- Resource management	- ResourceQuotas
- Auth management	- Rolebindings + Auth Webhook
- Config management	- Configmaps
- Validation and safeguards	- Admission Webhooks + PSP

Workspace Provisioning

Workspace Provisioning



Service Directory (in-house built)

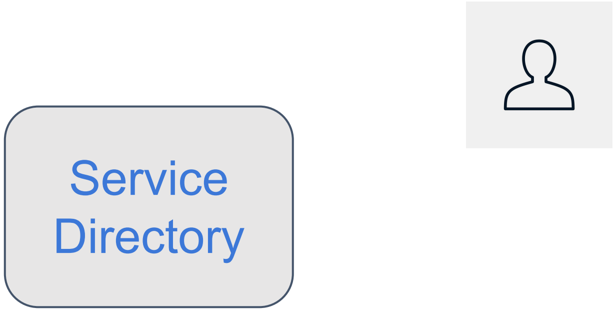
+

Namespace Controller (in-house built)

Booking IAM (in-house built)

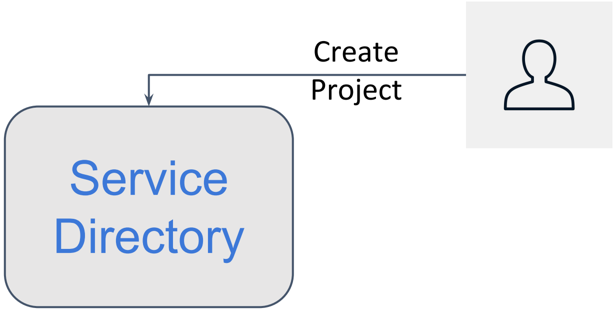
Workspace Provisioning

Workspace Provisioning

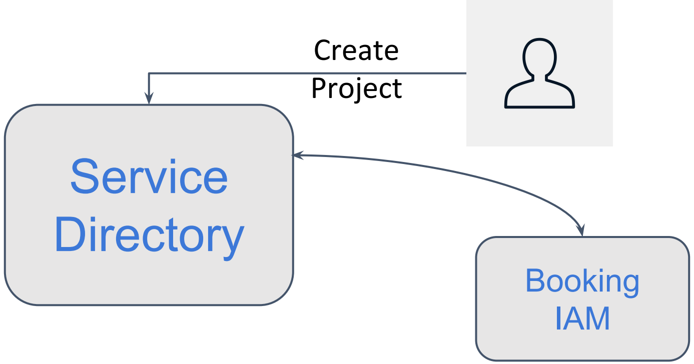


Service Directory

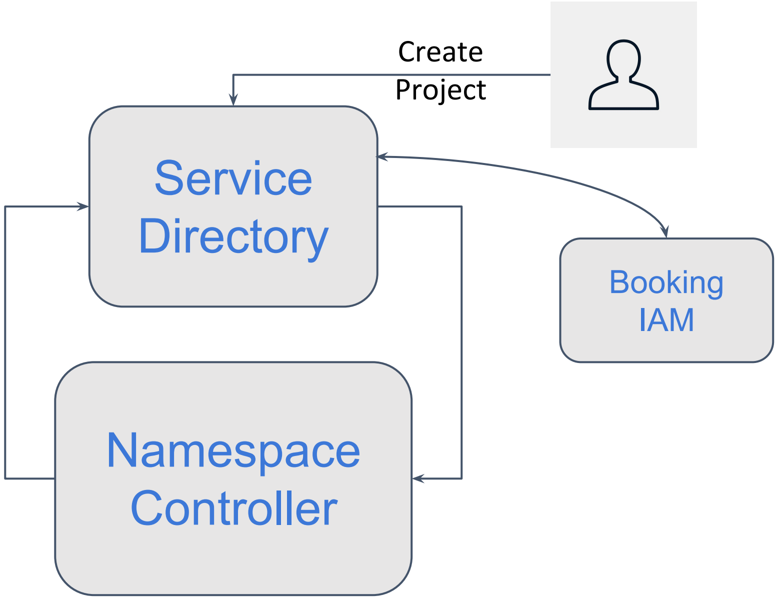
Workspace Provisioning



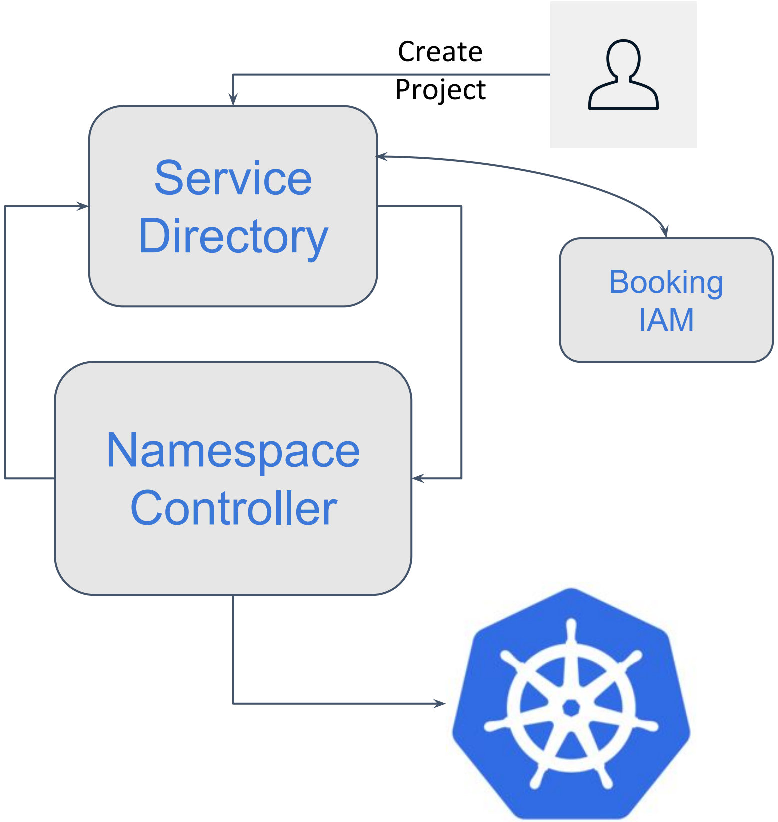
Workspace Provisioning



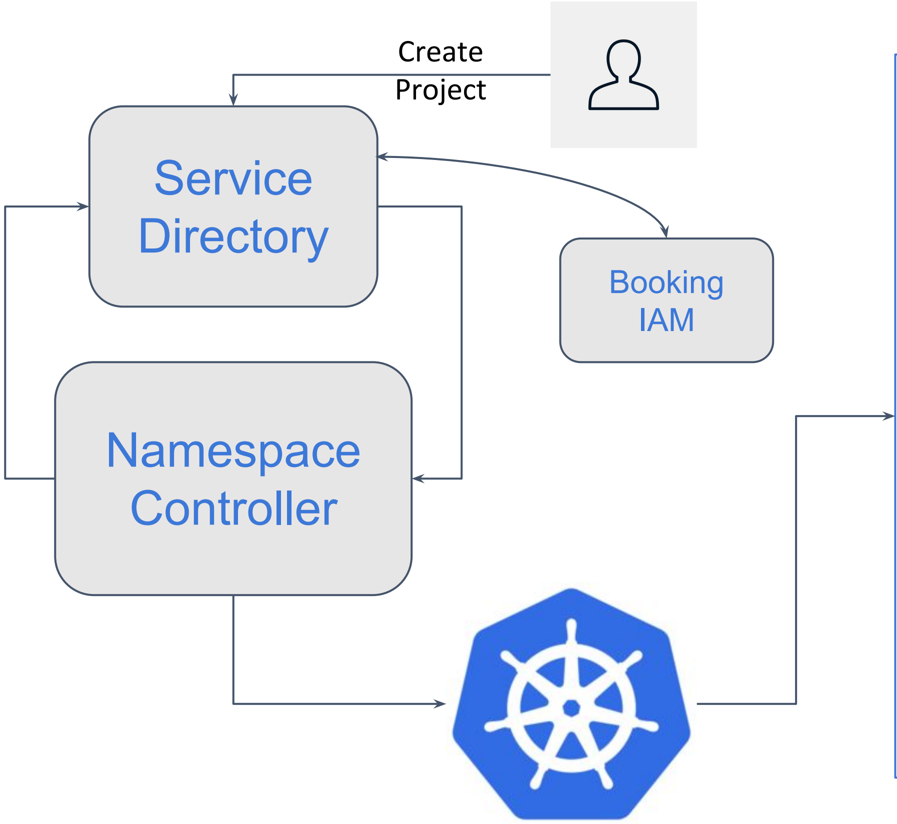
Workspace Provisioning



Workspace Provisioning



Workspace Provisioning

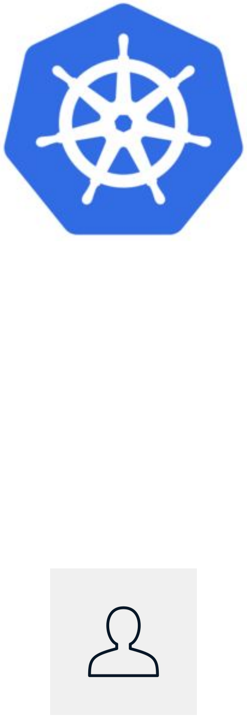


- Creates Namespace
- Creates Rolebinding
- Creates ResourceQuota
- Creates LimitRanges
- Creates Configmaps

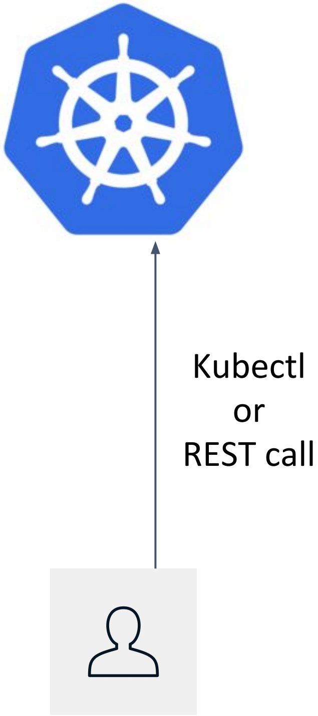
Lifecycle of kubectl command

Lifecycle of kubectl command

Lifecycle of kubectl command

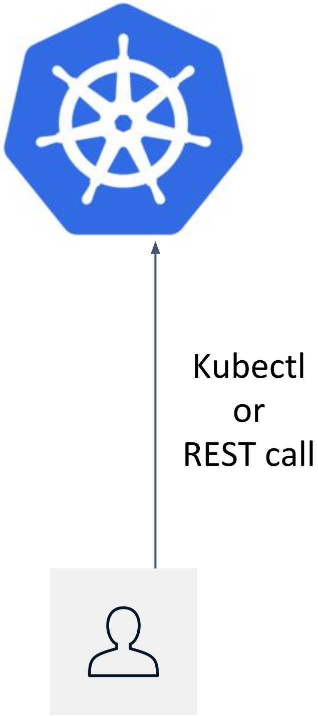


Lifecycle of kubectl command



Kubectl
or
REST call

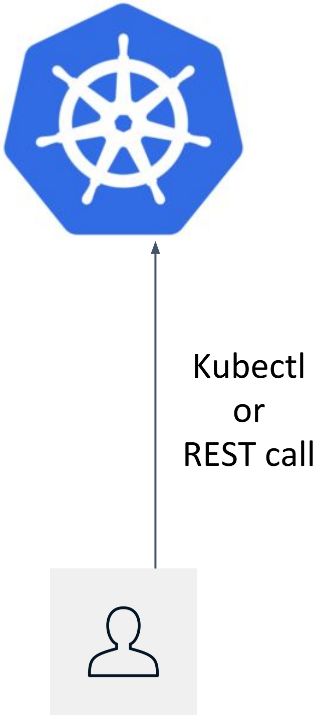
Lifecycle of kubectl command



exec-credential Plugin

```
apiVersion: v1
clusters:
- cluster:
    server: https://auth.example.com
  name: my-cluster
contexts:
- context:
    cluster: my-cluster
    namespace: my-ns
    user: example
  name: my-context
current-context: my-context
kind: Config
users:
- name: example
  user:
    exec:
      apiVersion: client.authentication.k8s.io/v1alpha1
      command: /usr/local/bin/generate-bearer-token
```

Lifecycle of kubectl command

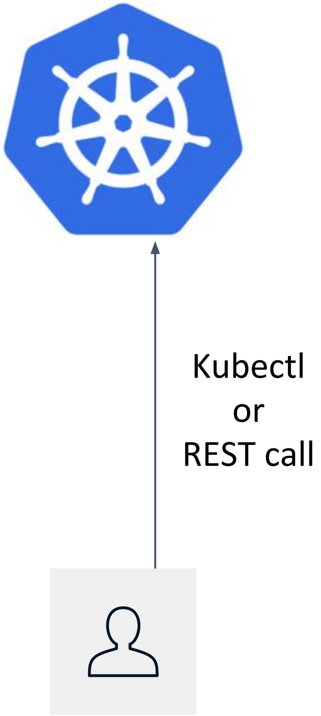


Kubectl
or
REST call

exec-credential Plugin

```
apiVersion: v1
clusters:
- cluster:
    server: https://auth.example.com
  name: my-cluster
contexts:
- context:
    cluster: my-cluster
    namespace: my-ns
    user: example
  name: my-context
current-context: my-context
kind: Config
users:
- name: example
  user:
    exec:
      apiVersion: client.authentication.k8s.io/v1alpha1
      command: /usr/local/bin/generate-bearer-token
```

Lifecycle of kubectl command

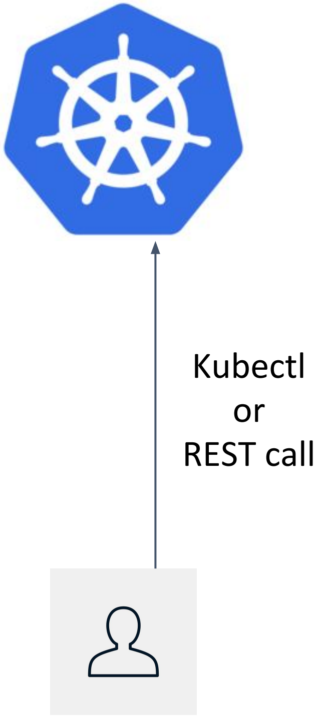


Kubectl
or
REST call

exec-credential Plugin

```
apiVersion: v1
clusters:
- cluster:
    server: https://auth.example.com
  name: my-cluster
contexts:
- context:
    cluster: my-cluster
    namespace: my-ns
    user: example
  name: my-context
current-context: my-context
kind: Config
users:
- name: example
  user:
    exec:
      apiVersion: client.authentication.k8s.io/v1alpha1
      command: /usr/local/bin/generate-bearer-token
```

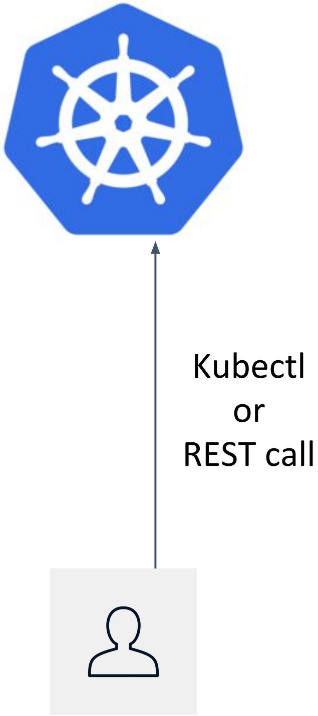
Lifecycle of kubectl command



exec-credential Plugin

```
apiVersion: v1
clusters:
- cluster:
    server: https://auth.example.com
  name: my-cluster
contexts:
- context:
    cluster: my-cluster
    namespace: my-ns
    user: example
  name: my-context
current-context: my-context
kind: Config
users:
- name: example
  user:
    exec:
      apiVersion: client.authentication.k8s.io/v1alpha1
      command: /usr/local/bin/generate-bearer-token
```

Lifecycle of kubectl command

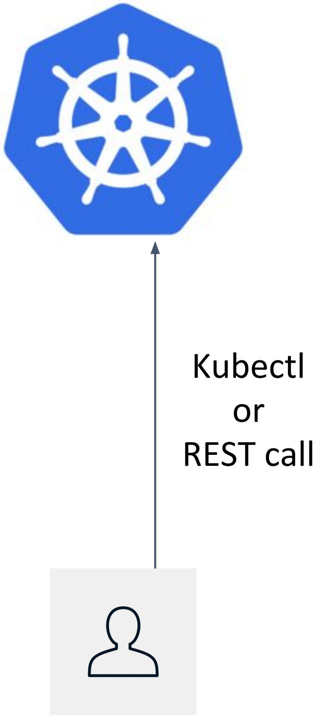


Kubectl
or
REST call

```
Example output:
```

```
{
  "apiVersion": "client.authentication.k8s.io/v1beta1",
  "kind": "ExecCredential",
  "status": {
    "token": "my-bearer-token",
    "expirationTimestamp": "2019-23-05T17:30:20-08:00"
  }
}
```

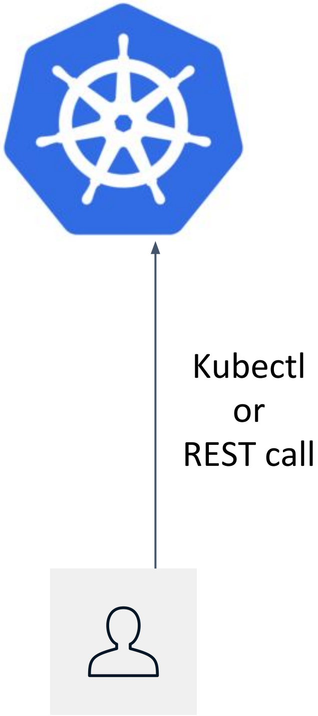
Lifecycle of kubectl command



Example output:

```
{
  "apiVersion": "client.authentication.k8s.io/v1beta1",
  "kind": "ExecCredential",
  "status": {
    "token": "my-bearer-token",
    "expirationTimestamp": "2019-23-05T17:30:20-08:00"
  }
}
```

Lifecycle of kubectl command

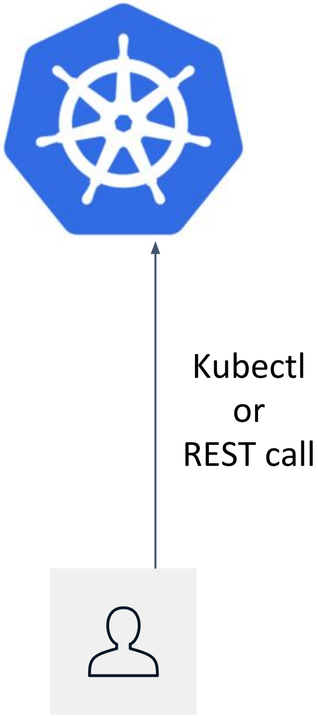


Kubectl
or
REST call

`Example output:`

```
{
  "apiVersion": "client.authentication.k8s.io/v1beta1",
  "kind": "ExecCredential",
  "status": {
    "token": "my-bearer-token",
    "expirationTimestamp": "2019-23-05T17:30:20-08:00"
  }
}
```

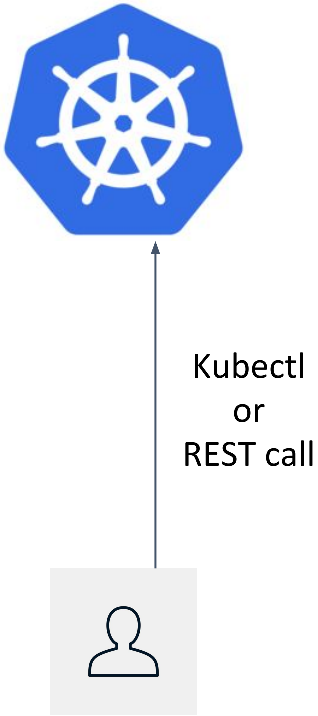
Lifecycle of kubectl command



Example output:

```
{
  "apiVersion": "client.authentication.k8s.io/v1beta1",
  "kind": "ExecCredential",
  "status": {
    "token": "my-bearer-token",
    "expirationTimestamp": "2019-23-05T17:30:20-08:00"
  }
}
```

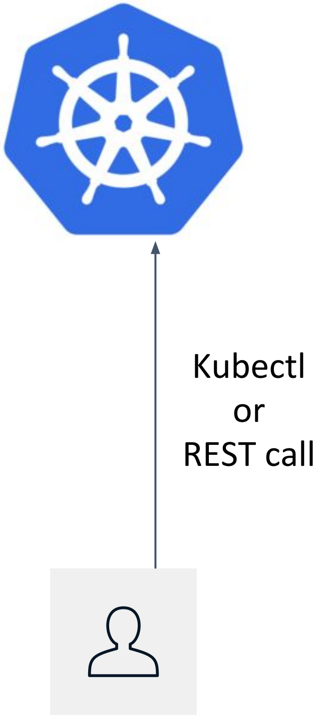
Lifecycle of kubectl command



`Example output:`

```
{
  "apiVersion": "client.authentication.k8s.io/v1beta1",
  "kind": "ExecCredential",
  "status": {
    "token": "my-bearer-token",
    "expirationTimestamp": "2019-23-05T17:30:20-08:00"
  }
}
```

Lifecycle of kubectl command

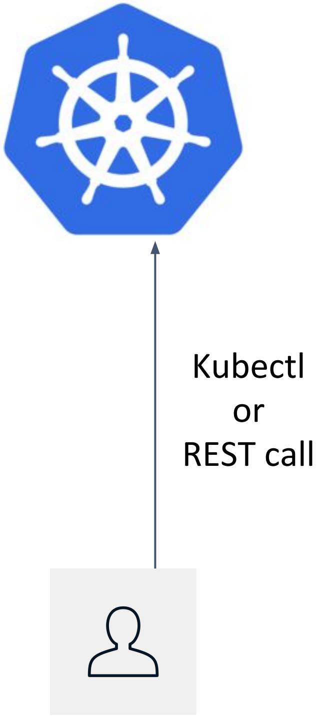


Kubectl
or
REST call

`Example output:`

```
{
  "apiVersion": "client.authentication.k8s.io/v1beta1",
  "kind": "ExecCredential",
  "status": {
    "token": "my-bearer-token",
    "expirationTimestamp": "2019-23-05T17:30:20-08:00"
  }
}
```

Lifecycle of kubectl command

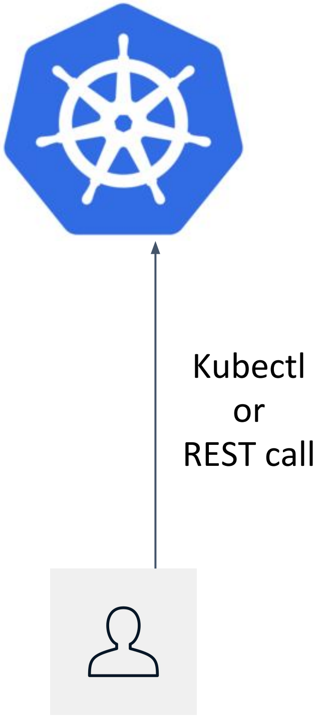


Kubectl
or
REST call

exec-credential Plugin

```
apiVersion: v1
clusters:
- cluster:
    server: https://auth.example.com
  name: my-cluster
contexts:
- context:
    cluster: my-cluster
    namespace: my-ns
    user: example
  name: my-context
current-context: my-context
kind: Config
users:
- name: example
  user:
    exec:
      apiVersion: client.authentication.k8s.io/v1alpha1
      command: /usr/local/bin/generate-bearer-token
```

Lifecycle of kubectl command

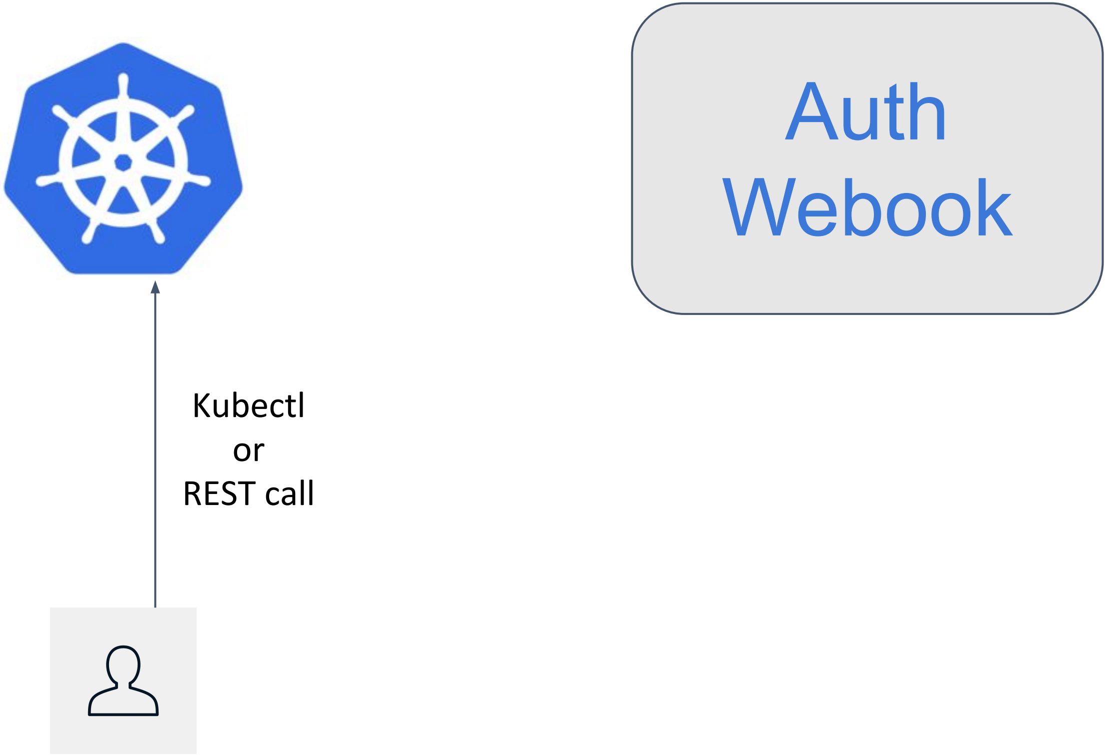


Kubectl
or
REST call

`Example output:`

```
{
  "apiVersion": "client.authentication.k8s.io/v1beta1",
  "kind": "ExecCredential",
  "status": {
    "token": "my-bearer-token",
    "expirationTimestamp": "2019-23-05T17:30:20-08:00"
  }
}
```

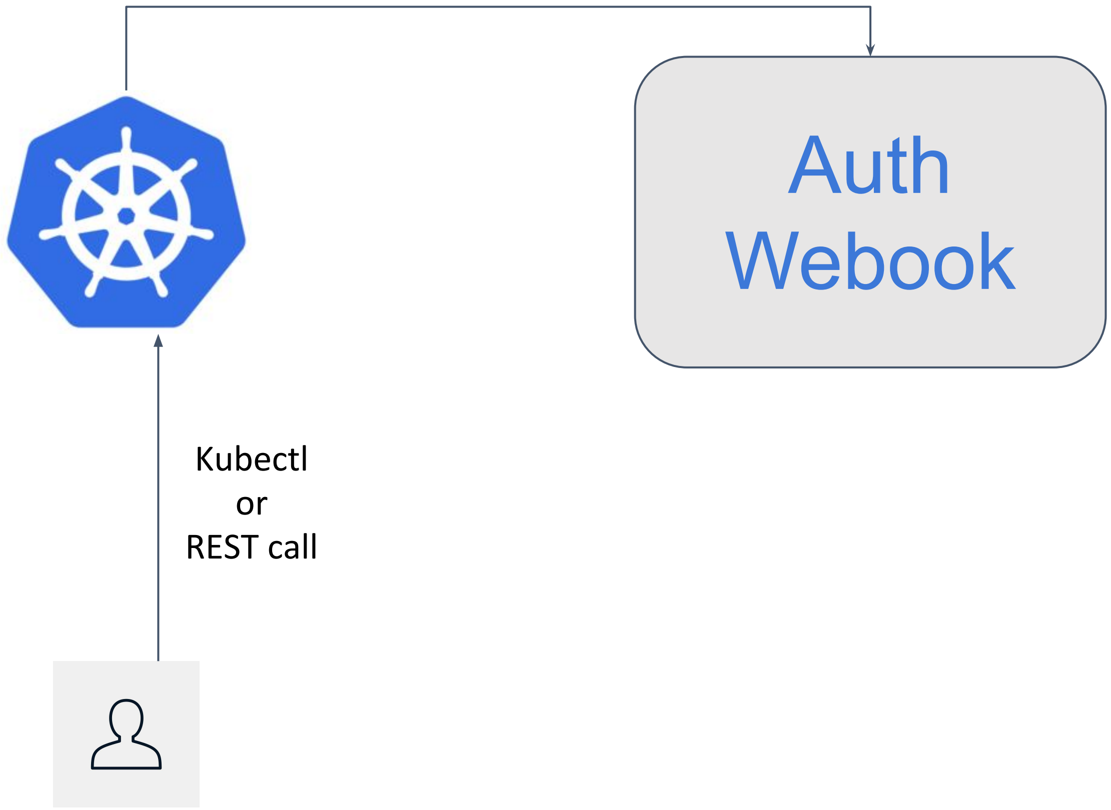
Lifecycle of kubectl command



Kubectl
or
REST call

Auth
Webook

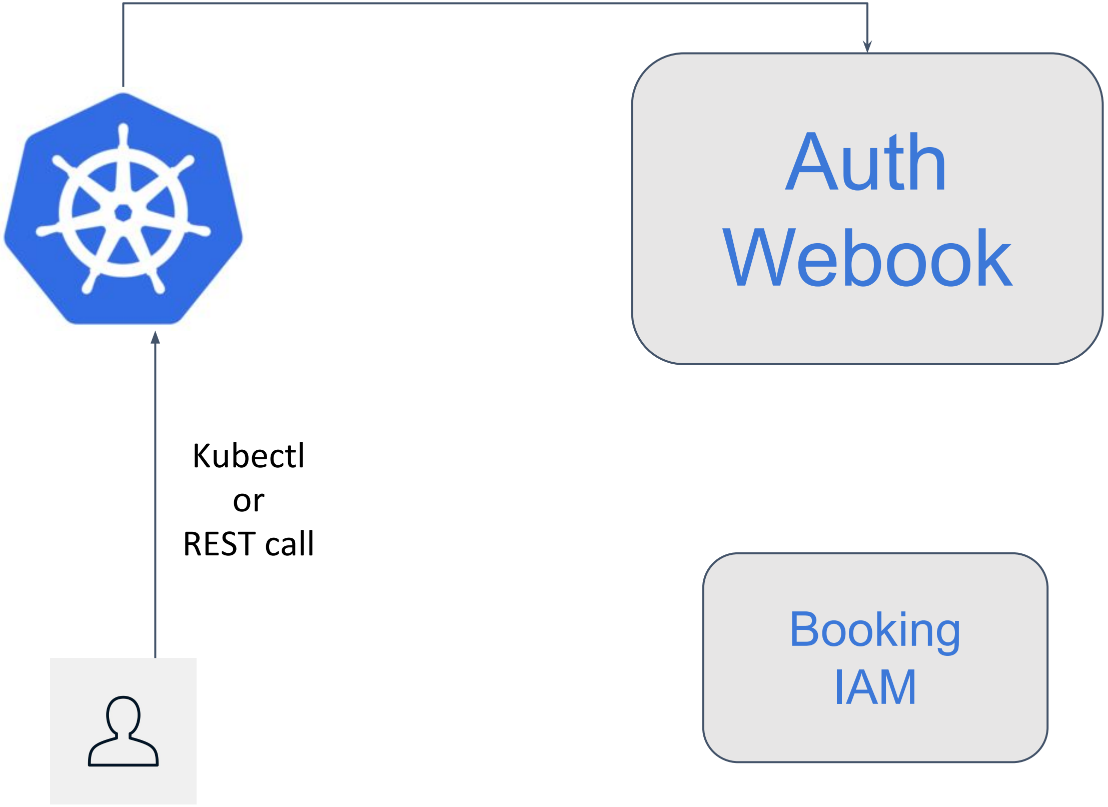
Lifecycle of kubectl command



Auth Webook

Kubectl or REST call

Lifecycle of kubectl command

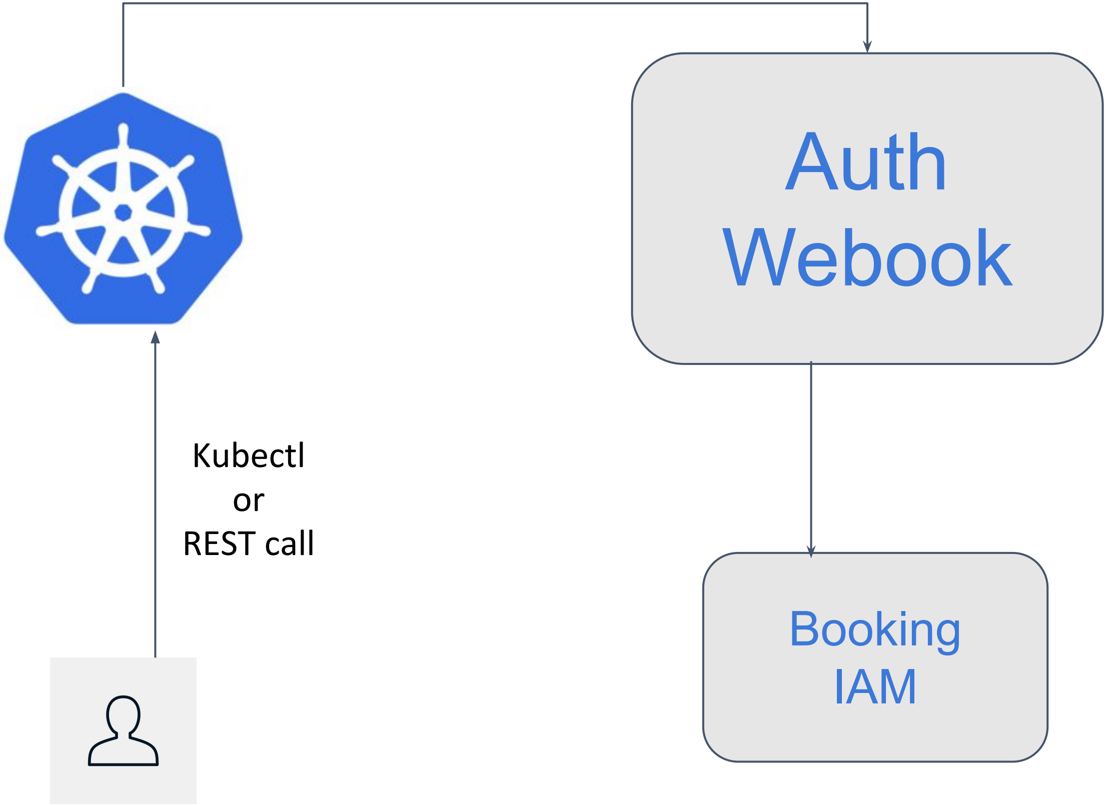


Auth Webook

Kubectl or REST call

Booking IAM

Lifecycle of kubectl command

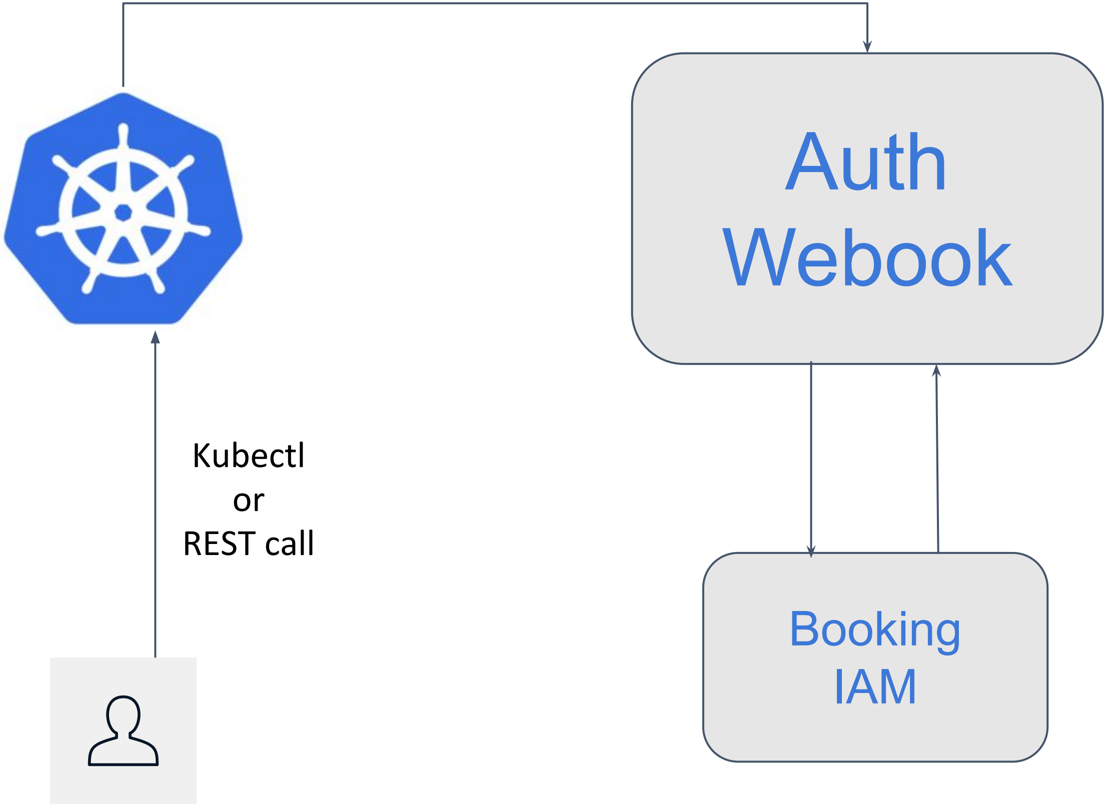


Auth Webook

Booking IAM

Kubectl or REST call

Lifecycle of kubectl command

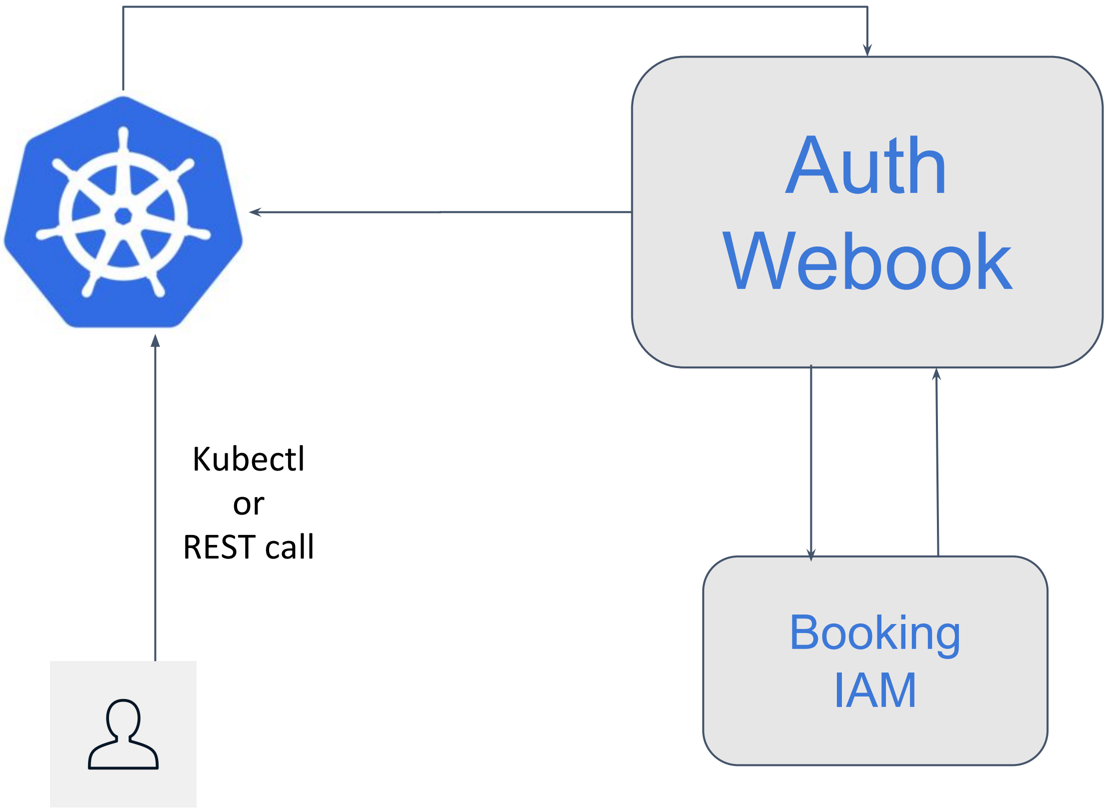


Auth Webook

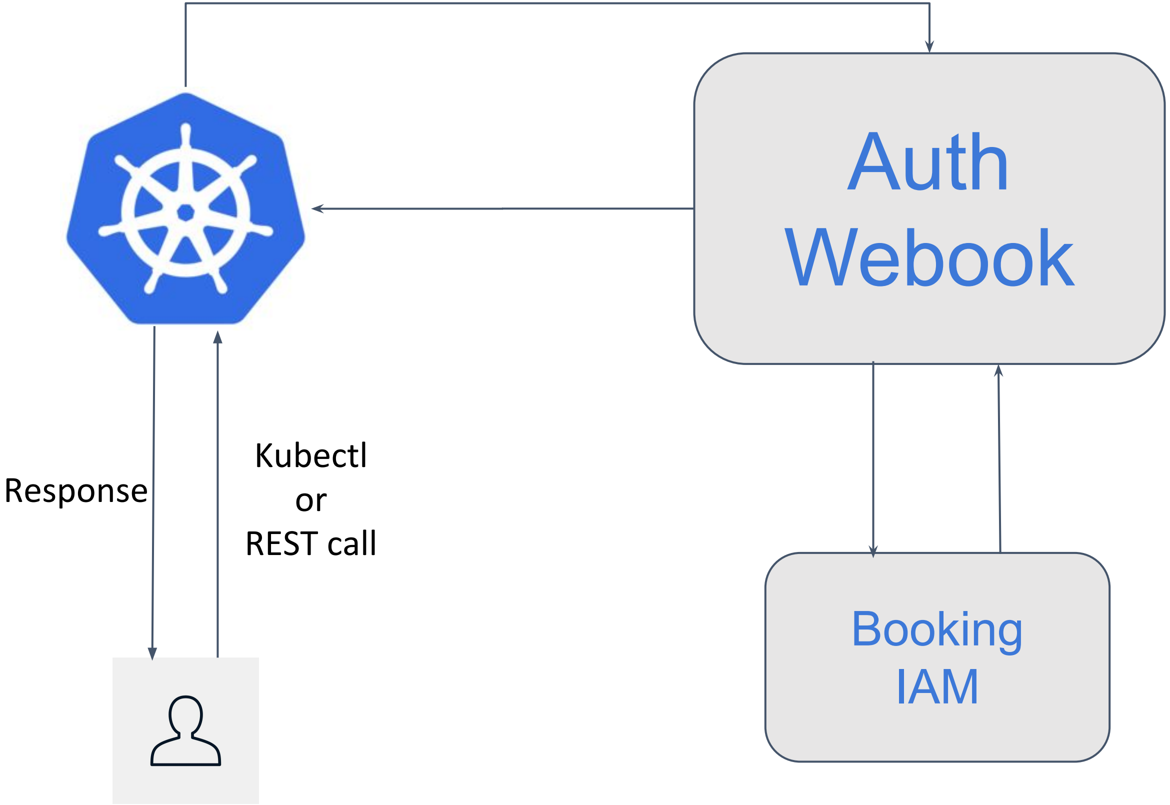
Booking IAM

Kubectl
or
REST call

Lifecycle of kubectl command



Lifecycle of kubectl command



Auth Webhook

Configure kube-apiserver with the webhook flag

```
--authorization-webhook-config-file=SOME_FILENAME
```

Auth Webhook

Configure kube-apiserver with the webhook flag

```
--authorization-webhook-config-file=SOME_FILENAME
```

```
apiVersion: v1
kind: Config
clusters:
  - name: name-of-remote-authz-service
    cluster:
      certificate-authority: /path/to/ca.pem
      # Webhook URL (must be https)
      server: https://authz.example.com/authorize
users:
  - name: name-of-api-server
    user:
      client-certificate: /path/to/cert.pem
      client-key: /path/to/key.pem
contexts:
- context:
    cluster: name-of-remote-authz-service
    user: name-of-api-server
  name: auth-webhook
current-context: auth-webhook
```

Auth Webhook

Configure kube-apiserver with the webhook flag

```
--authorization-webhook-config-file=SOME_FILENAME
```

```yaml
apiVersion: v1
kind: Config
clusters:
  - name: name-of-remote-authz-service
    cluster:
      certificate-authority: /path/to/ca.pem
      # Webhook URL (must be https)
      server: https://authz.example.com/authorize
users:
  - name: name-of-api-server
    user:
      client-certificate: /path/to/cert.pem
      client-key: /path/to/key.pem
contexts:
- context:
    cluster: name-of-remote-authz-service
    user: name-of-api-server
  name: auth-webhook
current-context: auth-webhook
```

Auth Webhook

Configure kube-apiserver with the webhook flag

```
--authorization-webhook-config-file=SOME_FILENAME
```

```yaml
apiVersion: v1
kind: Config
clusters:
  - name: name-of-remote-authz-service
    cluster:
      certificate-authority: /path/to/ca.pem
      # Webhook URL (must be https)
      server: https://authz.example.com/authorize
users:
  - name: name-of-api-server
    user:
      client-certificate: /path/to/cert.pem
      client-key: /path/to/key.pem
contexts:
- context:
    cluster: name-of-remote-authz-service
    user: name-of-api-server
  name: auth-webhook
current-context: auth-webhook
```

Auth Webhook

Configure kube-apiserver with the webhook flag

```
--authorization-webhook-config-file=SOME_FILENAME
```

```
apiVersion: v1
kind: Config
clusters:
  - name: name-of-remote-authz-service
    cluster:
      certificate-authority: /path/to/ca.pem
      # Webhook URL (must be https)
      server: https://authz.example.com/authorize
users:
  - name: name-of-api-server
    user:
      client-certificate: /path/to/cert.pem
      client-key: /path/to/key.pem
contexts:
- context:
    cluster: name-of-remote-authz-service
    user: name-of-api-server
  name: auth-webhook
current-context: auth-webhook
```

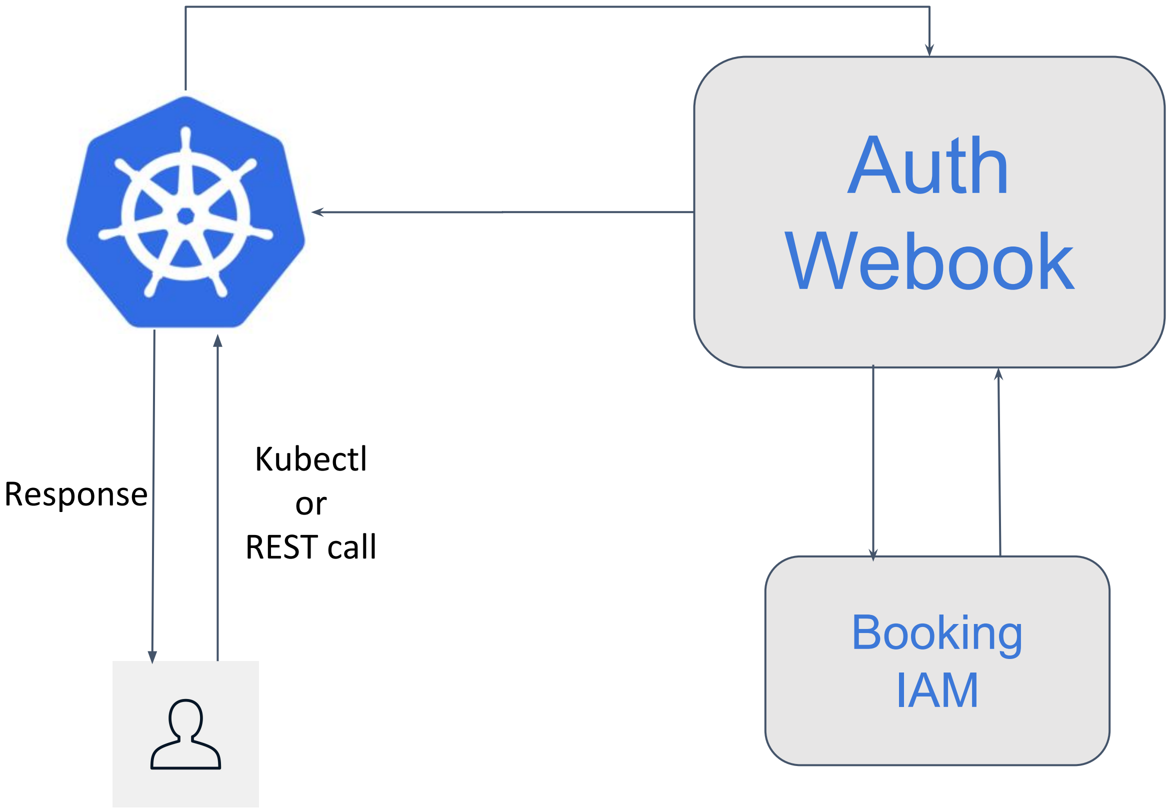
Auth Webhook

Configure kube-apiserver with the webhook flag

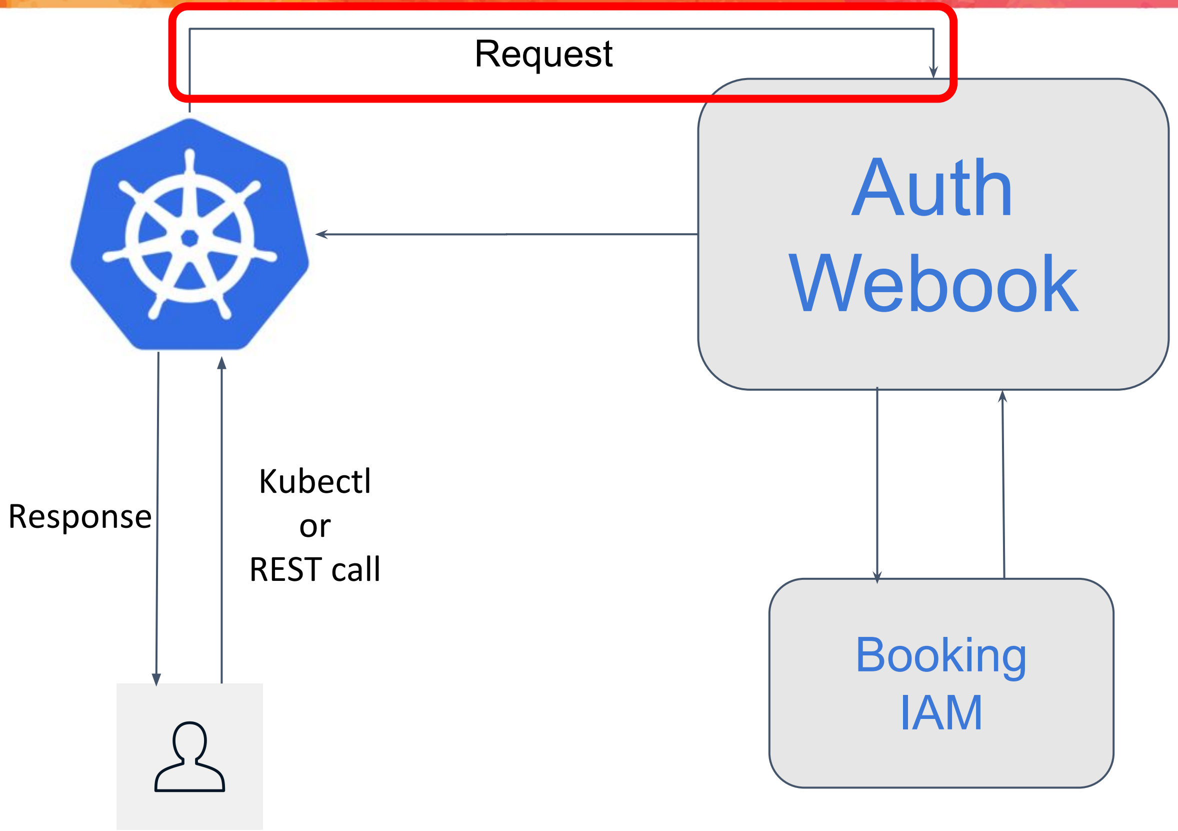
```
--authorization-webhook-config-file=SOME_FILENAME
```

```
apiVersion: v1
kind: Config
clusters:
  - name: name-of-remote-authz-service
    cluster:
      certificate-authority: /path/to/ca.pem
      # Webhook URL (must be https)
      server: https://authz.example.com/authorize
users:
  - name: name-of-api-server
    user:
      client-certificate: /path/to/cert.pem
      client-key: /path/to/key.pem
contexts:
- context:
    cluster: name-of-remote-authz-service
    user: name-of-api-server
  name: auth-webhook
current-context: auth-webhook
```

Lifecycle of kubectl command



Lifecycle of kubectl command



Auth Webhook

KubeCon CloudNativeCon Europe 2019

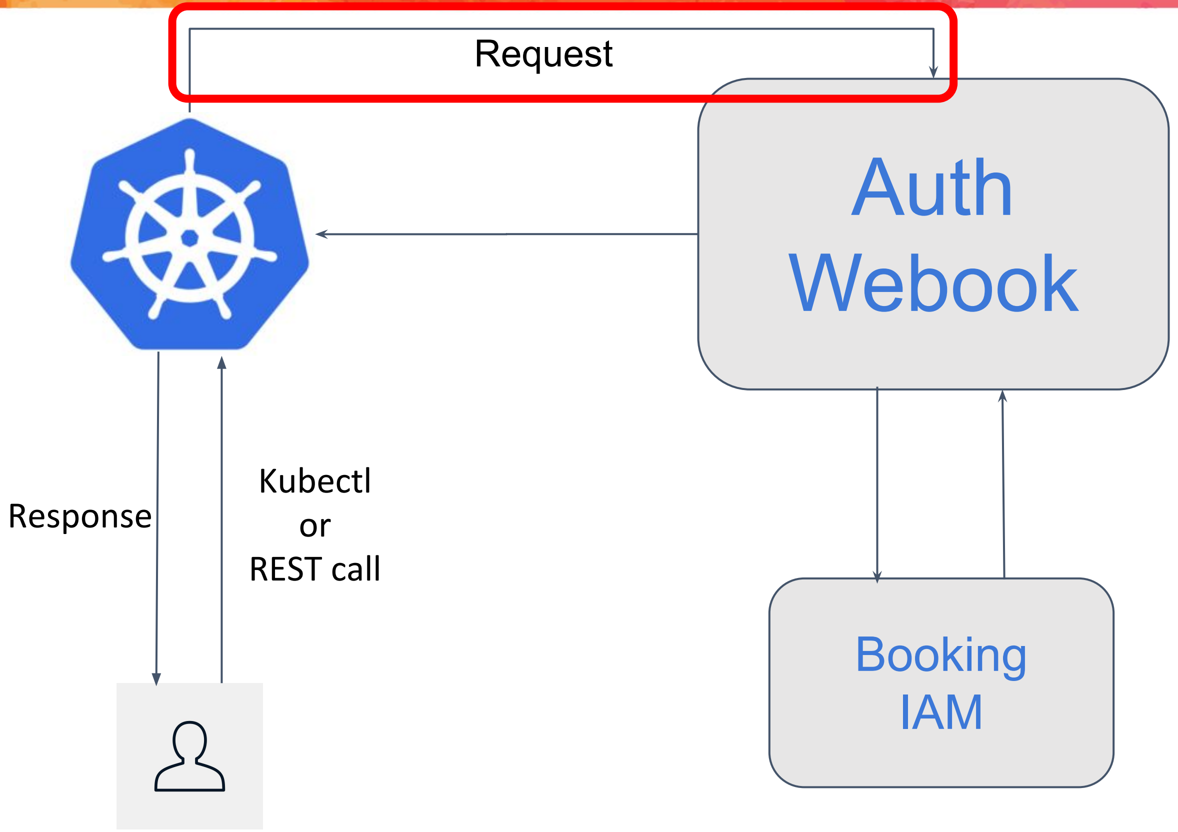
Example request:

```
{
  "apiVersion":
"authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "spec": {
    "resourceAttributes": {
      "namespace": "my-project",
      "verb": "get",
      "group": "apps/v1",
      "resource": "deployments"
    },
    "user": "sanjary",
    "group": []
  }
}
```

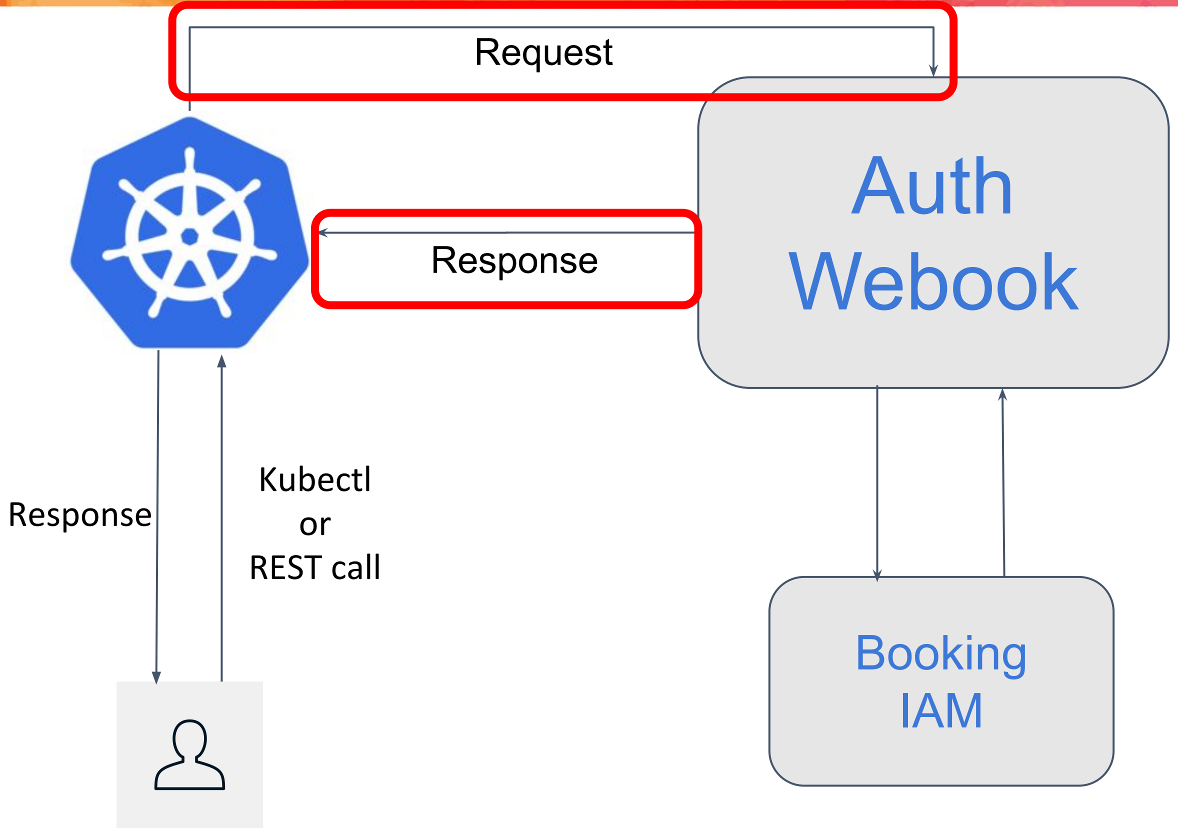
Example response:

```
Allow:
{
  "apiVersion": "authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "status": {
    "allowed": true
  }
   "user": "sanjary",
    "group": [
      "my-project:admin"
    ]
}
Deny:
{
  "apiVersion": "authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "status": {
    "allowed": false,
    "reason": "user sanjary does not have read access
to the namespace"
  }
}
```

Lifecycle of kubectl command



Lifecycle of kubectl command



Auth Webhook

Example request:

```
{
  "apiVersion":
"authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "spec": {
    "resourceAttributes": {
      "namespace": "my-project",
      "verb": "get",
      "group": "apps/v1",
      "resource": "deployments"
    },
    "user": "sanjary",
    "group": []
  }
}
```

Example response:

```
Allow:
{
  "apiVersion": "authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "status": {
    "allowed": true
  }
   "user": "sanjary",
    "group": [
      "my-project:admin"
    ]
}
Deny:
{
  "apiVersion": "authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "status": {
    "allowed": false,
    "reason": "user sanjary does not have read access
to the namespace"
  }
}
```

Auth Webhook

Example request:

```
{
  "apiVersion":
"authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "spec": {
    "resourceAttributes": {
      "namespace": "my-project",
      "verb": "get",
      "group": "apps/v1",
      "resource": "deployments"
    },
    "user": "sanjary",
    "group": []
  }
}
```

Example response:

```
Allow:
{
  "apiVersion": "authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "status": {
    "allowed": true
  }
   "user": "sanjary",
    "group": [
      "my-project:admin"
    ]
}
Deny:
{
  "apiVersion": "authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "status": {
    "allowed": false,
    "reason": "user sanjary does not have read access
to the namespace"
  }
}
```

Auth Webhook

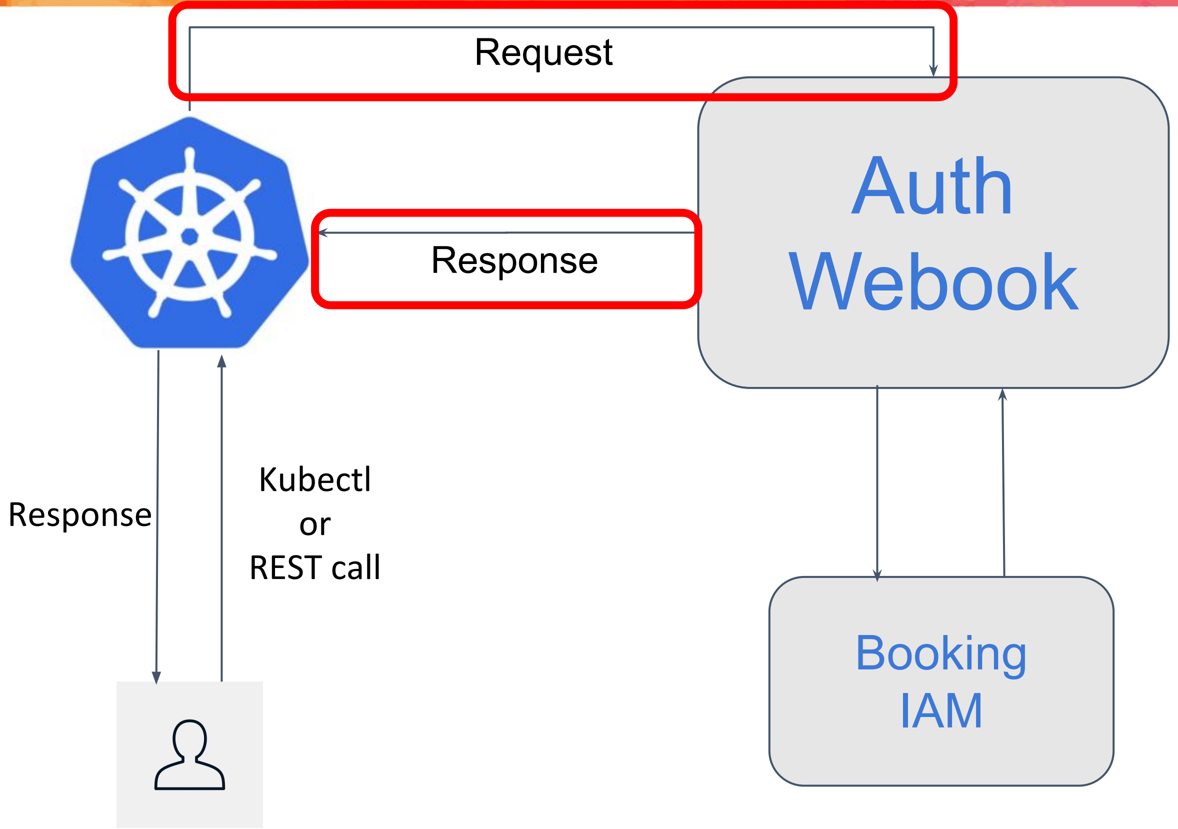
Example request:

```
{
  "apiVersion":
"authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "spec": {
    "resourceAttributes": {
      "namespace": "my-project",
      "verb": "get",
      "group": "apps/v1",
      "resource": "deployments"
    },
    "user": "sanjary",
    "group": []
  }
}
```

Example response:

```
Allow:
{
  "apiVersion": "authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "status": {
    "allowed": true
  }
   "user": "sanjary",
    "group": [
      "my-project:admin"
    ]
}
Deny:
{
  "apiVersion": "authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "status": {
    "allowed": false,
    "reason": "user sanjary does not have read access
to the namespace"
  }
}
```

Lifecycle of kubectl command



Request

Response

Auth Webook

Response

Kubectl
or
REST call

Booking IAM

Auth Webhook

Example request:

```
{
  "apiVersion":
"authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "spec": {
    "resourceAttributes": {
      "namespace": "my-project",
      "verb": "get",
      "group": "apps/v1",
      "resource": "deployments"
    },
    "user": "sanjary",
    "group": []
  }
}
```

Example response:

```
Allow:
{
  "apiVersion": "authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "status": {
    "allowed": true
  }
   "user": "sanjary",
    "group": [
      "my-project:admin"
    ]
}
Deny:
{
  "apiVersion": "authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "status": {
    "allowed": false,
    "reason": "user sanjary does not have read access
to the namespace"
  }
}
```

Auth Webhook

Example request:

```
{
  "apiVersion":
"authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "spec": {
    "resourceAttributes": {
      "namespace": "my-project",
      "verb": "get",
      "group": "apps/v1",
      "resource": "deployments"
    },
    "user": "sanjary",
    "group": []
  }
}
```

Example response:

```
Allow:
{
  "apiVersion": "authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "status": {
    "allowed": true
  }
   "user": "sanjary",
    "group": [
      "my-project:admin"
    ]
}
Deny:
{
  "apiVersion": "authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "status": {
    "allowed": false,
    "reason": "user sanjary does not have read access
to the namespace"
  }
}
```

Auth Webhook

Example request:

```
{
  "apiVersion":
"authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "spec": {
    "resourceAttributes": {
      "namespace": "my-project",
      "verb": "get",
      "group": "apps/v1",
      "resource": "deployments"
    },
    "user": "sanjary",
    "group": []
  }
}
```

Example response:

Allow:

```
{
  "apiVersion": "authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "status": {
    "allowed": true
  }
   "user": "sanjary",
    "group": [
      "my-project:admin"
    ]
}
```

Deny:

```
{
  "apiVersion": "authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "status": {
    "allowed": false,
    "reason": "user sanjary does not have read access
to the namespace"
  }
}
```

Auth Webhook

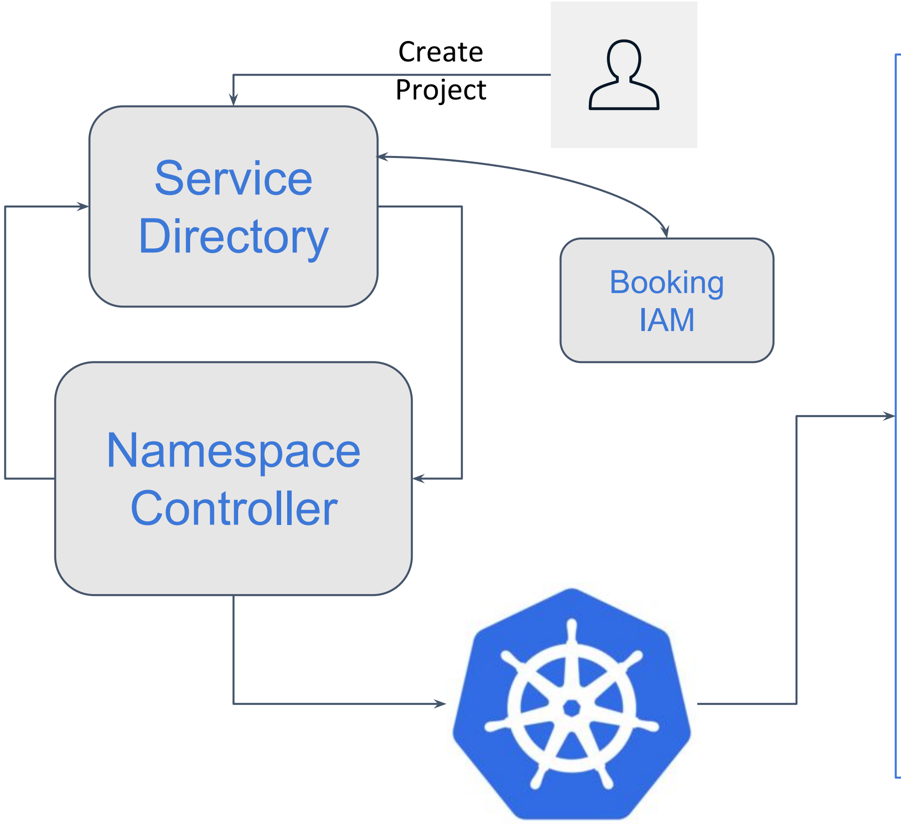
Example request:

```
{
  "apiVersion":
"authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "spec": {
    "resourceAttributes": {
      "namespace": "my-project",
      "verb": "get",
      "group": "apps/v1",
      "resource": "deployments"
    },
    "user": "sanjary",
    "group": []
  }
}
```

Example response:

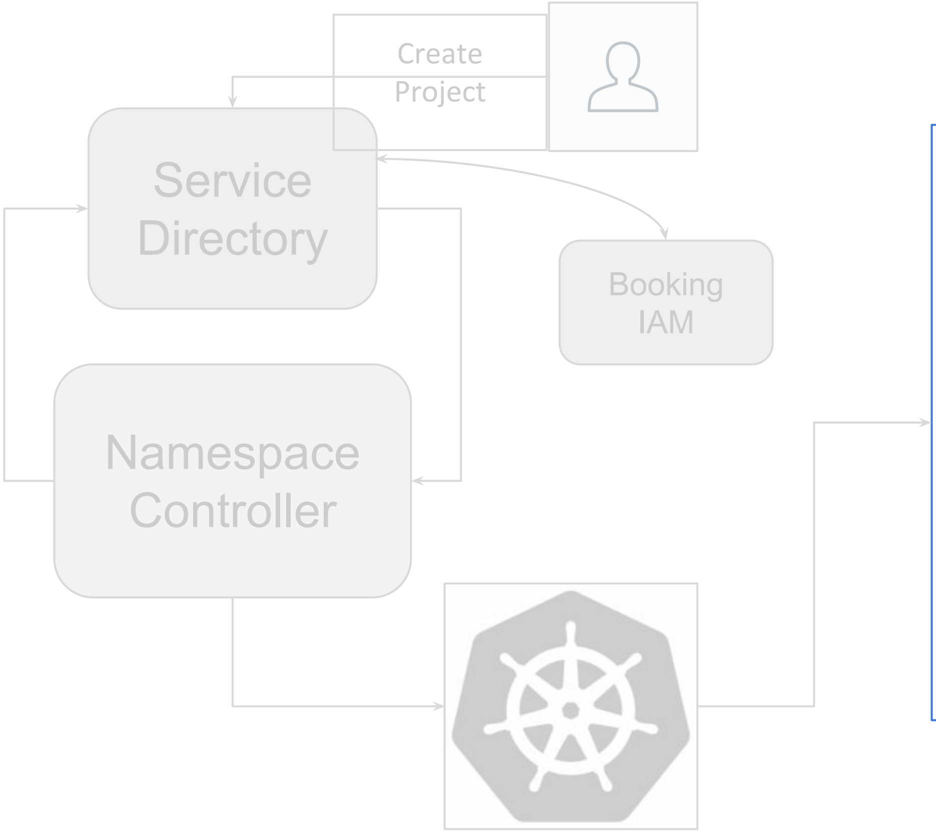
```
Allow:
{
  "apiVersion": "authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "status": {
    "allowed": true
  }
   "user": "sanjary",
    "group": [
      "my-project:admin"
    ]
}
Deny:
{
  "apiVersion": "authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "status": {
    "allowed": false,
    "reason": "user sanjary does not have read access
to the namespace"
  }
}
```

Workspace Provisioning



- Creates Namespace
- Creates Rolebinding
- Creates ResourceQuota
- Creates LimitRanges
- Creates Configmaps

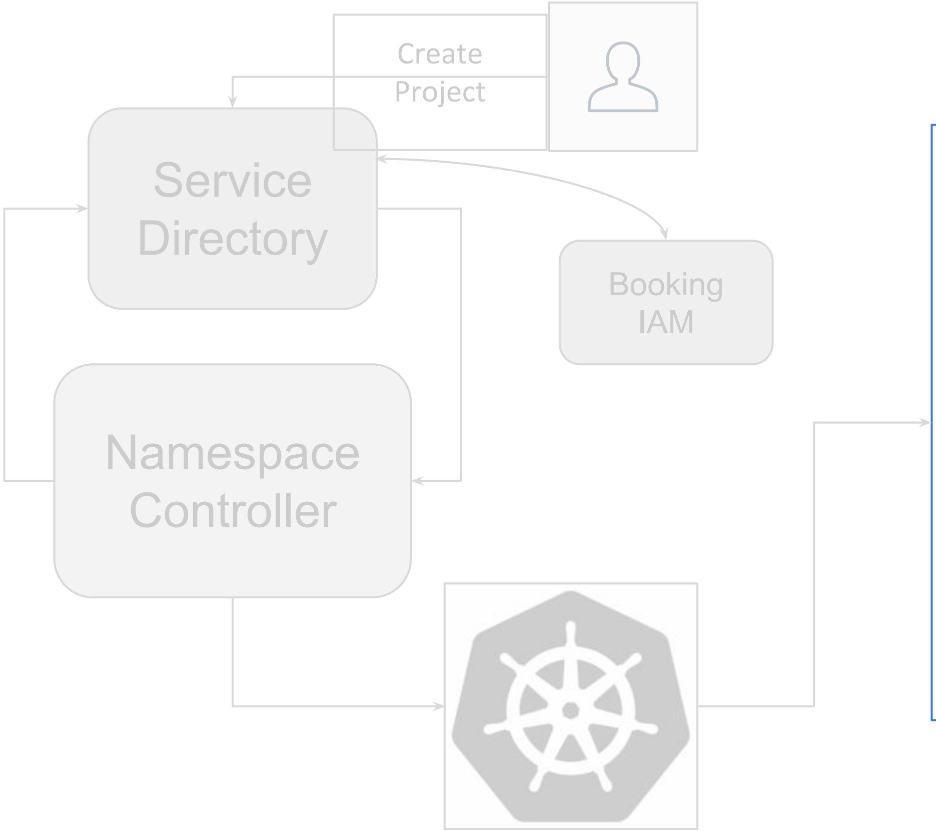
Auth Webhook + Namespace Controller



Rolebinding:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: admin
  namespace: my-project
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: admin
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: my-project:admin
```

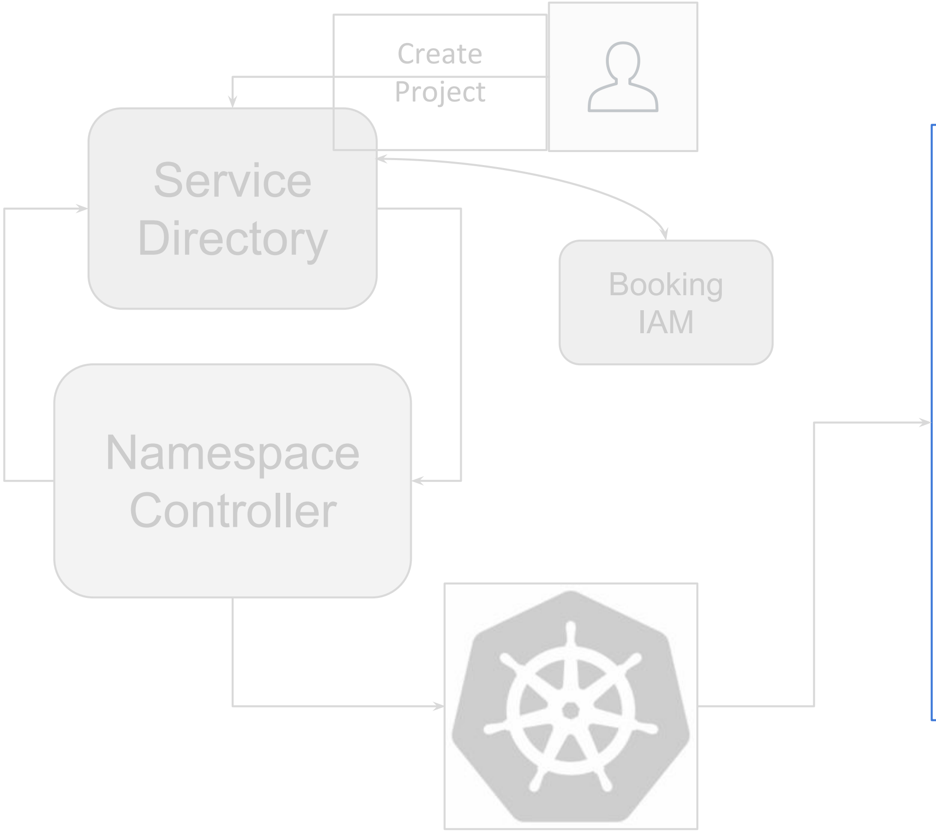
Auth Webhook + Namespace Controller



Rolebinding:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: admin
  namespace: my-project
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: admin
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: my-project:admin
```

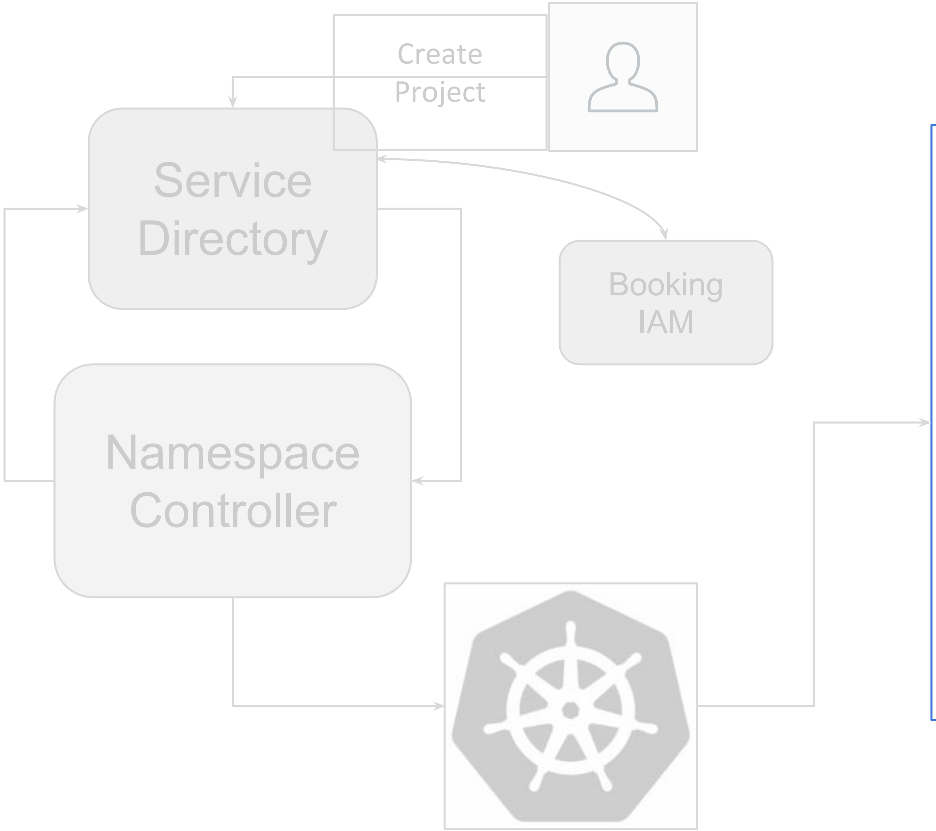
Auth Webhook + Namespace Controller



Rolebinding:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: admin
  namespace: my-project
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: admin
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: my-project:admin
```

Auth Webhook + Namespace Controller



Rolebinding:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: admin
  namespace: my-project
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: admin
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: my-project:admin
```

Auth Webhook

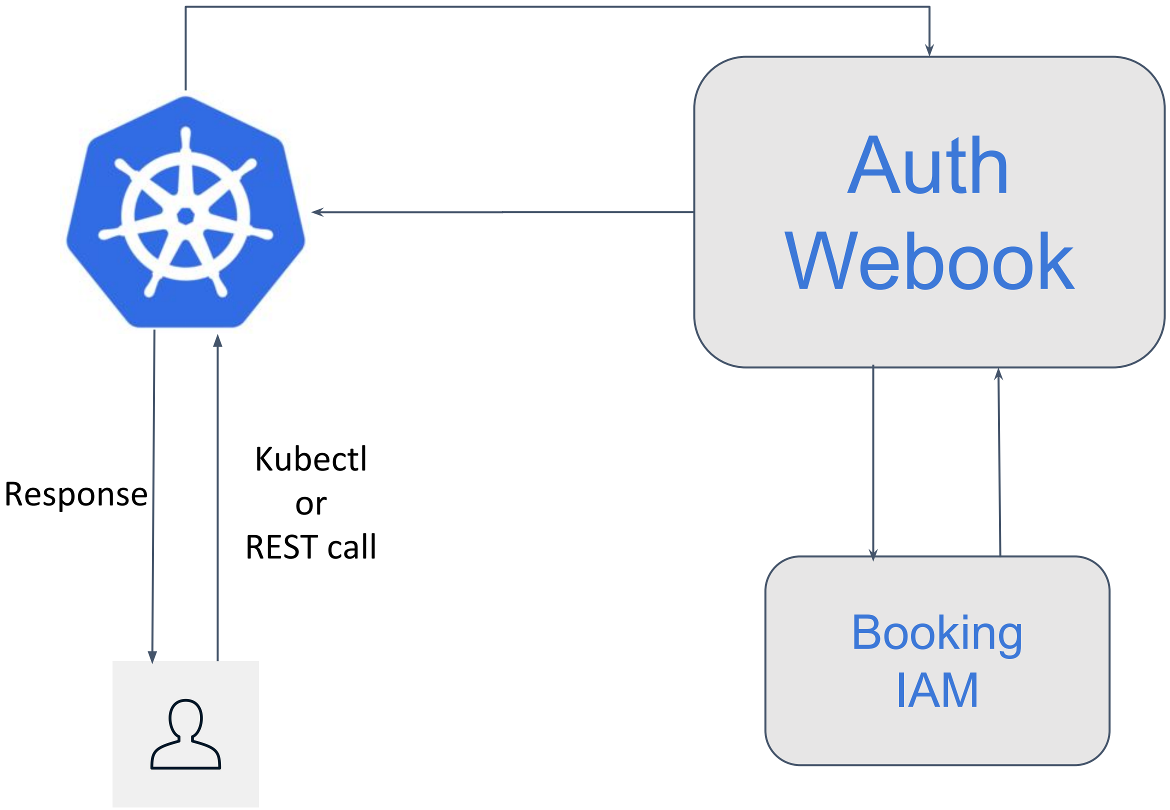
Example request:

```
{
  "apiVersion":
"authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "spec": {
    "resourceAttributes": {
      "namespace": "my-project",
      "verb": "get",
      "group": "apps/v1",
      "resource": "deployments"
    },
    "user": "sanjary",
    "group": []
  }
}
```

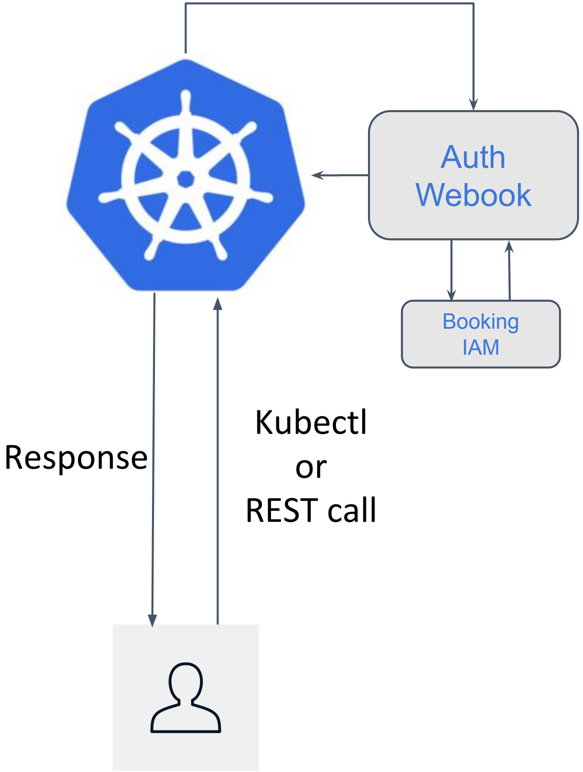
Example response:

```
Allow:
{
  "apiVersion": "authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "status": {
    "allowed": true
  }
   "user": "sanjary",
    "group": [
      "my-project:admin"
    ]
}
Deny:
{
  "apiVersion": "authorization.k8s.io/v1beta1",
  "kind": "SubjectAccessReview",
  "status": {
    "allowed": false,
    "reason": "user sanjary does not have read access
to the namespace"
  }
}
```

Lifecycle of kubectl command



Lifecycle of kubectl command



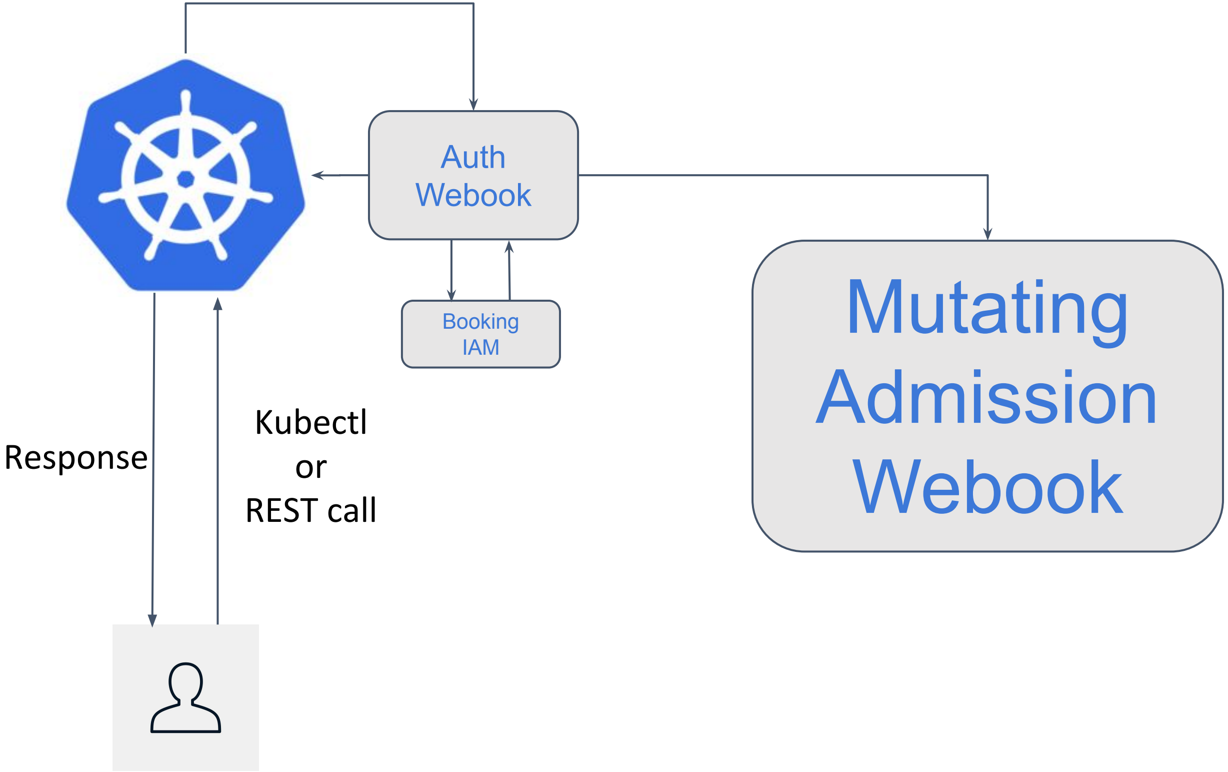
Auth Webook

Booking IAM

Response

Kubectl or REST call

Lifecycle of kubectl command



Auth Webook

Booking IAM

Mutating Admission Webook

Response

Kubectl or REST call

Admission Webhook (Mutation)

Example use cases:

Admission Webhook (Mutation)

Example use cases:

- Assign random non-root UID to pod

Admission Webhook (Mutation)

Example use cases:

- Assign random non-root UID to pod
- Inject environment variables in pod

Admission Webhook (Mutation)

Example use cases:

- Assign random non-root UID to pod
- Inject environment variables in pod
- Inject labels on pod

Admission Webhook (Mutation)

Example use cases:

- Assign random non-root UID to pod
- Inject environment variables in pod
- Inject labels on pod
- Inject init-containers/sidecars in pod

Admission Webhook (Mutation)

Configure kube-apiserver
with the webhook flag

--enable-admission-plugins=
MutatingAdmissionWebhook,..
.

Admission Webhook (Mutation)

Configure kube-apiserver with the webhook flag

--enable-admission-plugins=
MutatingAdmissionWebhook,..
.

```
apiVersion: admissionregistration.k8s.io/v1beta1
kind: MutatingWebhookConfiguration
metadata:
  name: pod-mutation
webhooks:
- clientConfig:
    caBundle: <CA bundle>
    service: null
    Url: https://mutate.example.com/v1/mutate-pods
  failurePolicy: Fail
  name: pod-mutation
  rules:
  - apiGroups:
    - '*'
    apiVersions:
    - '*'
    operations:
    - CREATE
    resources:
    - pods
```

Admission Webhook (Mutation)

Configure kube-apiserver with the webhook flag

--enable-admission-plugins=
MutatingAdmissionWebhook,...
.

```
apiVersion: admissionregistration.k8s.io/v1beta1
kind: MutatingWebhookConfiguration
metadata:
  name: pod-mutation
webhooks:
- clientConfig:
    caBundle: <CA bundle>
    service: null
    Url: https://mutate.example.com/v1/mutate-pods
  failurePolicy: Fail
  name: pod-mutation
  rules:
  - apiGroups:
    - '*'
    apiVersions:
    - '*'
    operations:
    - CREATE
    resources:
    - pods
```

Admission Webhook (Mutation)

Configure kube-apiserver with the webhook flag

--enable-admission-plugins=
MutatingAdmissionWebhook,..
.

```
apiVersion: admissionregistration.k8s.io/v1beta1
kind: MutatingWebhookConfiguration
metadata:
  name: pod-mutation
webhooks:
- clientConfig:
    caBundle: <CA bundle>
    service: null
    Url: https://mutate.example.com/v1/mutate-pods
  failurePolicy: Fail
  name: pod-mutation
  rules:
  - apiGroups:
    - '*'
    apiVersions:
    - '*'
    operations:
    - CREATE
    resources:
    - pods
```

Admission Webhook (Mutation)

Configure kube-apiserver with the webhook flag

--enable-admission-plugins=
MutatingAdmissionWebhook,..
.

```
apiVersion: admissionregistration.k8s.io/v1beta1
kind: MutatingWebhookConfiguration
metadata:
  name: pod-mutation
webhooks:
- clientConfig:
    caBundle: <CA bundle>
    service: null
    Url: https://mutate.example.com/v1/mutate-pods
  failurePolicy: Fail
  name: pod-mutation
  rules:
  - apiGroups:
    - '*'
    apiVersions:
    - '*'
    operations:
    - CREATE
    resources:
    - pods
```

Admission Webhook (Mutation)

Configure kube-apiserver with the webhook flag

--enable-admission-plugins=
MutatingAdmissionWebhook,..
.

```
apiVersion: admissionregistration.k8s.io/v1beta1
kind: MutatingWebhookConfiguration
metadata:
  name: pod-mutation
webhooks:
- clientConfig:
    caBundle: <CA bundle>
    service: null
    Url: https://mutate.example.com/v1/mutate-pods
  failurePolicy: Fail
  name: pod-mutation
  rules:
  - apiGroups:
    - '*'
    apiVersions:
    - '*'
    operations:
    - CREATE
    resources:
    - pods
```

Admission Webhook (Mutation)

Configure kube-apiserver with the webhook flag

--enable-admission-plugins= MutatingAdmissionWebhook,...
.

```
apiVersion: admissionregistration.k8s.io/v1beta1
kind: MutatingWebhookConfiguration
metadata:
  name: pod-mutation
webhooks:
- clientConfig:
    caBundle: <CA bundle>
    service: null
    Url: https://mutate.example.com/v1/mutate-pods
  failurePolicy: Fail
  name: pod-mutation
  rules:
  - apiGroups:
    - '*'
    apiVersions:
    - '*'
    operations:
    - CREATE
    resources:
    - pods
```

Admission Webhook (Mutation)

Configure kube-apiserver with the webhook flag

--enable-admission-plugins=
MutatingAdmissionWebhook,..
.

```
apiVersion: admissionregistration.k8s.io/v1beta1
kind: MutatingWebhookConfiguration
metadata:
  name: pod-mutation
webhooks:
- clientConfig:
    caBundle: <CA bundle>
    service: null
    Url: https://mutate.example.com/v1/mutate-pods
  failurePolicy: Fail
  name: pod-mutation
  rules:
  - apiGroups:
    - '*'
    apiVersions:
    - '*'
    operations:
    - CREATE
    resources:
    - pods
```

Admission Webhook (Mutation)

Configure kube-apiserver with the webhook flag

--enable-admission-plugins=
MutatingAdmissionWebhook,...
.

```
apiVersion: admissionregistration.k8s.io/v1beta1
kind: MutatingWebhookConfiguration
metadata:
  name: pod-mutation
webhooks:
- clientConfig:
    caBundle: <CA bundle>
    service: null
    Url: https://mutate.example.com/v1/mutate-pods
  failurePolicy: Fail
  name: pod-mutation
  rules:
  - apiGroups:
    - '*'
    apiVersions:
    - '*'
    operations:
    - CREATE
    resources:
    - pods
```

Admission Webhook (Mutation)

Configure kube-apiserver with the webhook flag

--enable-admission-plugins= MutatingAdmissionWebhook,..
.

```
apiVersion: admissionregistration.k8s.io/v1beta1
kind: MutatingWebhookConfiguration
metadata:
  name: pod-mutation
webhooks:
- clientConfig:
    caBundle: <CA bundle>
    service: null
    Url: https://mutate.example.com/v1/mutate-pods
  failurePolicy: Fail
  name: pod-mutation
  rules:
  - apiGroups:
    - '*'
    apiVersions:
    - '*'
    operations:
    - CREATE
    resources:
    - pods
```

Admission Webhook (Mutation)

Configure kube-apiserver with the webhook flag

--enable-admission-plugins= MutatingAdmissionWebhook,...
.

```
apiVersion: admissionregistration.k8s.io/v1beta1
kind: MutatingWebhookConfiguration
metadata:
  name: pod-mutation
webhooks:
- clientConfig:
    caBundle: <CA bundle>
    service: null
    Url: https://mutate.example.com/v1/mutate-pods
  failurePolicy: Fail
  name: pod-mutation
  rules:
  - apiGroups:
    - '*'
    apiVersions:
    - '*'
    operations:
    - CREATE
    resources:
    - pods
```

Admission Webhook (Mutation)

Configure kube-apiserver with the webhook flag

--enable-admission-plugins= MutatingAdmissionWebhook,...

.

```
apiVersion: admissionregistration.k8s.io/v1beta1
kind: MutatingWebhookConfiguration
metadata:
  name: pod-mutation
webhooks:
- clientConfig:
    caBundle: <CA bundle>
    service: null
    Url: https://mutate.example.com/v1/mutate-pods
  failurePolicy: Fail
  name: pod-mutation
  rules:
  - apiGroups:
    - '*'
    apiVersions:
    - '*'
    operations:
    - CREATE
    resources:
    - pods
```

Admission Webhook (Mutation)

Configure kube-apiserver with the webhook flag

--enable-admission-plugins= MutatingAdmissionWebhook,...

.

```
apiVersion: admissionregistration.k8s.io/v1beta1
kind: MutatingWebhookConfiguration
metadata:
  name: pod-mutation
webhooks:
- clientConfig:
    caBundle: <CA bundle>
    service: null
    Url: https://mutate.example.com/v1/mutate-pods
  failurePolicy: Fail
  name: pod-mutation
  rules:
  - apiGroups:
    - '*'
    apiVersions:
    - '*'
    operations:
    - CREATE
    resources:
    - pods
```

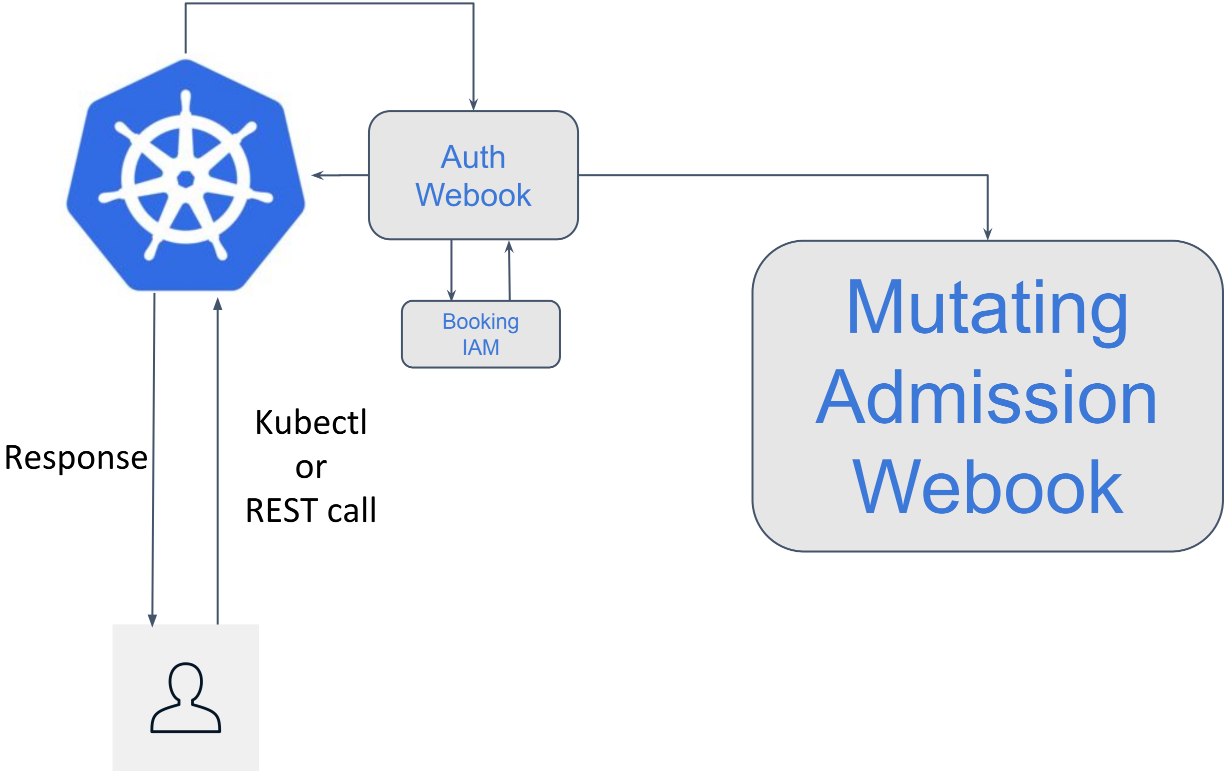
Admission Webhook (Mutation)

Configure kube-apiserver with the webhook flag

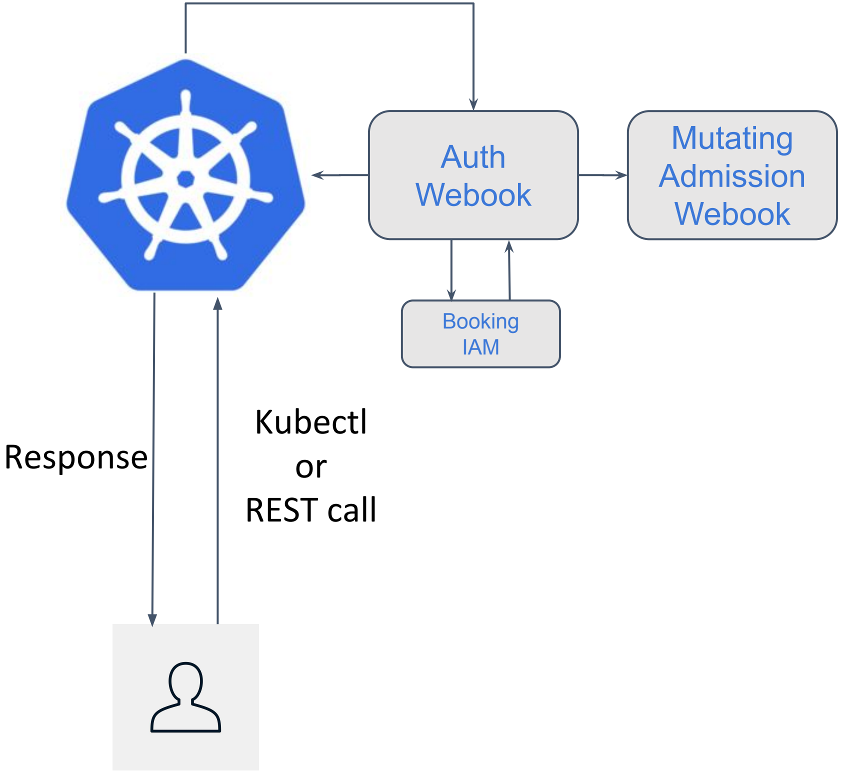
--enable-admission-plugins= MutatingAdmissionWebhook,...
.

```
apiVersion: admissionregistration.k8s.io/v1beta1
kind: MutatingWebhookConfiguration
metadata:
  name: pod-mutation
webhooks:
- clientConfig:
    caBundle: <CA bundle>
    service: null
    Url: https://mutate.example.com/v1/mutate-pods
  failurePolicy: Fail
  name: pod-mutation
  rules:
  - apiGroups:
    - '*'
    apiVersions:
    - '*'
    operations:
    - CREATE
    resources:
    - pods
```

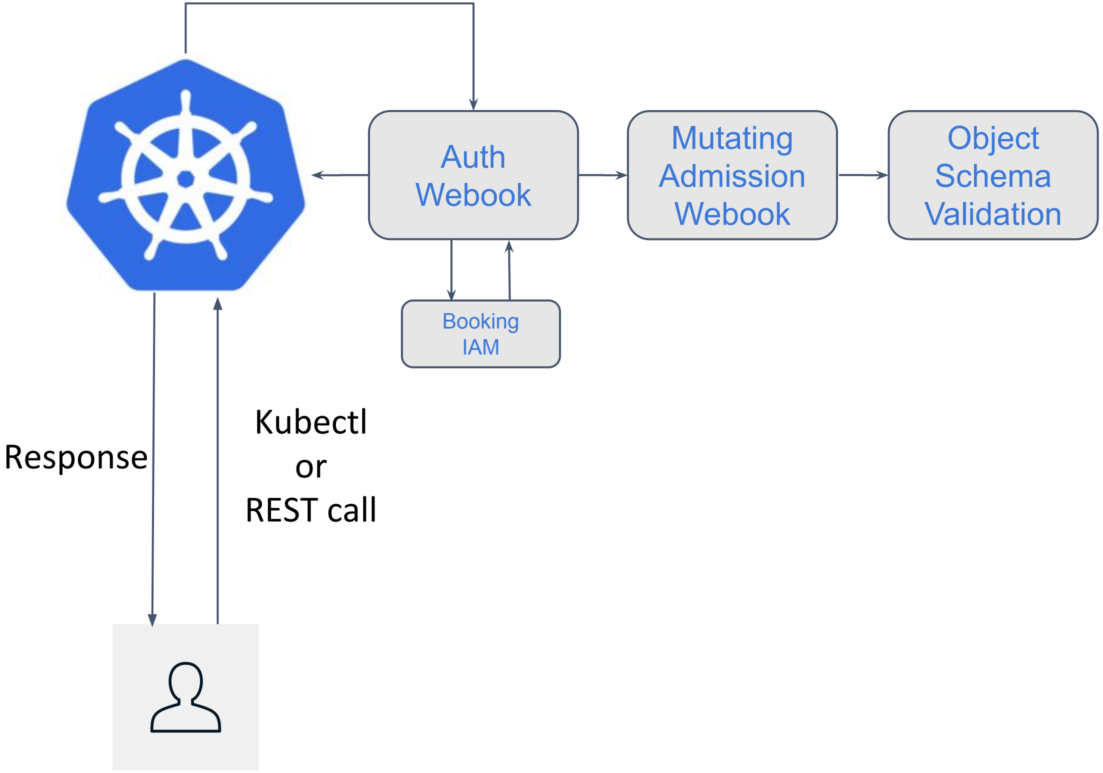
Lifecycle of kubectl command



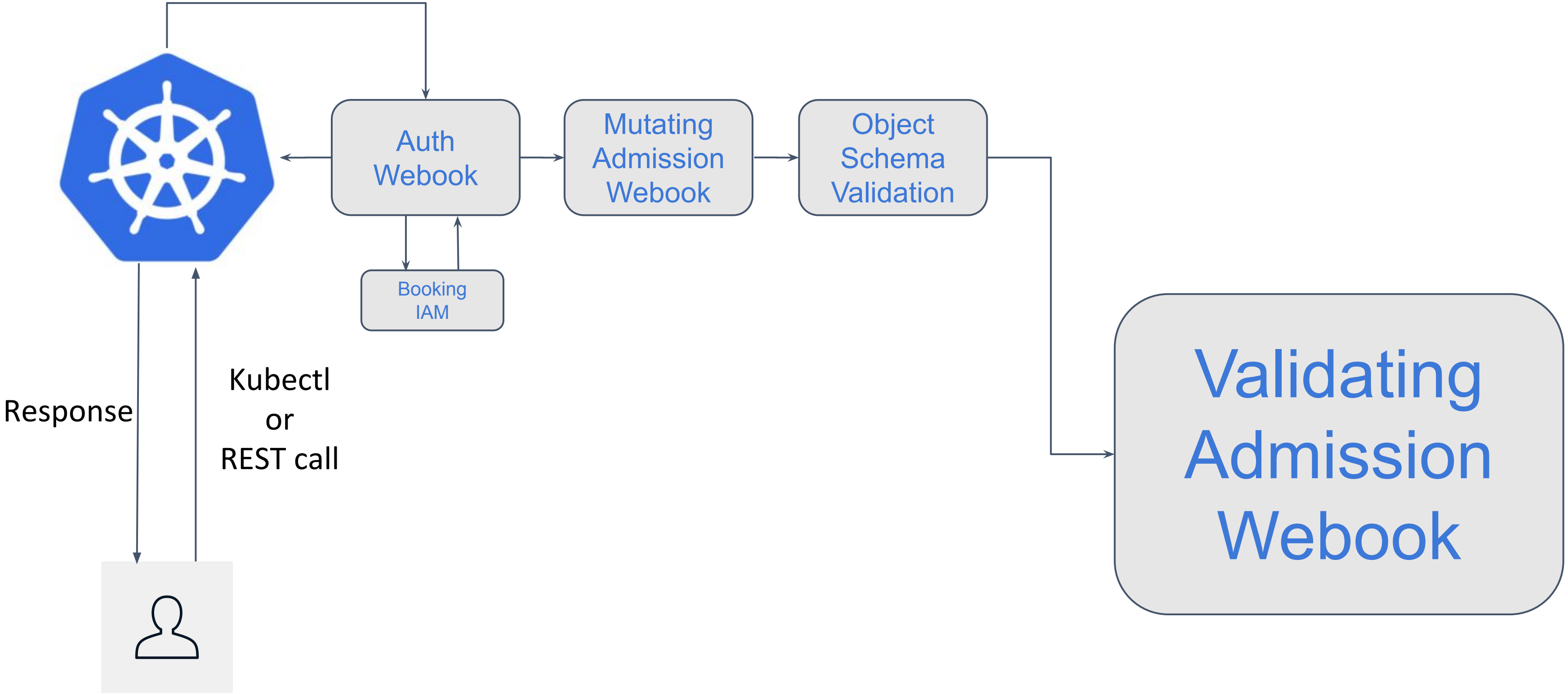
Lifecycle of kubectl command



Lifecycle of kubectl command



Lifecycle of kubectl command



Admission Webhook (Validation)

Example use cases:

Admission Webhook (Validation)

Example use cases:

- Check legitimacy (eg. registration in Service Directory in our case)

Admission Webhook (Validation)

Example use cases:

- Check legitimacy (eg. registration in Service Directory in our case)
- Ensure running images only from trusted sources

Admission Webhook (Validation)

Example use cases:

- Check legitimacy (eg. registration in Service Directory in our case)
- Ensure running images only from trusted sources
- Check number of containers in pod

Admission Webhook (Validation)

Example use cases:

- Check legitimacy (eg. registration in Service Directory in our case)
- Ensure running images only from trusted sources
- Check number of containers in pod
- Check presence of certain labels on pod

Admission Webhook (Validation)

Example use cases:

- Check legitimacy (eg. registration in Service Directory in our case)
- Ensure running images only from trusted sources
- Check number of containers in pod
- Check presence of certain labels on pod
- Enforce certain best practices for kubernetes resource declaration

Admission Webhook (Validation)

Configure kube-apiserver with the webhook flag

--enable-admission-plugins=
ValidatingAdmissionWebhook,
...

```
apiVersion: admissionregistration.k8s.io/v1beta1
kind: ValidatingWebhookConfiguration
metadata:
  name: pod-validation
webhooks:
- clientConfig:
    caBundle: <CA bundle>
    service: null
    Url: https://validate.example.com/v1/validate-pods
  failurePolicy: Fail
  name: pod-validation
  rules:
  - apiGroups:
    - '*'
    apiVersions:
    - '*'
    operations:
    - CREATE
    resources:
    - pods
```

Admission Webhook (Validation)

Configure kube-apiserver with the webhook flag

--enable-admission-plugins=
ValidatingAdmissionWebhook,
...

```
apiVersion: admissionregistration.k8s.io/v1beta1
kind: ValidatingWebhookConfiguration
metadata:
  name: pod-validation
webhooks:
- clientConfig:
    caBundle: <CA bundle>
    service: null
    Url: https://validate.example.com/v1/validate-pods
  failurePolicy: Fail
  name: pod-validation
  rules:
  - apiGroups:
    - '*'
    apiVersions:
    - '*'
    operations:
    - CREATE
    resources:
    - pods
```

Admission Webhook (Validation)

Configure kube-apiserver with the webhook flag

--enable-admission-plugins=
ValidatingAdmissionWebhook,
...

```
apiVersion: admissionregistration.k8s.io/v1beta1
kind: ValidatingWebhookConfiguration
metadata:
  name: pod-validation
webhooks:
- clientConfig:
    caBundle: <CA bundle>
    service: null
    Url: https://validate.example.com/v1/validate-pods
  failurePolicy: Fail
  name: pod-validation
  rules:
  - apiGroups:
    - '*'
    apiVersions:
    - '*'
    operations:
    - CREATE
    resources:
    - pods
```

Admission Webhook (Validation)

Configure kube-apiserver with the webhook flag

--enable-admission-plugins=
ValidatingAdmissionWebhook,
...

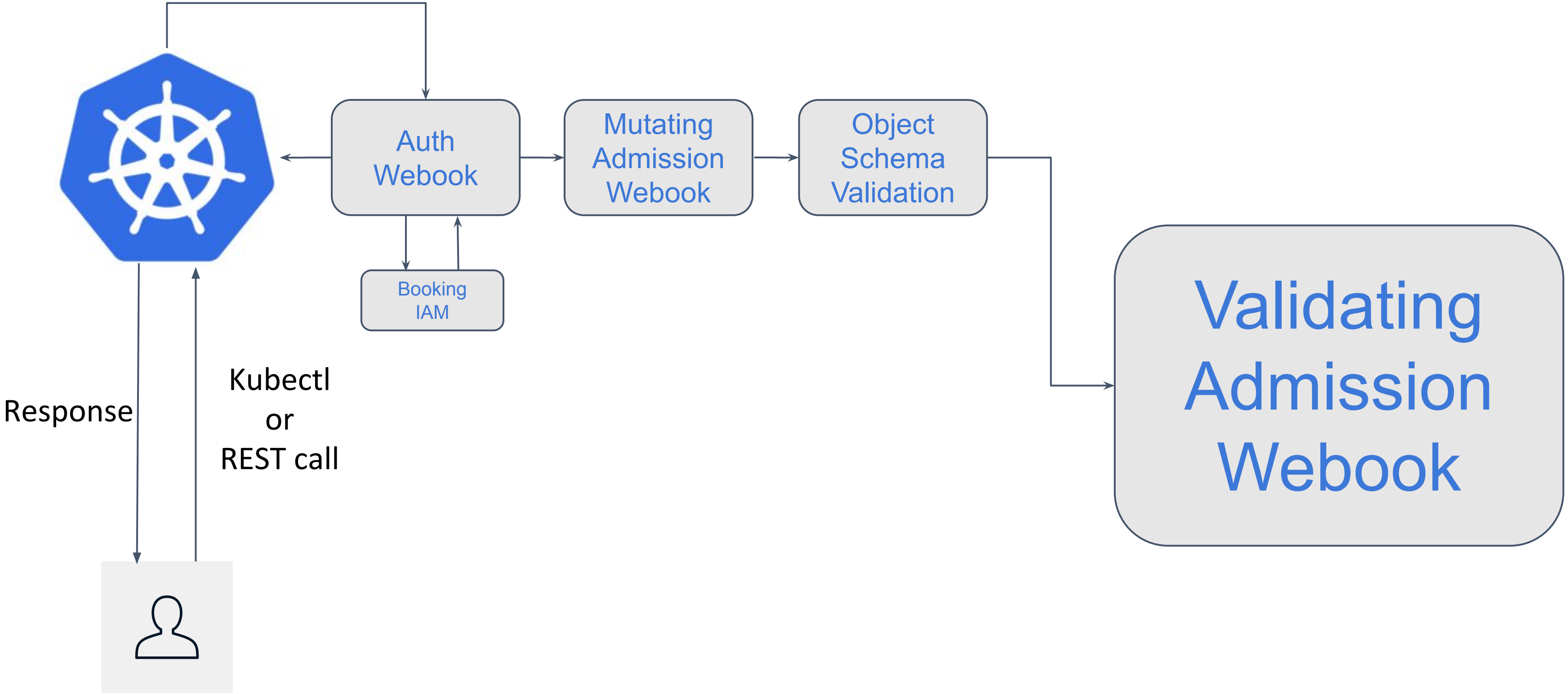
```
apiVersion: admissionregistration.k8s.io/v1beta1
kind: ValidatingWebhookConfiguration
metadata:
  name: pod-validation
webhooks:
- clientConfig:
    caBundle: <CA bundle>
    service: null
    Url: https://validate.example.com/v1/validate-pods
  failurePolicy: Fail
  name: pod-validation
  rules:
  - apiGroups:
    - '*'
    apiVersions:
    - '*'
    operations:
    - CREATE
    resources:
    - pods
```

Admission Webhook

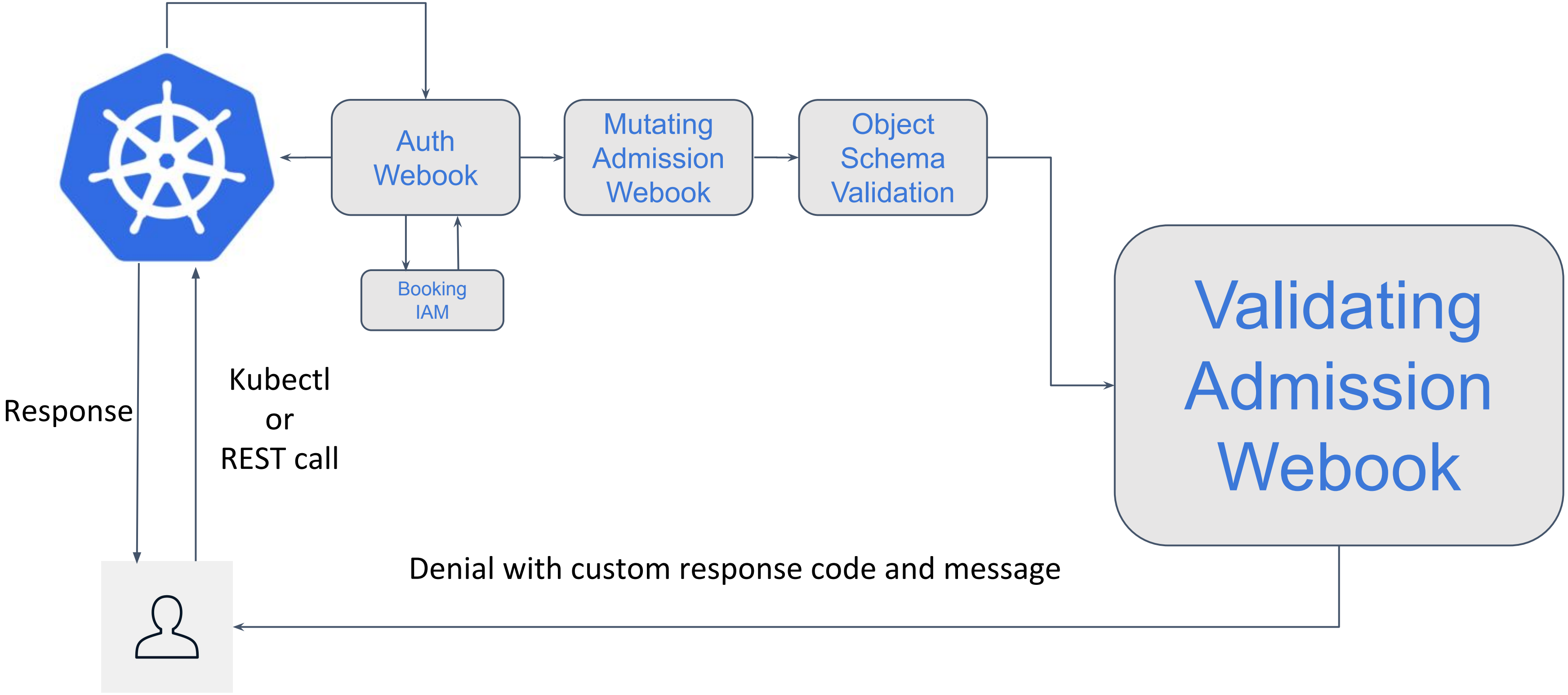
Full example implementation can be found here:

https://github.com/kubernetes/kubernetes/tree/v1.13.0/test/images/webhook

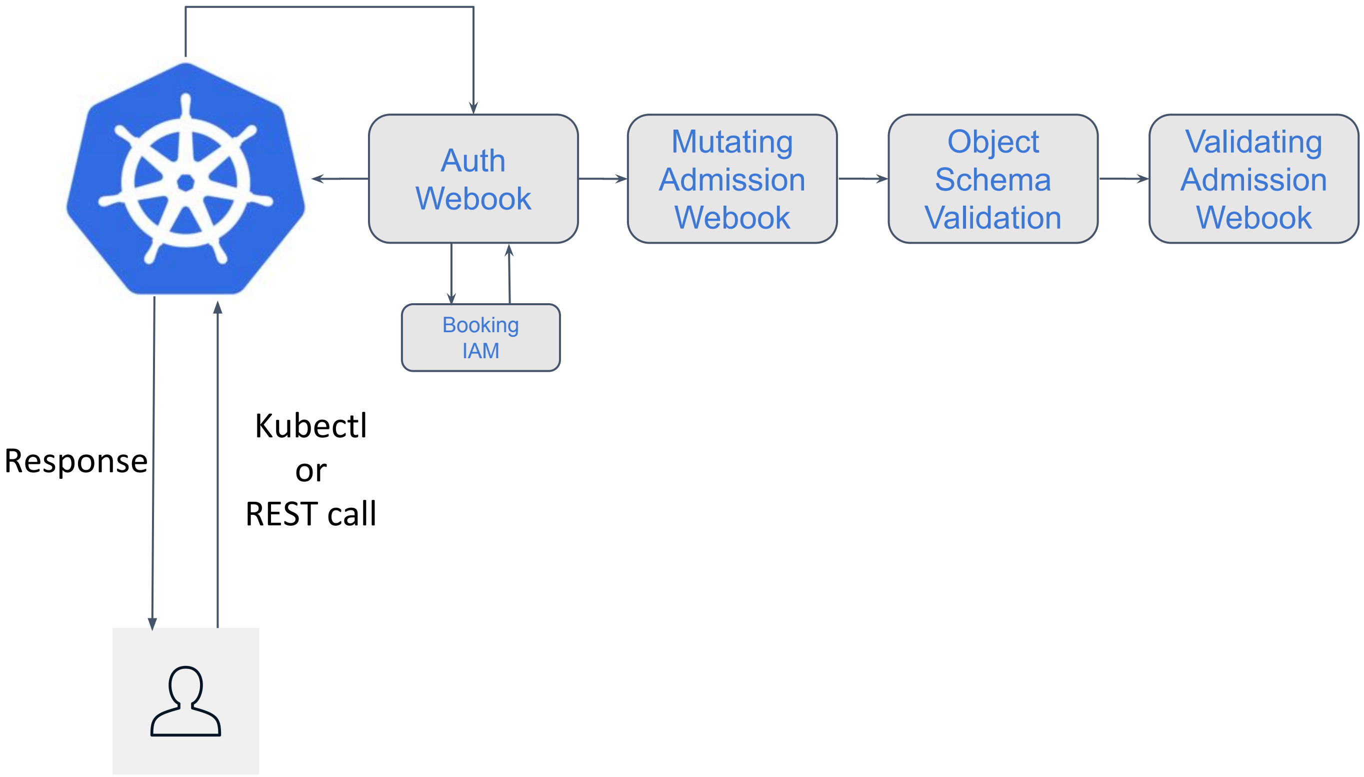
Lifecycle of kubectl command



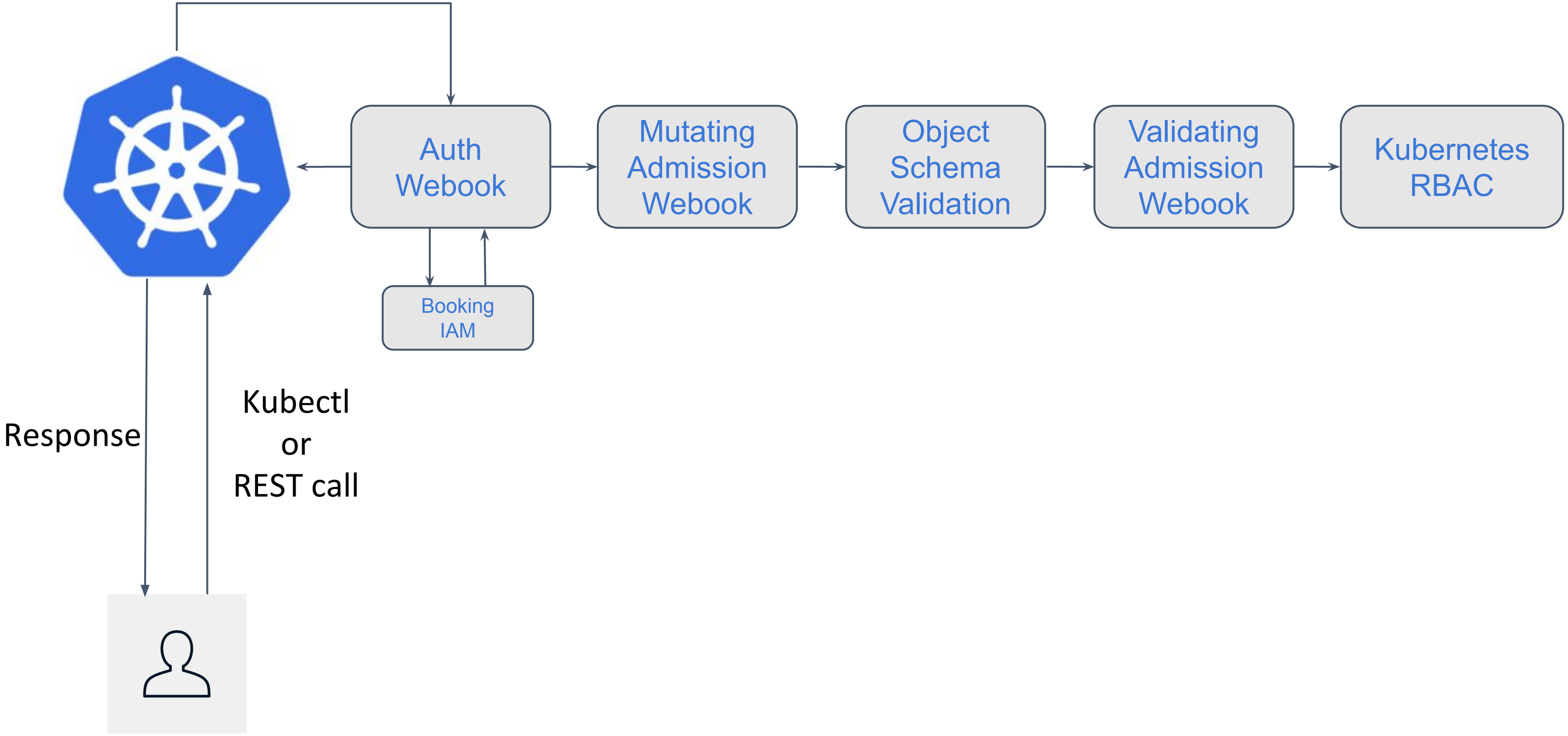
Lifecycle of kubectl command



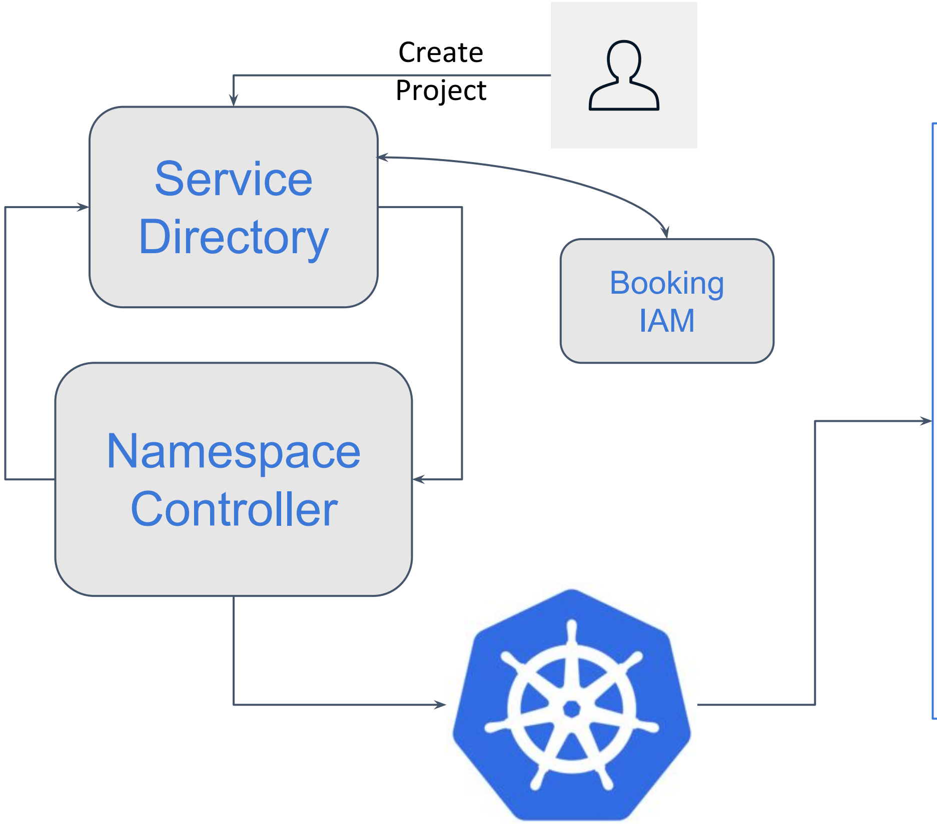
Lifecycle of kubectl command



Lifecycle of kubectl command



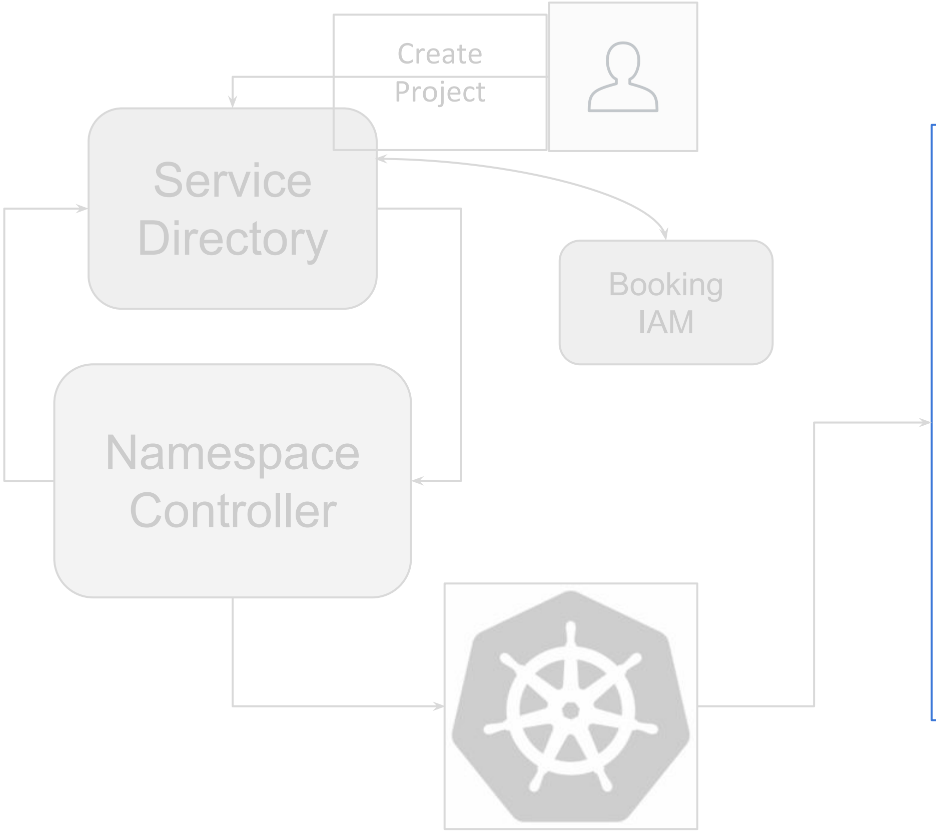
Auth Webhook + Namespace Controller



Rolebinding:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: admin
  namespace: my-project
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: admin
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: my-project:admin
```

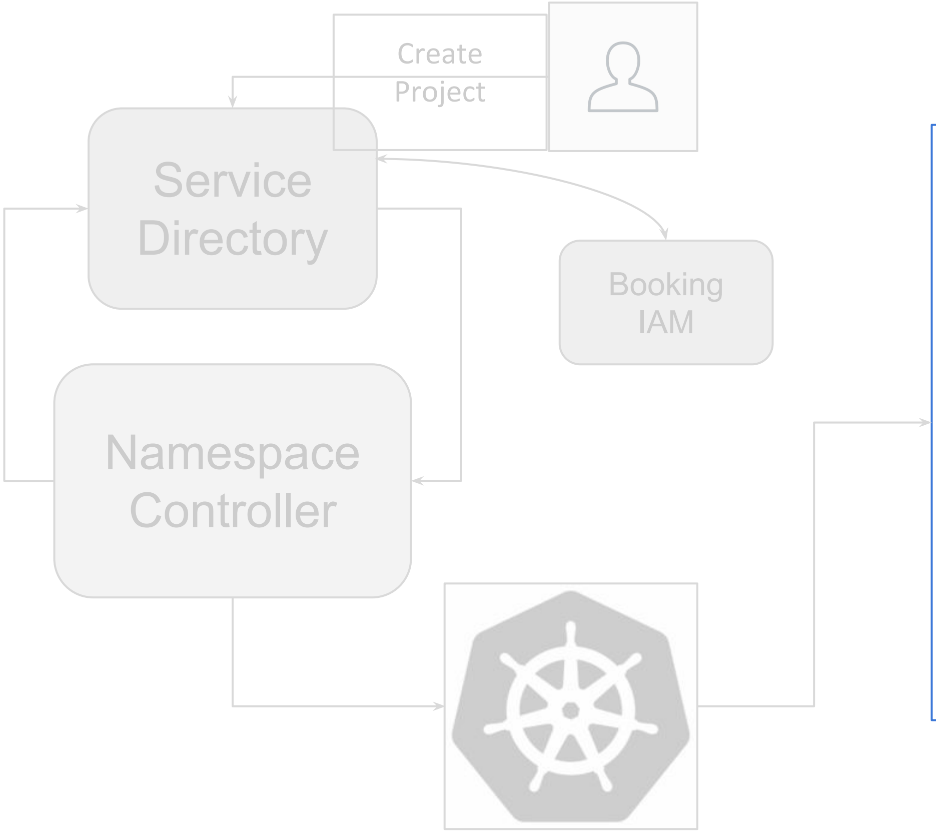
Auth Webhook + Namespace Controller



Rolebinding:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: admin
  namespace: my-project
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: admin
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: my-project:admin
```

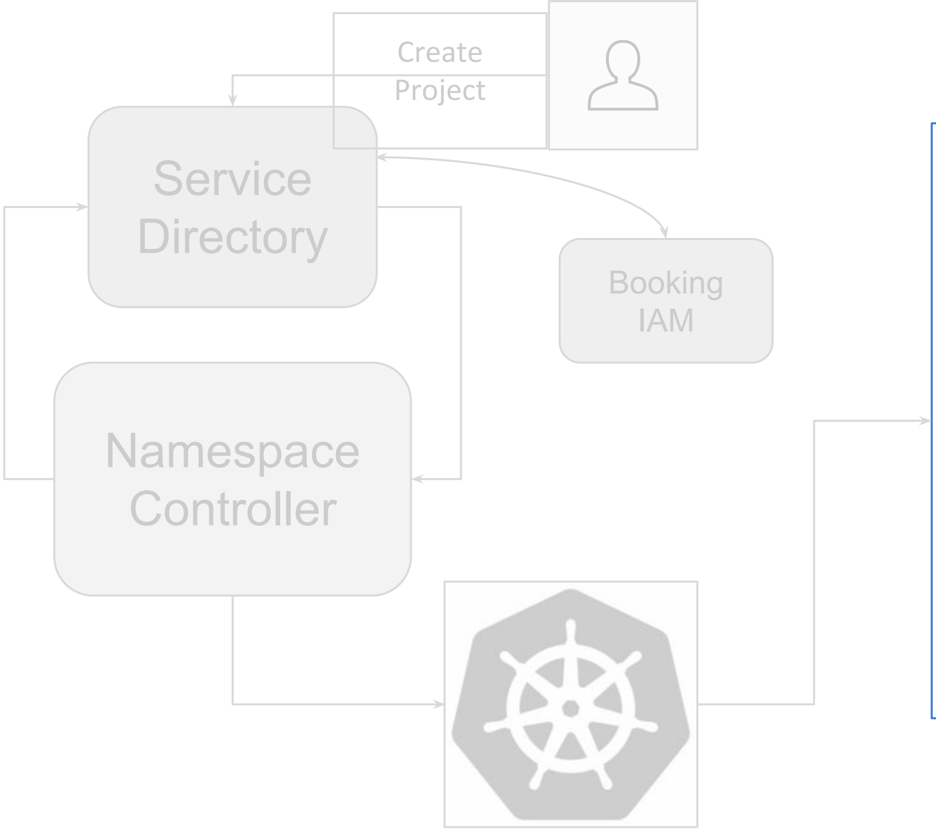
Auth Webhook + Namespace Controller



Rolebinding:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: admin
  namespace: my-project
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: admin
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: my-project:admin
```

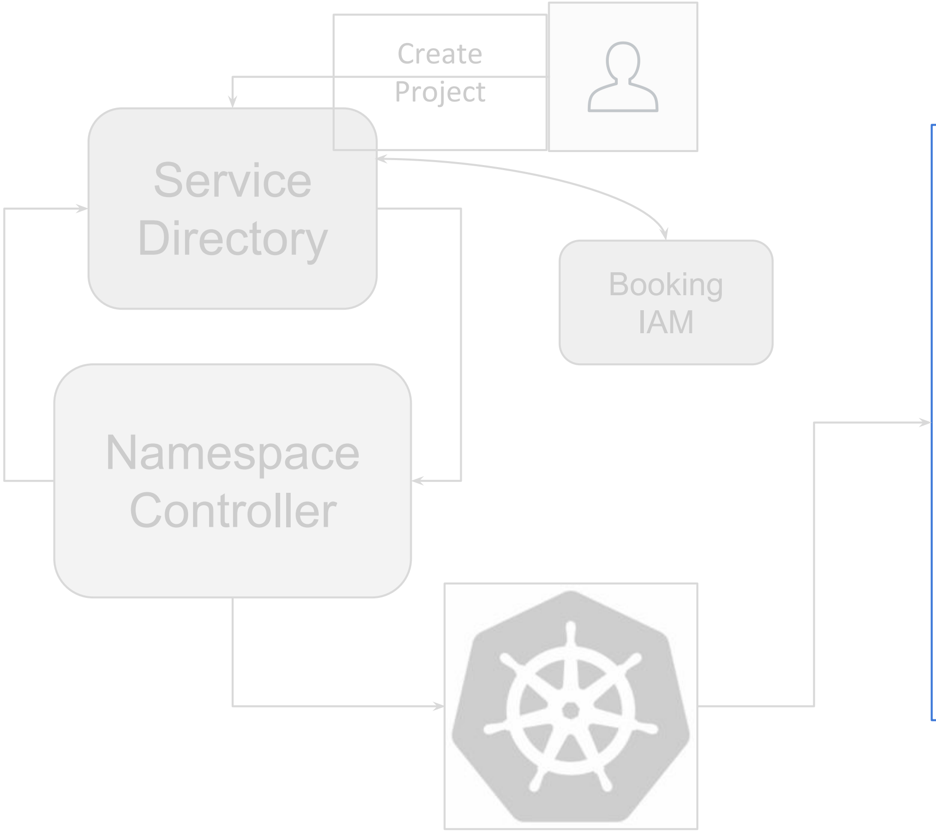
Auth Webhook + Namespace Controller



Rolebinding:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: admin
  namespace: my-project
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: admin
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: my-project:admin
```

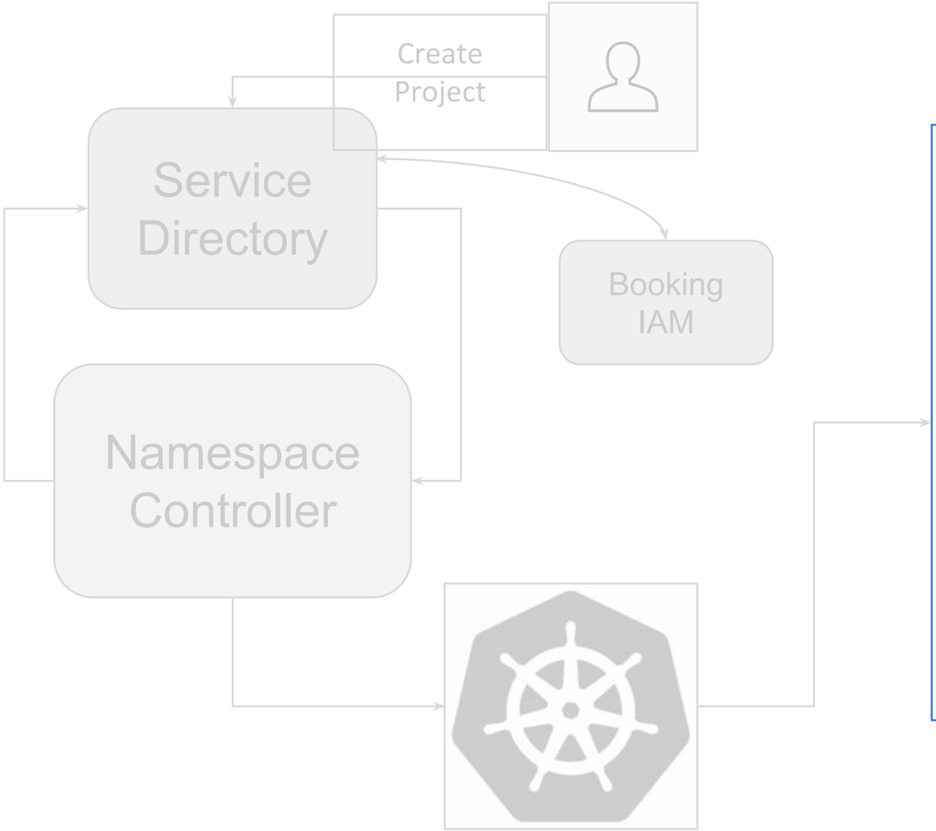
Auth Webhook + Namespace Controller



Rolebinding:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: admin
  namespace: my-project
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: admin
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: my-project:admin
```

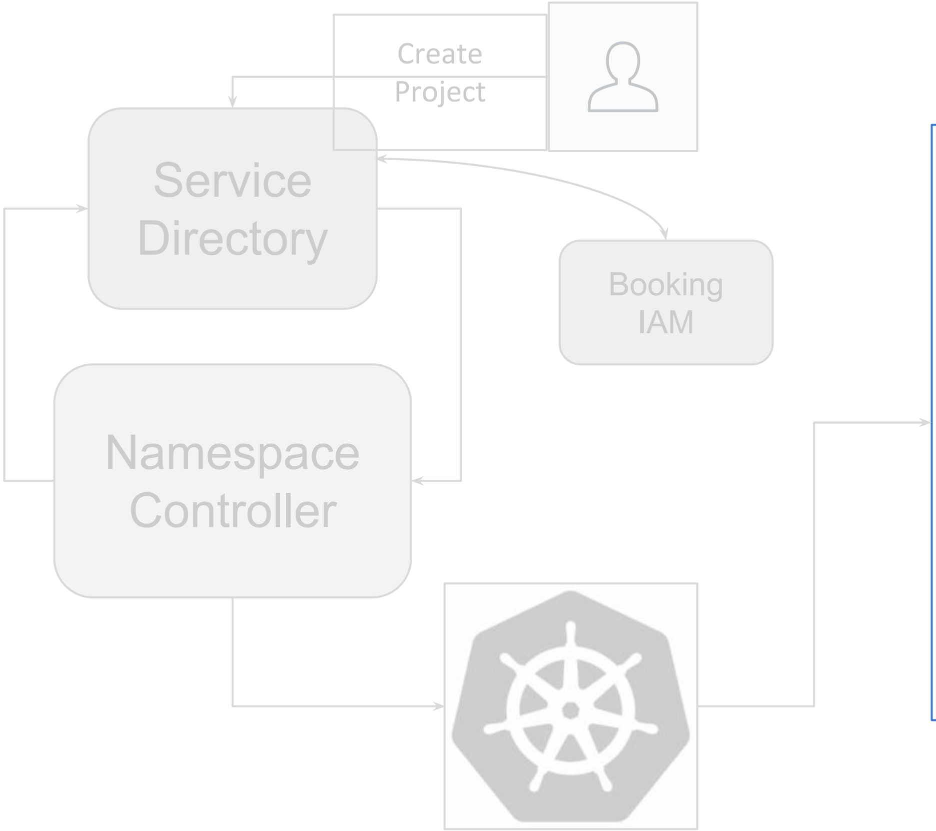
Auth Webhook + Namespace Controller



Rolebinding:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: admin
  namespace: my-project
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: admin
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: my-project:admin
```

Auth Webhook + Namespace Controller

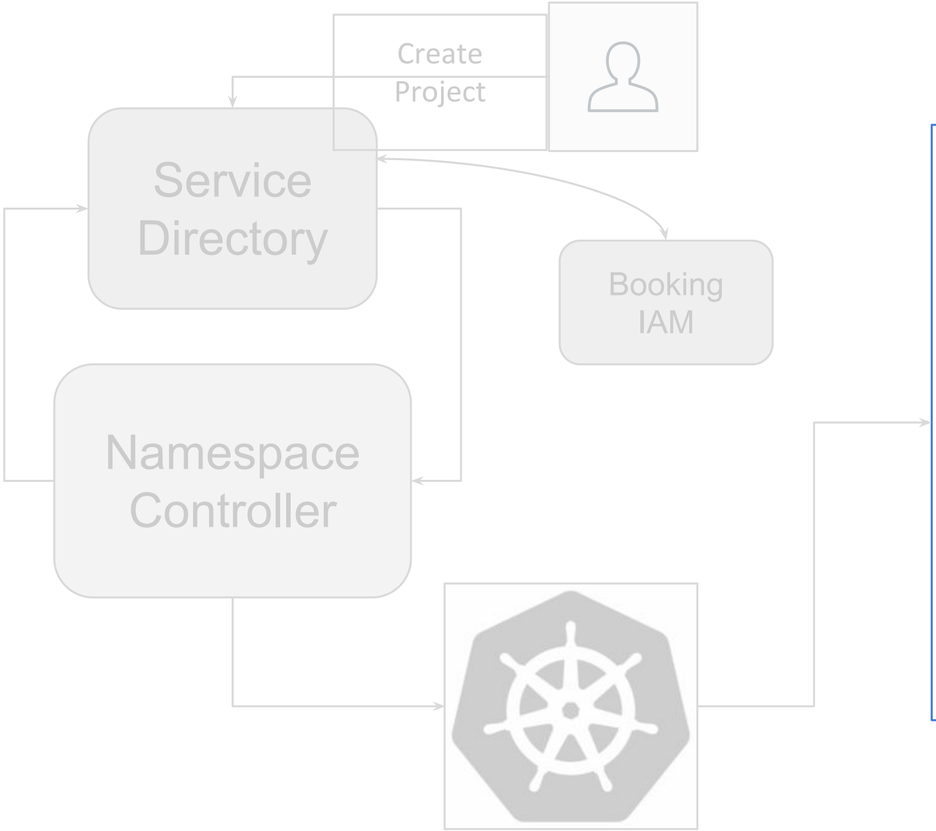


Rolebinding:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: admin
  namespace: my-project
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: admin
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: my-project:admin
```

Auth Webhook + Namespace Controller

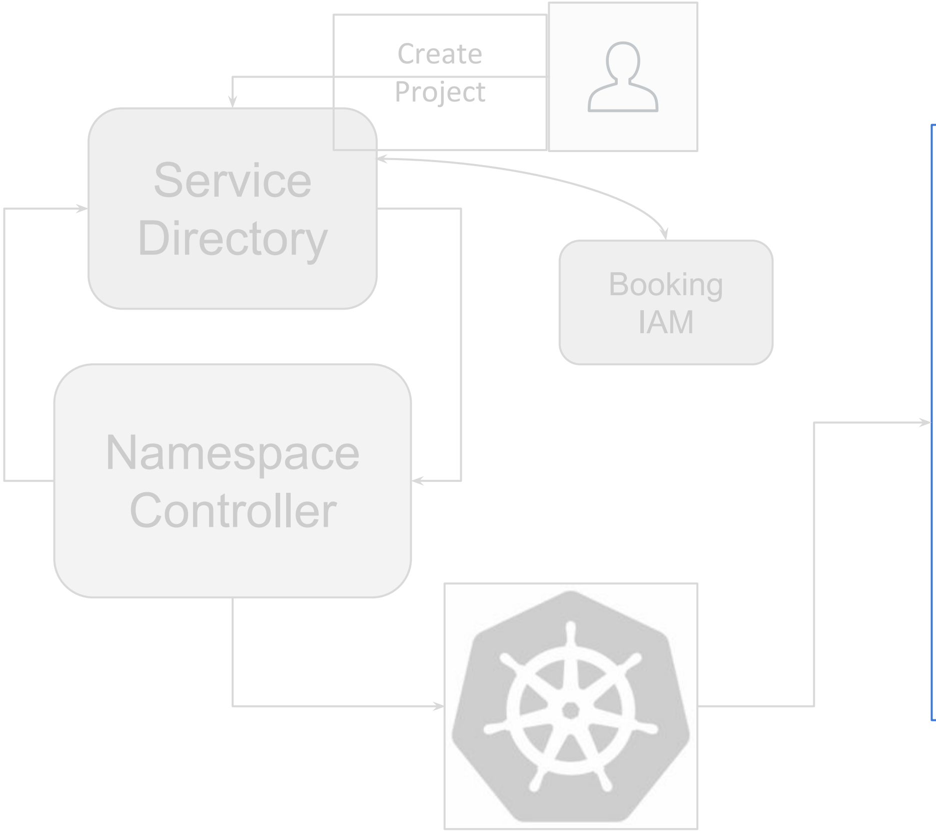
KubeCon CloudNativeCon Europe 2019



Rolebinding:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: admin
  namespace: my-project
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: admin
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: my-project:admin
```

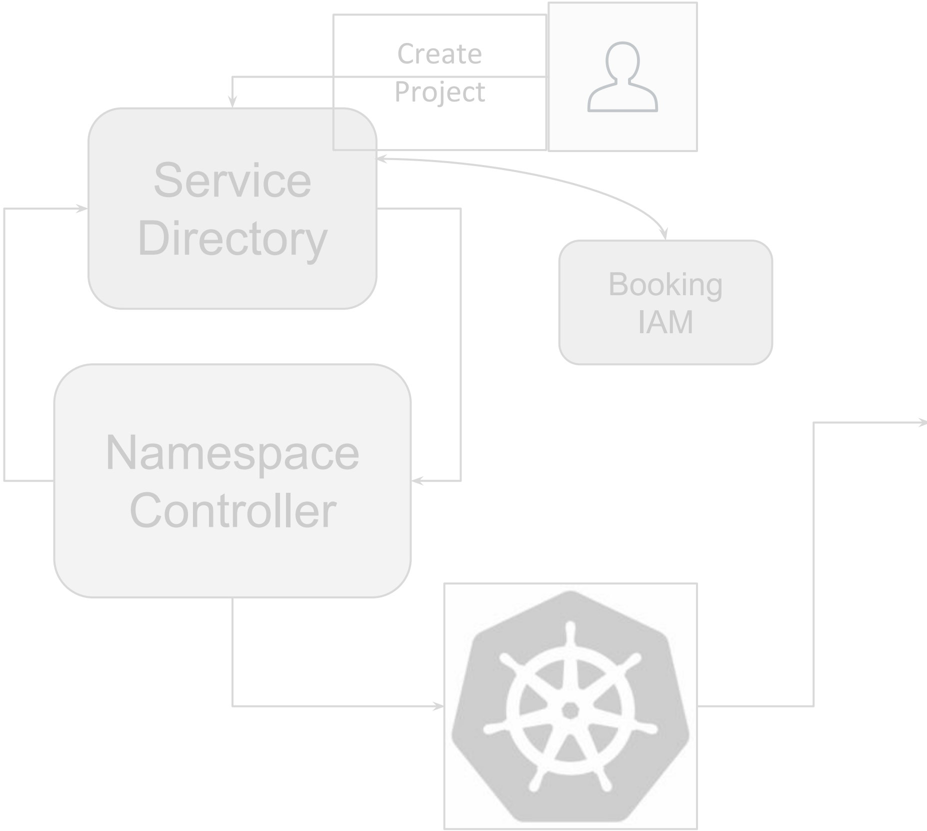
Auth Webhook + Namespace Controller



Rolebinding:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: admin
  namespace: my-project
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: admin
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: my-project:admin
```

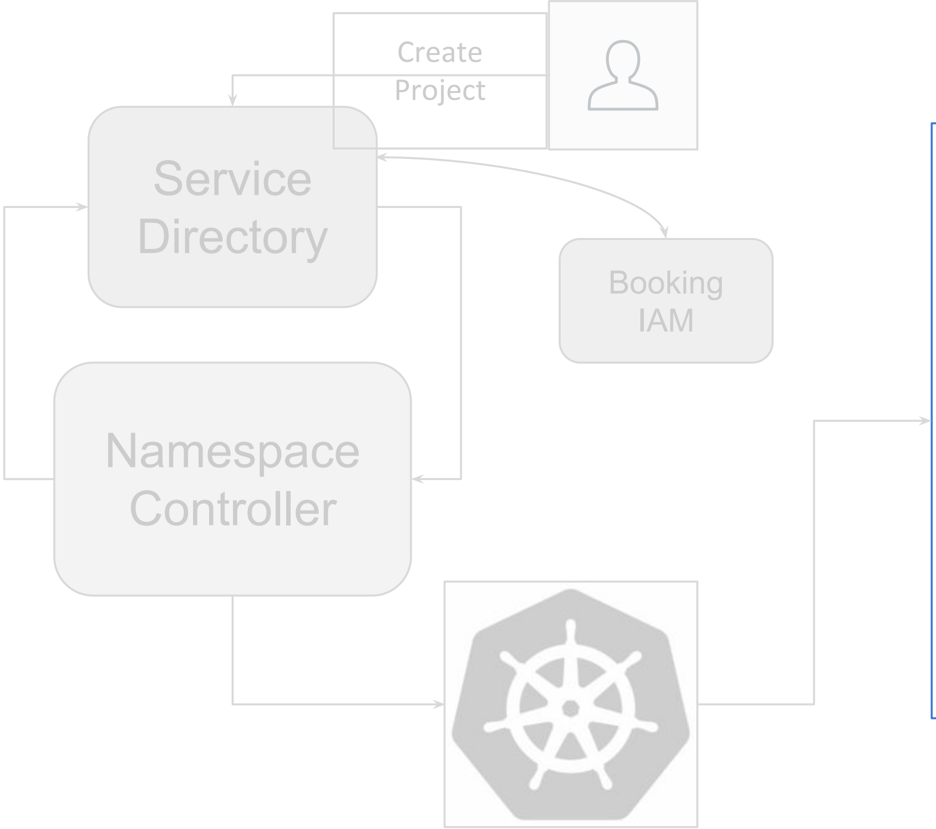
Auth Webhook + Namespace Controller



Rolebinding:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: admin
  namespace: my-project
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: admin
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: my-project:admin
```

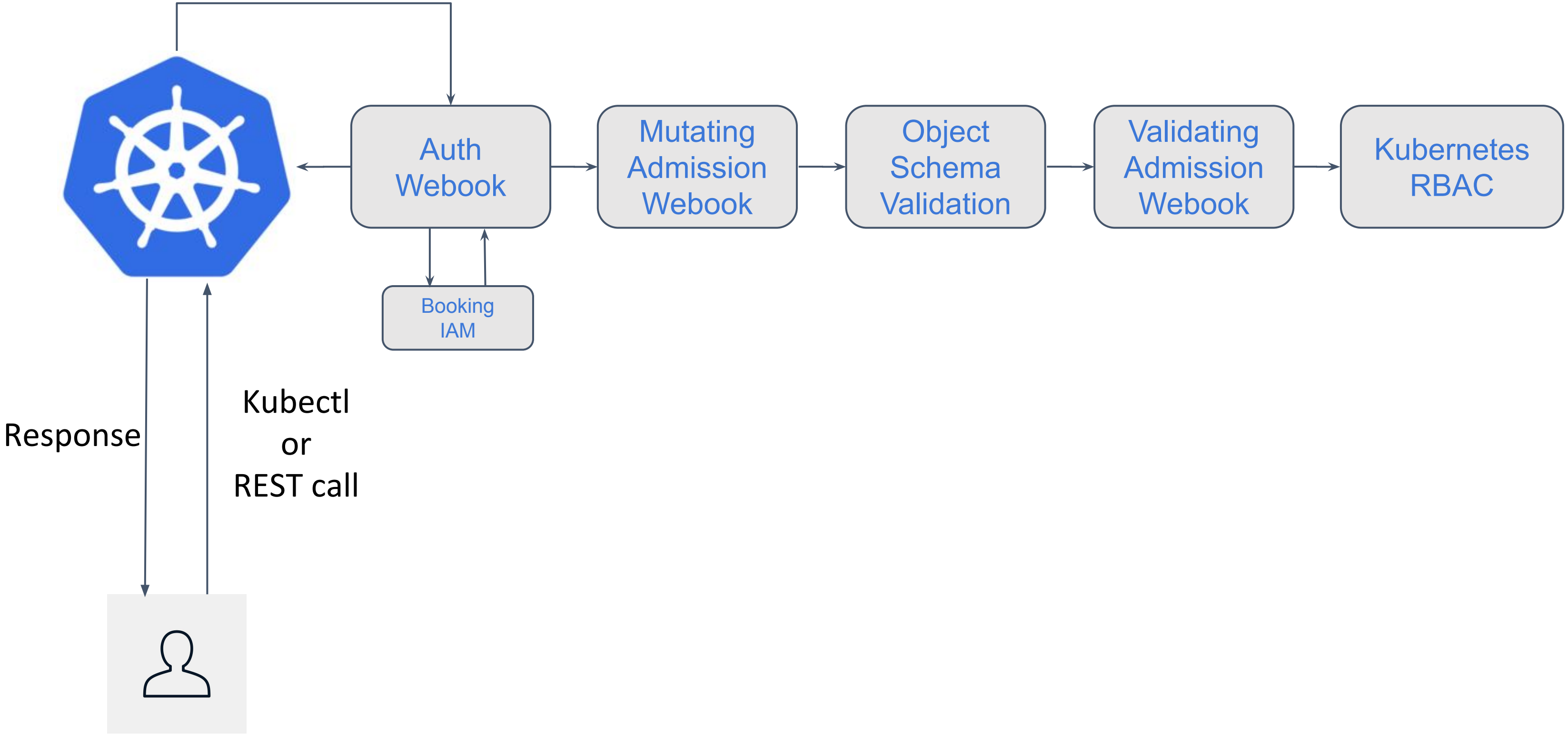
Auth Webhook + Namespace Controller



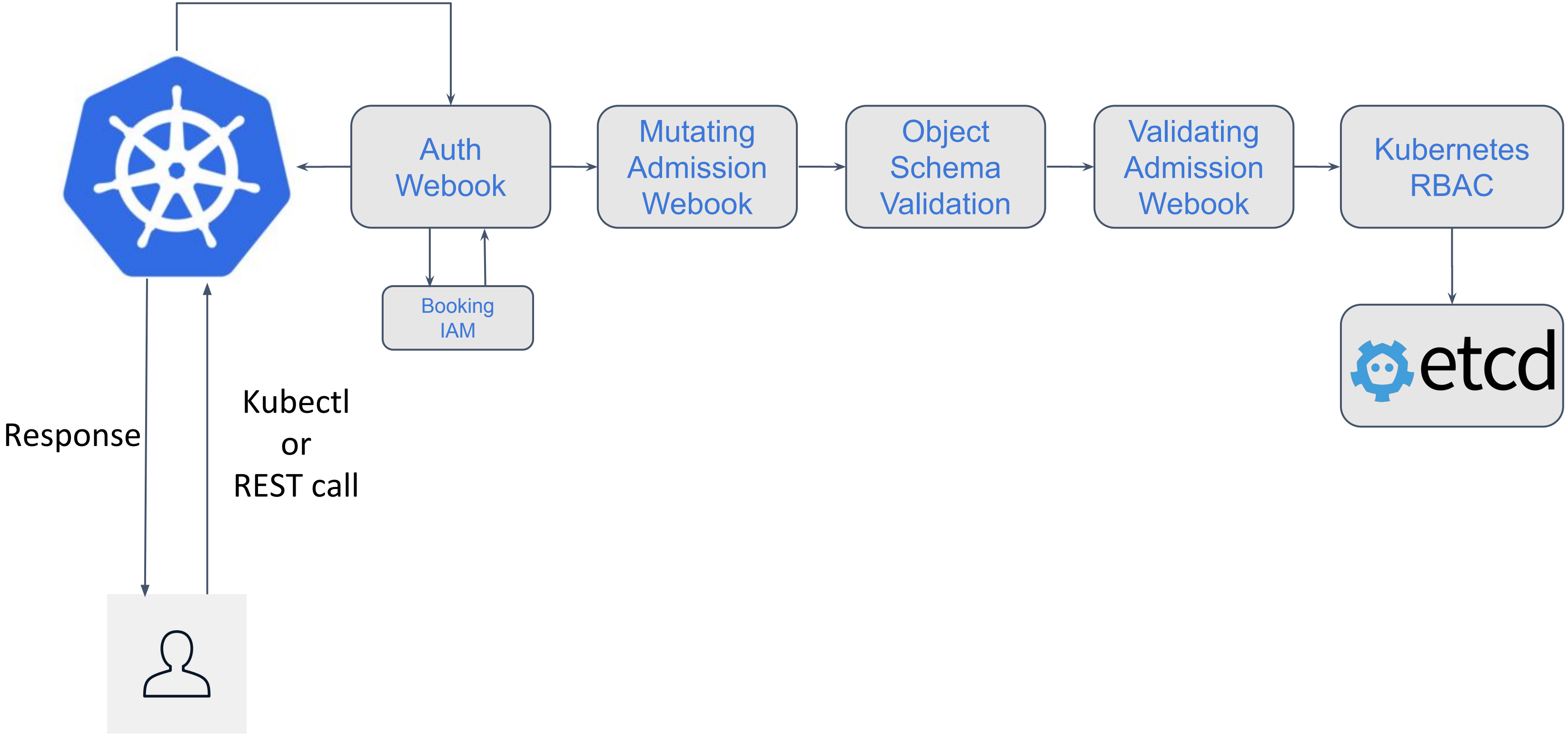
Rolebinding:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: admin
  namespace: my-project
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: admin
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: my-project:admin
```

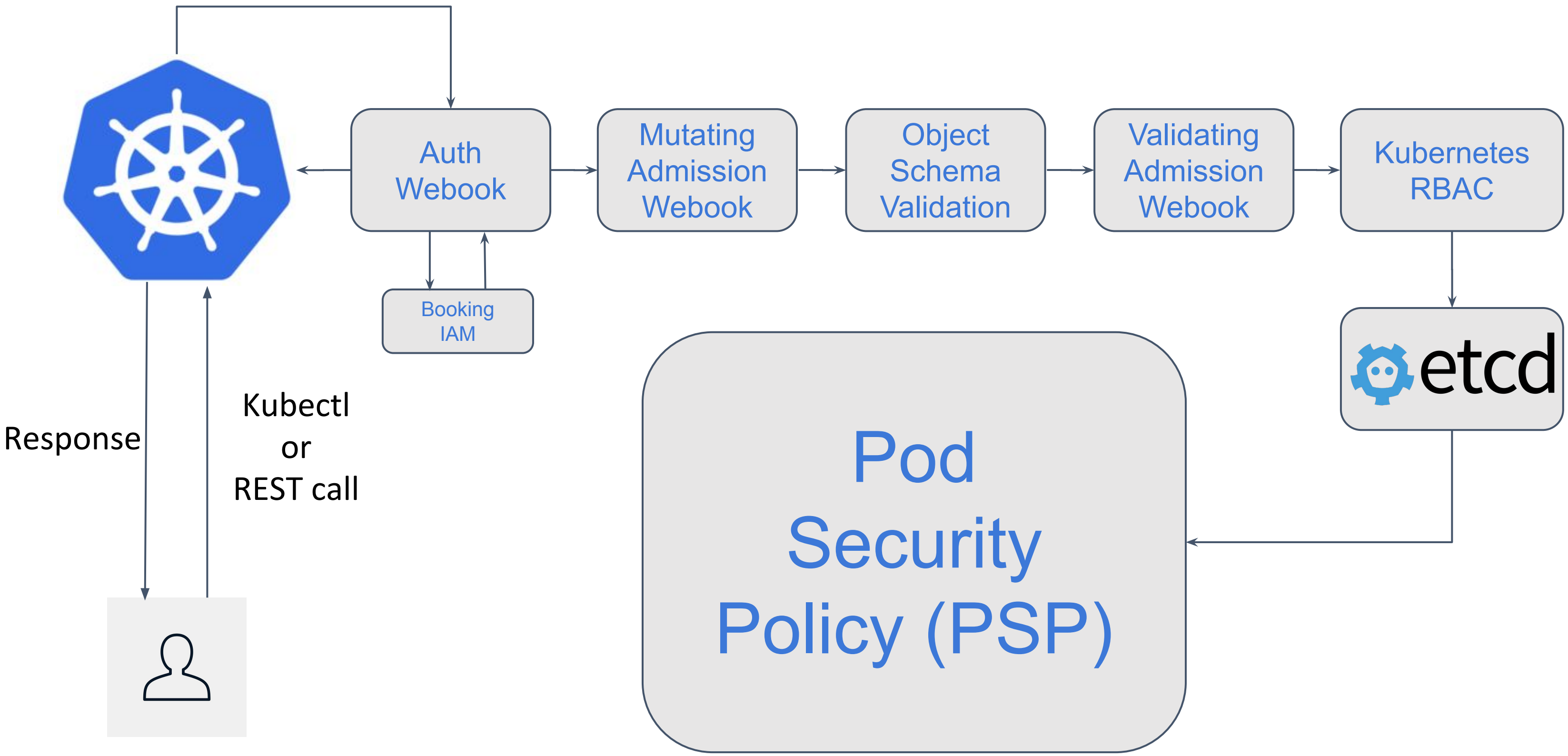
Lifecycle of kubectl command



Lifecycle of kubectl command



Lifecycle of kubectl command



Pod Security Policy (PSP)

Example use cases:

Pod Security Policy (PSP)

Example use cases:

- Deny running pod with UID 0 (root)

Pod Security Policy (PSP)

Example use cases:

- Deny running pod with UID 0 (root)
- Provide no access to host file system from within a pod

Pod Security Policy (PSP)

Example use cases:

- Deny running pod with UID 0 (root)
- Provide no access to host file system from within a pod
- Do not allow containerized processes to share

Pod Security Policy (PSP)

Example use cases:

- Deny running pod with UID 0 (root)
- Provide no access to host file system from within a pod
- Do not allow containerized processes to share
  - Host network
  - Host IPC
  - Host Process ID Namespace

Pod Security Policy (PSP)

Example use cases:

- Deny running pod with UID 0 (root)
- Provide no access to host file system from within a pod
- Do not allow containerized processes to share
  - Host network
  - Host IPC
  - Host Process ID Namespace
- Limit linux capabilities (eg. CAP_NET_ADMIN)

Pod Security Policy (PSP)

Example use cases:

- Deny running pod with UID 0 (root)
- Provide no access to host file system from within a pod
- Do not allow containerized processes to share
  - Host network
  - Host IPC
  - Host Process ID Namespace
- Limit linux capabilities (eg. CAP_NET_ADMIN)
- Allow certain range of UIDs

Pod Security Policy (PSP)

Example use cases:

- Deny running pod with UID 0 (root)
- Provide no access to host file system from within a pod
- Do not allow containerized processes to share
  - Host network
  - Host IPC
  - Host Process ID Namespace
- Limit linux capabilities (eg. CAP_NET_ADMIN)
- Allow certain range of UIDs
- Allow certain types of volumes (eg. secret, pvc, configmaps, downward api etc.)

Pod Security Policy (PSP)

- Configure kube-apiserver with the required flag

```yaml
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted'
spec:
  privileged: false
  allowPrivilegeEscalation: false
  requiredDropCapabilities:
    - ALL
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'secret'
    - 'downwardAPI'
    - 'persistentVolumeClaim'
  hostNetwork: false
  hostIPC: false
  hostPID: false
  runAsUser:
    rule: 'MustRunAsNonRoot'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  fsGroup:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  readOnlyRootFilesystem: false
```

Pod Security Policy (PSP)

- Configure kube-apiserver with the required flag

--enable-admission-plugins=
PodSecurityPolicy,...

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted'
spec:
  privileged: false
  allowPrivilegeEscalation: false
  requiredDropCapabilities:
    - ALL
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'secret'
    - 'downwardAPI'
    - 'persistentVolumeClaim'
  hostNetwork: false
  hostIPC: false
  hostPID: false
  runAsUser:
    rule: 'MustRunAsNonRoot'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  fsGroup:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  readOnlyRootFilesystem: false
```

Pod Security Policy (PSP)

- Configure kube-apiserver with the required flag

--enable-admission-plugins=
PodSecurityPolicy,...

- Create restricted PSP

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted'
spec:
  privileged: false
  allowPrivilegeEscalation: false
  requiredDropCapabilities:
    - ALL
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'secret'
    - 'downwardAPI'
    - 'persistentVolumeClaim'
  hostNetwork: false
  hostIPC: false
  hostPID: false
  runAsUser:
    rule: 'MustRunAsNonRoot'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  fsGroup:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  readOnlyRootFilesystem: false
```

Pod Security Policy (PSP)

- Configure kube-apiserver with the required flag

--enable-admission-plugins=
PodSecurityPolicy,...

- Create restricted PSP

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted'
spec:
  privileged: false
  allowPrivilegeEscalation: false
  requiredDropCapabilities:
    - ALL
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'secret'
    - 'downwardAPI'
    - 'persistentVolumeClaim'
  hostNetwork: false
  hostIPC: false
  hostPID: false
  runAsUser:
    rule: 'MustRunAsNonRoot'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  fsGroup:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  readOnlyRootFilesystem: false
```

Pod Security Policy (PSP)

- Configure kube-apiserver with the required flag

--enable-admission-plugins=
PodSecurityPolicy,...

- Create restricted PSP

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted'
spec:
  privileged: false
  allowPrivilegeEscalation: false
  requiredDropCapabilities:
    - ALL
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'secret'
    - 'downwardAPI'
    - 'persistentVolumeClaim'
  hostNetwork: false
  hostIPC: false
  hostPID: false
  runAsUser:
    rule: 'MustRunAsNonRoot'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  fsGroup:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  readOnlyRootFilesystem: false
```

Pod Security Policy (PSP)

- Configure kube-apiserver with the required flag

--enable-admission-plugins=
PodSecurityPolicy,...

- Create restricted PSP

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted'
spec:
  privileged: false
  allowPrivilegeEscalation: false
  requiredDropCapabilities:
    - ALL
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'secret'
    - 'downwardAPI'
    - 'persistentVolumeClaim'
  hostNetwork: false
  hostIPC: false
  hostPID: false
  runAsUser:
    rule: 'MustRunAsNonRoot'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  fsGroup:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  readOnlyRootFilesystem: false
```

Pod Security Policy (PSP)

- Configure kube-apiserver with the required flag

--enable-admission-plugins=
PodSecurityPolicy,...

- Create restricted PSP

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted'
spec:
  privileged: false
  allowPrivilegeEscalation: false
  requiredDropCapabilities:
    - ALL
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'secret'
    - 'downwardAPI'
    - 'persistentVolumeClaim'
  hostNetwork: false
  hostIPC: false
  hostPID: false
  runAsUser:
    rule: 'MustRunAsNonRoot'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  fsGroup:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  readOnlyRootFilesystem: false
```

Pod Security Policy (PSP)

- Configure kube-apiserver with the required flag

--enable-admission-plugins=
PodSecurityPolicy,...

- Create restricted PSP

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted'
spec:
  privileged: false
  allowPrivilegeEscalation: false
  requiredDropCapabilities:
    - ALL
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'secret'
    - 'downwardAPI'
    - 'persistentVolumeClaim'
  hostNetwork: false
  hostIPC: false
  hostPID: false
  runAsUser:
    rule: 'MustRunAsNonRoot'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  fsGroup:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  readOnlyRootFilesystem: false
```

Pod Security Policy (PSP)

- Configure kube-apiserver with the required flag

--enable-admission-plugins=
PodSecurityPolicy,...

- Create restricted PSP

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted'
spec:
  privileged: false
  allowPrivilegeEscalation: false
  requiredDropCapabilities:
    - ALL
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'secret'
    - 'downwardAPI'
    - 'persistentVolumeClaim'
  hostNetwork: false
  hostIPC: false
  hostPID: false
  runAsUser:
    rule: 'MustRunAsNonRoot'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  fsGroup:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  readOnlyRootFilesystem: false
```

Pod Security Policy (PSP)

- Configure kube-apiserver with the required flag

--enable-admission-plugins=
PodSecurityPolicy,...

- Create restricted PSP

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted'
spec:
  privileged: false
  allowPrivilegeEscalation: false
  requiredDropCapabilities:
    - ALL
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'secret'
    - 'downwardAPI'
    - 'persistentVolumeClaim'
  hostNetwork: false
  hostIPC: false
  hostPID: false
  runAsUser:
    rule: 'MustRunAsNonRoot'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  fsGroup:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  readOnlyRootFilesystem: false
```

Pod Security Policy (PSP)

- Configure kube-apiserver with the required flag

--enable-admission-plugins=
PodSecurityPolicy,...

- Create restricted PSP

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted'
spec:
  privileged: false
  allowPrivilegeEscalation: false
  requiredDropCapabilities:
    - ALL
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'secret'
    - 'downwardAPI'
    - 'persistentVolumeClaim'
  hostNetwork: false
  hostIPC: false
  hostPID: false
  runAsUser:
    rule: 'MustRunAsNonRoot'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  fsGroup:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  readOnlyRootFilesystem: false
```

Pod Security Policy (PSP)

- Configure kube-apiserver with the required flag

--enable-admission-plugins=
PodSecurityPolicy,...

- Create restricted PSP

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted'
spec:
  privileged: false
  allowPrivilegeEscalation: false
  requiredDropCapabilities:
    - ALL
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'secret'
    - 'downwardAPI'
    - 'persistentVolumeClaim'
  hostNetwork: false
  hostIPC: false
  hostPID: false
  runAsUser:
    rule: 'MustRunAsNonRoot'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  fsGroup:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  readOnlyRootFilesystem: false
```

Pod Security Policy (PSP)

- Configure kube-apiserver with the required flag

--enable-admission-plugins=
PodSecurityPolicy,...

- Create restricted PSP

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted'
spec:
  privileged: false
  allowPrivilegeEscalation: false
  requiredDropCapabilities:
    - ALL
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'secret'
    - 'downwardAPI'
    - 'persistentVolumeClaim'
  hostNetwork: false
  hostIPC: false
  hostPID: false
  runAsUser:
    rule: 'MustRunAsNonRoot'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  fsGroup:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  readOnlyRootFilesystem: false
```

Pod Security Policy (PSP)

- Configure kube-apiserver with the required flag

--enable-admission-plugins=
PodSecurityPolicy,...

- Create restricted PSP

```yaml
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted'
spec:
  privileged: false
  allowPrivilegeEscalation: false
  requiredDropCapabilities:
    - ALL
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'secret'
    - 'downwardAPI'
    - 'persistentVolumeClaim'
  hostNetwork: false
  hostIPC: false
  hostPID: false
  runAsUser:
    rule: 'MustRunAsNonRoot'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  fsGroup:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  readOnlyRootFilesystem: false
```

Pod Security Policy (PSP)

- Configure kube-apiserver with the required flag

--enable-admission-plugins=
PodSecurityPolicy,...

- Create restricted PSP

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted'
spec:
  privileged: false
  allowPrivilegeEscalation: false
  requiredDropCapabilities:
    - ALL
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'secret'
    - 'downwardAPI'
    - 'persistentVolumeClaim'
  hostNetwork: false
  hostIPC: false
  hostPID: false
  runAsUser:
    rule: 'MustRunAsNonRoot'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  fsGroup:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  readOnlyRootFilesystem: false
```

Pod Security Policy (PSP)

- Configure kube-apiserver with the required flag

--enable-admission-plugins=
PodSecurityPolicy,...

- Create restricted PSP

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted'
spec:
  privileged: false
  allowPrivilegeEscalation: false
  requiredDropCapabilities:
    - ALL
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'secret'
    - 'downwardAPI'
    - 'persistentVolumeClaim'
  hostNetwork: false
  hostIPC: false
  hostPID: false
  runAsUser:
    rule: 'MustRunAsNonRoot'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  fsGroup:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  readOnlyRootFilesystem: false
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for restricted PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: restricted
rules:
- apiGroups:
  - extensions
  resourceNames:
  - restricted
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restricted
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restricted
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for restricted PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: restricted
rules:
- apiGroups:
  - extensions
  resourceNames:
  - restricted
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restricted
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restricted
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for restricted PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: restricted
rules:
- apiGroups:
  - extensions
  resourceNames:
  - restricted
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restricted
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restricted
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for restricted PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: restricted
rules:
- apiGroups:
  - extensions
  resourceNames:
  - restricted
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restricted
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restricted
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for restricted PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: restricted
rules:
- apiGroups:
  - extensions
  resourceNames:
  - restricted
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restricted
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restricted
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for restricted PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: restricted
rules:
- apiGroups:
  - extensions
  resourceNames:
  - restricted
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restricted
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restricted
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for restricted PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: restricted
rules:
- apiGroups:
  - extensions
  resourceNames:
  - restricted
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restricted
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restricted
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for restricted PSP

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: restricted
rules:
- apiGroups:
  - extensions
  resourceNames:
  - restricted
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restricted
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restricted
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for restricted PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: restricted
rules:
- apiGroups:
  - extensions
  resourceNames:
  - restricted
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restricted
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restricted
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for restricted PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: restricted
rules:
- apiGroups:
  - extensions
  resourceNames:
  - restricted
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restricted
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restricted
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for restricted PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: restricted
rules:
- apiGroups:
  - extensions
  resourceNames:
  - restricted
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restricted
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restricted
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for restricted PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: restricted
rules:
- apiGroups:
  - extensions
  resourceNames:
  - restricted
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restricted
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restricted
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for restricted PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: restricted
rules:
- apiGroups:
  - extensions
  resourceNames:
  - restricted
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restricted
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restricted
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for restricted PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: restricted
rules:
- apiGroups:
  - extensions
  resourceNames:
  - restricted
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restricted
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restricted
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for restricted PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: restricted
rules:
- apiGroups:
  - extensions
  resourceNames:
  - restricted
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restricted
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restricted
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for restricted PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: restricted
rules:
- apiGroups:
  - extensions
  resourceNames:
  - restricted
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restricted
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restricted
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for restricted PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: restricted
rules:
- apiGroups:
  - extensions
  resourceNames:
  - restricted
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restricted
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restricted
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for restricted PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: restricted
rules:
- apiGroups:
  - extensions
  resourceNames:
  - restricted
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restricted
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restricted
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated
```

Pod Security Policy (PSP)

Create privileged PSP

```yaml
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: privileged
spec:
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  fsGroup:
    rule: RunAsAny
  hostIPC: true
  hostNetwork: true
  hostPID: true
  hostPorts:
  - max: 65535
    min: 0
  privileged: true
  runAsUser:
    rule: RunAsAny
  seLinux:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - '*'
```

Pod Security Policy (PSP)

Create privileged PSP

```yaml
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: privileged
spec:
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  fsGroup:
    rule: RunAsAny
  hostIPC: true
  hostNetwork: true
  hostPID: true
  hostPorts:
  - max: 65535
    min: 0
  privileged: true
  runAsUser:
    rule: RunAsAny
  seLinux:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - '*'
```

Pod Security Policy (PSP)

Create privileged PSP

```
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: privileged
spec:
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  fsGroup:
    rule: RunAsAny
  hostIPC: true
  hostNetwork: true
  hostPID: true
  hostPorts:
  - max: 65535
    min: 0
  privileged: true
  runAsUser:
    rule: RunAsAny
  seLinux:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - '*'
```

Pod Security Policy (PSP)

Create privileged PSP

```
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: privileged
spec:
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  fsGroup:
    rule: RunAsAny
  hostIPC: true
  hostNetwork: true
  hostPID: true
  hostPorts:
  - max: 65535
    min: 0
  privileged: true
  runAsUser:
    rule: RunAsAny
  seLinux:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - '*'
```

Pod Security Policy (PSP)

Create privileged PSP

```
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: privileged
spec:
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  fsGroup:
    rule: RunAsAny
  hostIPC: true
  hostNetwork: true
  hostPID: true
  hostPorts:
  - max: 65535
    min: 0
  privileged: true
  runAsUser:
    rule: RunAsAny
  seLinux:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - '*'
```

Pod Security Policy (PSP)

Create privileged PSP

```yaml
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: privileged
spec:
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  fsGroup:
    rule: RunAsAny
  hostIPC: true
  hostNetwork: true
  hostPID: true
  hostPorts:
  - max: 65535
    min: 0
  privileged: true
  runAsUser:
    rule: RunAsAny
  seLinux:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - '*'
```

Pod Security Policy (PSP)

Create privileged PSP

```
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: privileged
spec:
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  fsGroup:
    rule: RunAsAny
  hostIPC: true
  hostNetwork: true
  hostPID: true
  hostPorts:
  - max: 65535
    min: 0
  privileged: true
  runAsUser:
    rule: RunAsAny
  seLinux:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - '*'
```

Pod Security Policy (PSP)

Create privileged PSP

```
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: privileged
spec:
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  fsGroup:
    rule: RunAsAny
  hostIPC: true
  hostNetwork: true
  hostPID: true
  hostPorts:
  - max: 65535
    min: 0
  privileged: true
  runAsUser:
    rule: RunAsAny
  seLinux:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - '*'
```

Pod Security Policy (PSP)

Create privileged PSP

```
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: privileged
spec:
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  fsGroup:
    rule: RunAsAny
  hostIPC: true
  hostNetwork: true
  hostPID: true
  hostPorts:
  - max: 65535
    min: 0
  privileged: true
  runAsUser:
    rule: RunAsAny
  seLinux:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - '*'
```

Pod Security Policy (PSP)

Create privileged PSP

```
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: privileged
spec:
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  fsGroup:
    rule: RunAsAny
  hostIPC: true
  hostNetwork: true
  hostPID: true
  hostPorts:
  - max: 65535
    min: 0
  privileged: true
  runAsUser:
    rule: RunAsAny
  seLinux:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - '*'
```

Pod Security Policy (PSP)

Create privileged PSP

```
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: privileged
spec:
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  fsGroup:
    rule: RunAsAny
  hostIPC: true
  hostNetwork: true
  hostPID: true
  hostPorts:
  - max: 65535
    min: 0
  privileged: true
  runAsUser:
    rule: RunAsAny
  seLinux:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - '*'
```

Pod Security Policy (PSP)

Create privileged PSP

```
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: privileged
spec:
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  fsGroup:
    rule: RunAsAny
  hostIPC: true
  hostNetwork: true
  hostPID: true
  hostPorts:
  - max: 65535
    min: 0
  privileged: true
  runAsUser:
    rule: RunAsAny
  seLinux:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - '*'
```

Pod Security Policy (PSP)

Create privileged PSP

```yaml
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: privileged
spec:
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  fsGroup:
    rule: RunAsAny
  hostIPC: true
  hostNetwork: true
  hostPID: true
  hostPorts:
  - max: 65535
    min: 0
  privileged: true
  runAsUser:
    rule: RunAsAny
  seLinux:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - '*'
```

Pod Security Policy (PSP)

Create privileged PSP

```yaml
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: privileged
spec:
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  fsGroup:
    rule: RunAsAny
  hostIPC: true
  hostNetwork: true
  hostPID: true
  hostPorts:
  - max: 65535
    min: 0
  privileged: true
  runAsUser:
    rule: RunAsAny
  seLinux:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - '*'
```

Pod Security Policy (PSP)

Create privileged PSP

```yaml
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: privileged
spec:
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  fsGroup:
    rule: RunAsAny
  hostIPC: true
  hostNetwork: true
  hostPID: true
  hostPorts:
  - max: 65535
    min: 0
  privileged: true
  runAsUser:
    rule: RunAsAny
  seLinux:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - '*'
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for privileged PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: privileged
rules:
- apiGroups:
  - extensions
  resourceNames:
  - privileged
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: privileged
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: privileged
subjects:
- kind: ServiceAccount
  name: default
  namespace: kube-system
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: clusterAdmins
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for privileged PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: privileged
rules:
- apiGroups:
  - extensions
  resourceNames:
  - privileged
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: privileged
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: privileged
subjects:
- kind: ServiceAccount
  name: default
  namespace: kube-system
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: clusterAdmins
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for privileged PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: privileged
rules:
- apiGroups:
  - extensions
  resourceNames:
  - privileged
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: privileged
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: privileged
subjects:
- kind: ServiceAccount
  name: default
  namespace: kube-system
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: clusterAdmins
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for privileged PSP

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: privileged
rules:
- apiGroups:
  - extensions
  resourceNames:
  - privileged
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: privileged
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: privileged
subjects:
- kind: ServiceAccount
  name: default
  namespace: kube-system
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: clusterAdmins
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for privileged PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: privileged
rules:
- apiGroups:
  - extensions
  resourceNames:
  - privileged
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: privileged
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: privileged
subjects:
- kind: ServiceAccount
  name: default
  namespace: kube-system
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: clusterAdmins
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for privileged PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: privileged
rules:
- apiGroups:
  - extensions
  resourceNames:
  - privileged
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: privileged
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: privileged
subjects:
- kind: ServiceAccount
  name: default
  namespace: kube-system
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: clusterAdmins
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for privileged PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: privileged
rules:
- apiGroups:
  - extensions
  resourceNames:
  - privileged
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: privileged
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: privileged
subjects:
- kind: ServiceAccount
  name: default
  namespace: kube-system
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: clusterAdmins
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for privileged PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: privileged
rules:
- apiGroups:
  - extensions
  resourceNames:
  - privileged
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: privileged
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: privileged
subjects:
- kind: ServiceAccount
  name: default
  namespace: kube-system
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: clusterAdmins
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for privileged PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: privileged
rules:
- apiGroups:
  - extensions
  resourceNames:
  - privileged
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: privileged
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: privileged
subjects:
- kind: ServiceAccount
  name: default
  namespace: kube-system
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: clusterAdmins
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for privileged PSP

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: privileged
rules:
- apiGroups:
  - extensions
  resourceNames:
  - privileged
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: privileged
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: privileged
subjects:
- kind: ServiceAccount
  name: default
  namespace: kube-system
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: clusterAdmins
```

Pod Security Policy (PSP)

Create cluster role and cluster role binding for privileged PSP

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: privileged
rules:
- apiGroups:
  - extensions
  resourceNames:
  - privileged
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: privileged
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: privileged
subjects:
- kind: ServiceAccount
  name: default
  namespace: kube-system
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: clusterAdmins
```

Pod Security Policy (Ordering)

What if multiple policies are applicable to a pod creation request?

Pod Security Policy (Ordering)

What if multiple policies are applicable to a pod creation request?

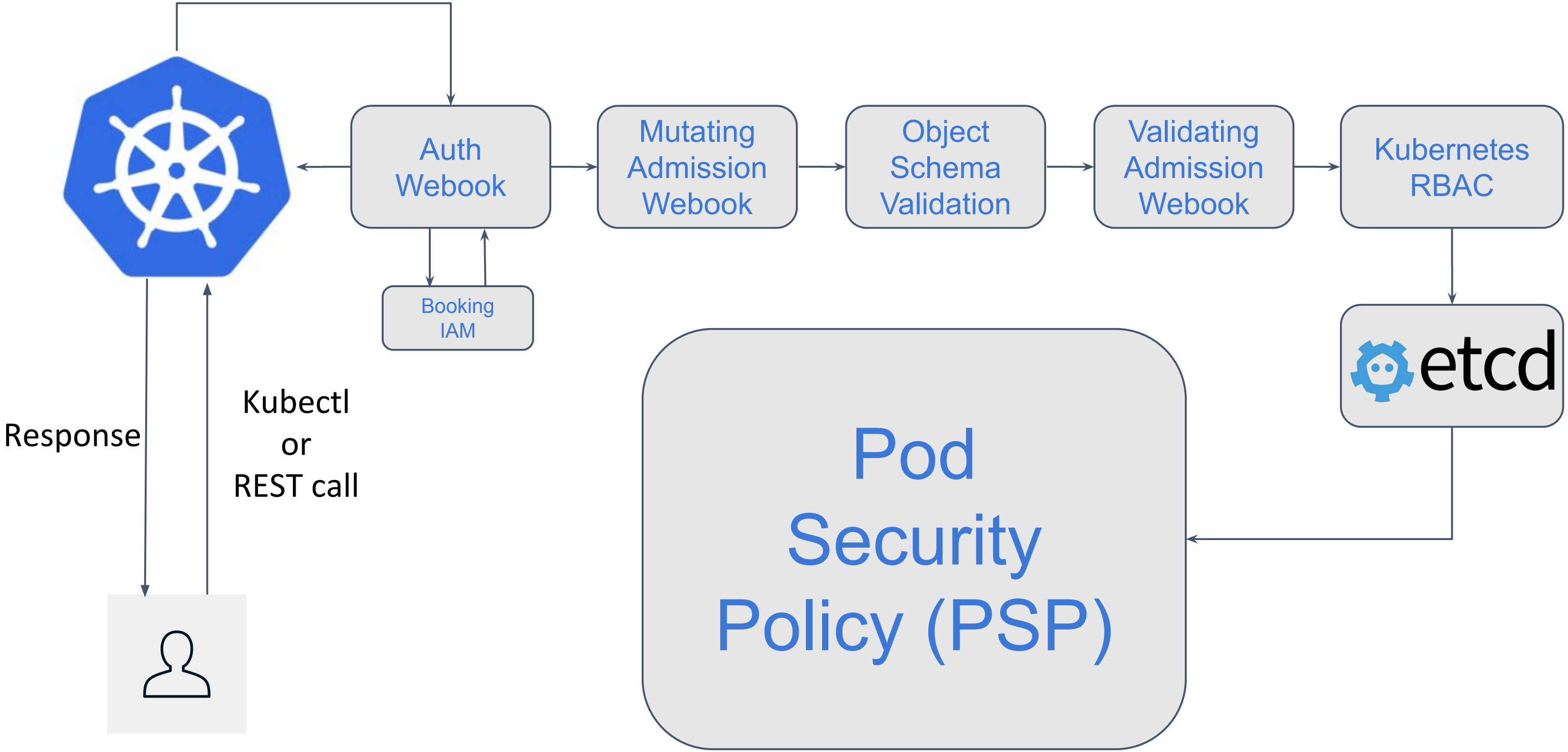
- Any guesses?

Pod Security Policy (Ordering)

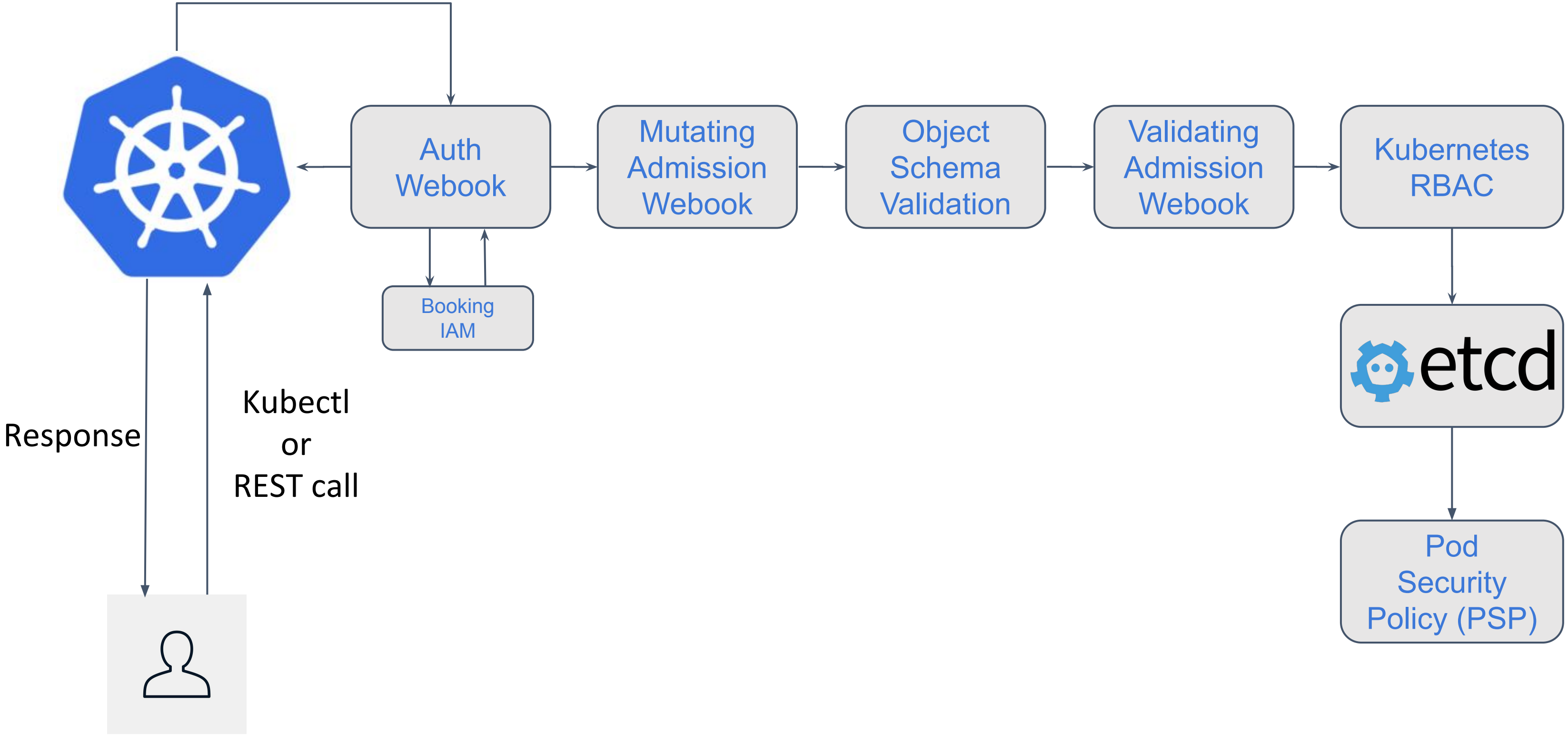
What if multiple policies are applicable to a pod creation request?

- The first valid policy in alphabetical order is used

Lifecycle of kubectl command

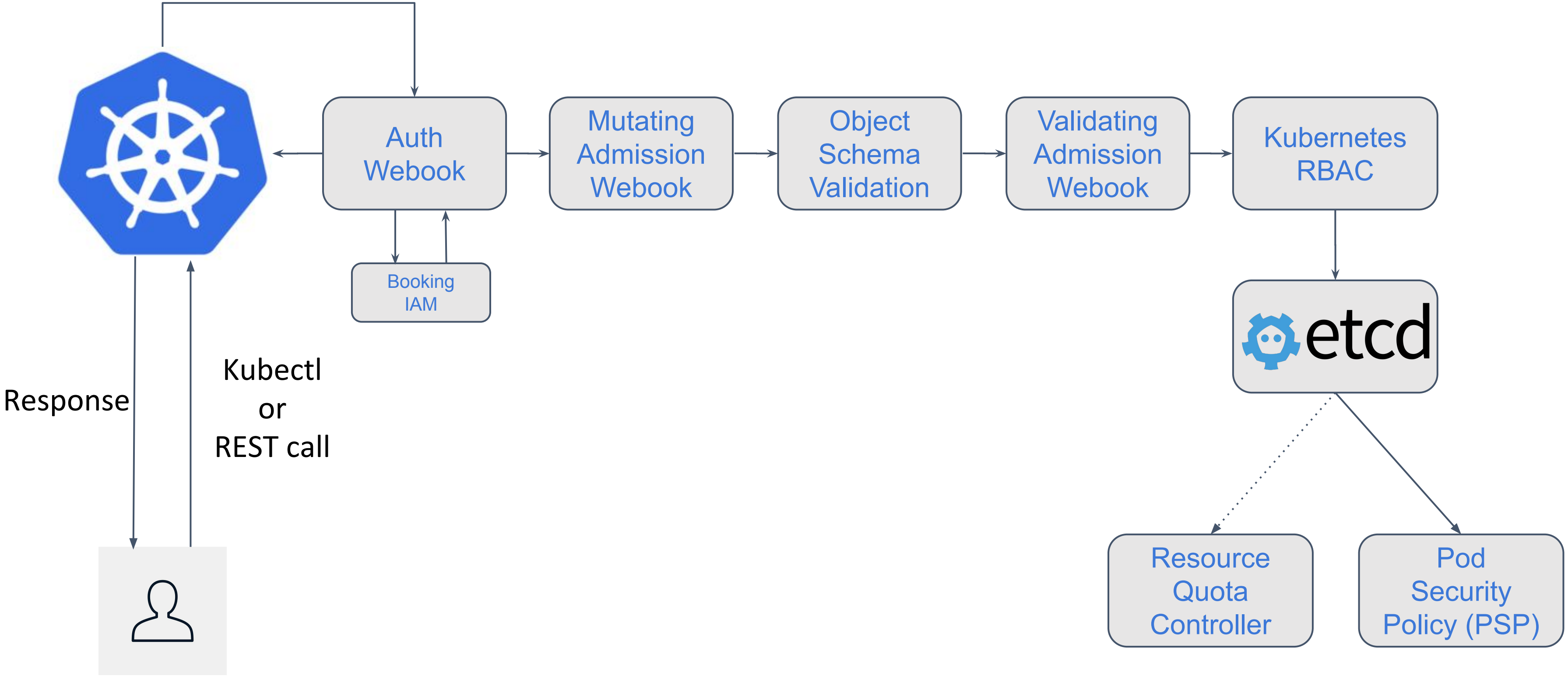


Lifecycle of kubectl command



Lifecycle of kubectl command

KubeCon CloudNativeCon Europe 2019



Auth Webook
Booking IAM
Mutating Admission Webook
Object Schema Validation
Validating Admission Webook
Kubernetes RBAC
etcd
Resource Quota Controller
Pod Security Policy (PSP)
Response
Kubectl or REST call

Key Takeaways

Key Takeaways

- Customize workflow using custom controllers (maybe using a framework), which opens the door to limitless automation

Key Takeaways

- Customize workflow using custom controllers (maybe using a framework), which opens the door to limitless automation

- Re-use your organization's existing auth workflow with your kubernetes setup using Kubernetes auth webhook

Key Takeaways

- Customize workflow using custom controllers (maybe using a framework), which opens the door to limitless automation

- Re-use your organization's existing auth workflow with your kubernetes setup using Kubernetes auth webhook

- Kubernetes admission controllers provide (using webhooks) a lot of opportunities to secure and customize resources being created in your kubernetes clusters

Key Takeaways

- Customize workflow using custom controllers (maybe using a framework), which opens the door to limitless automation
- Re-use your organization’s existing auth workflow with your kubernetes setup using Kubernetes auth webhook
- Kubernetes admission controllers provide (using webhooks) a lot of opportunities to secure and customize resources being created in your kubernetes clusters
- Take the opportunity to use PSPs (Pod Security Policies) to enforce a secure environment for your workloads to run

Thank you!

Thank you!

We’re Hiring

careers.booking.com

Booking.com

Questions

Q/A