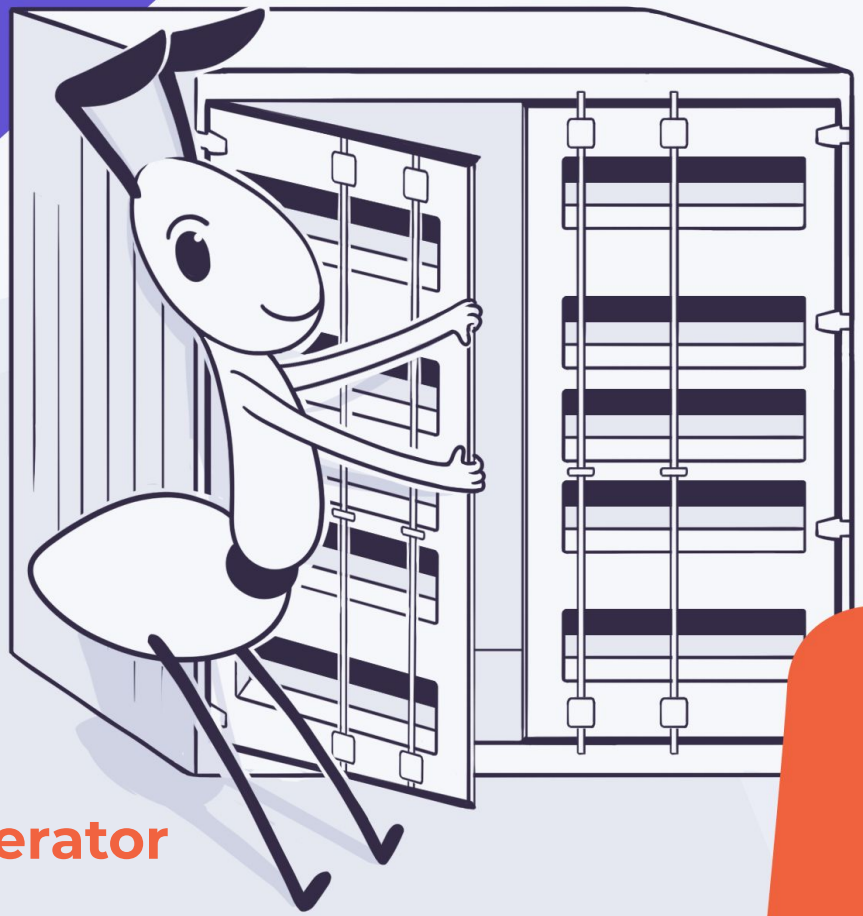
FLANT

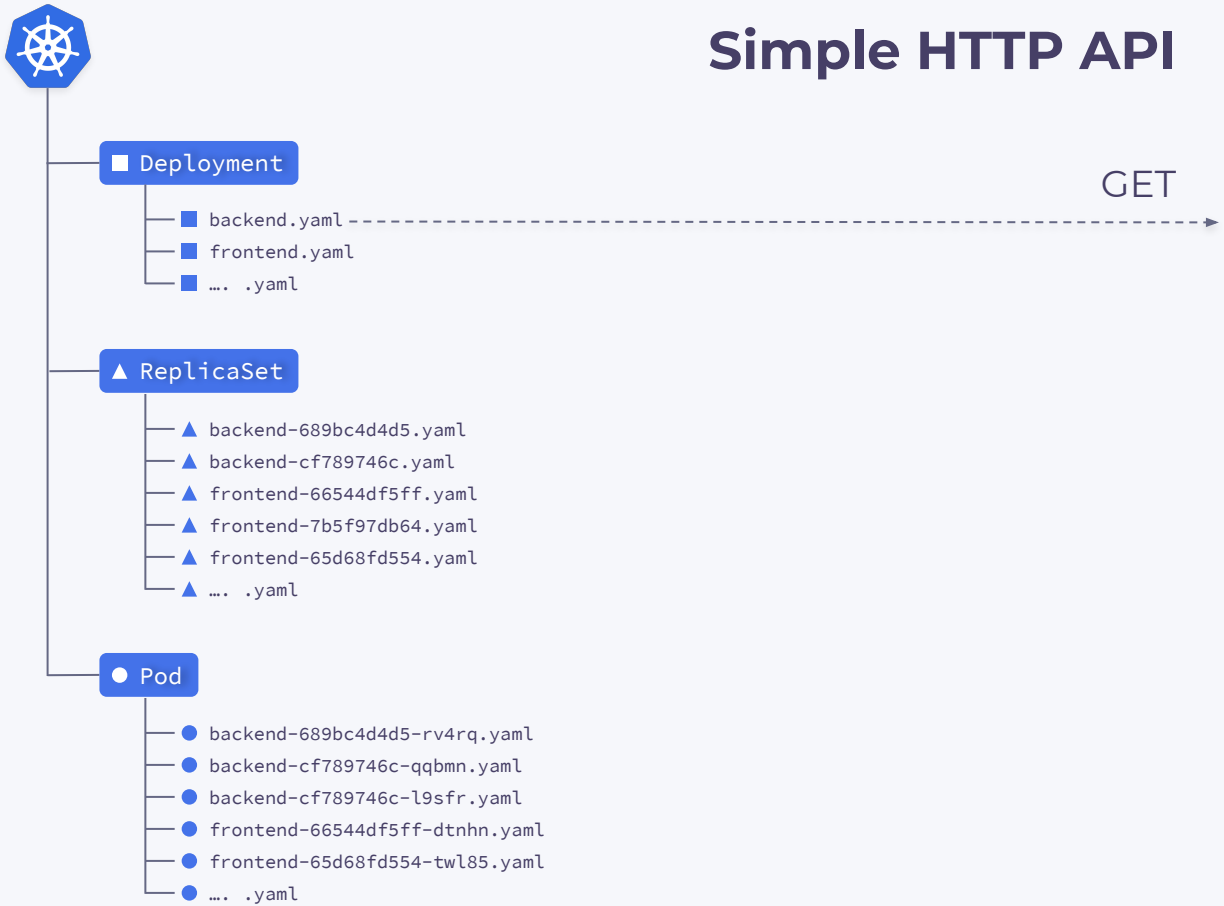


Go? Bash! Meet the Shell-operator

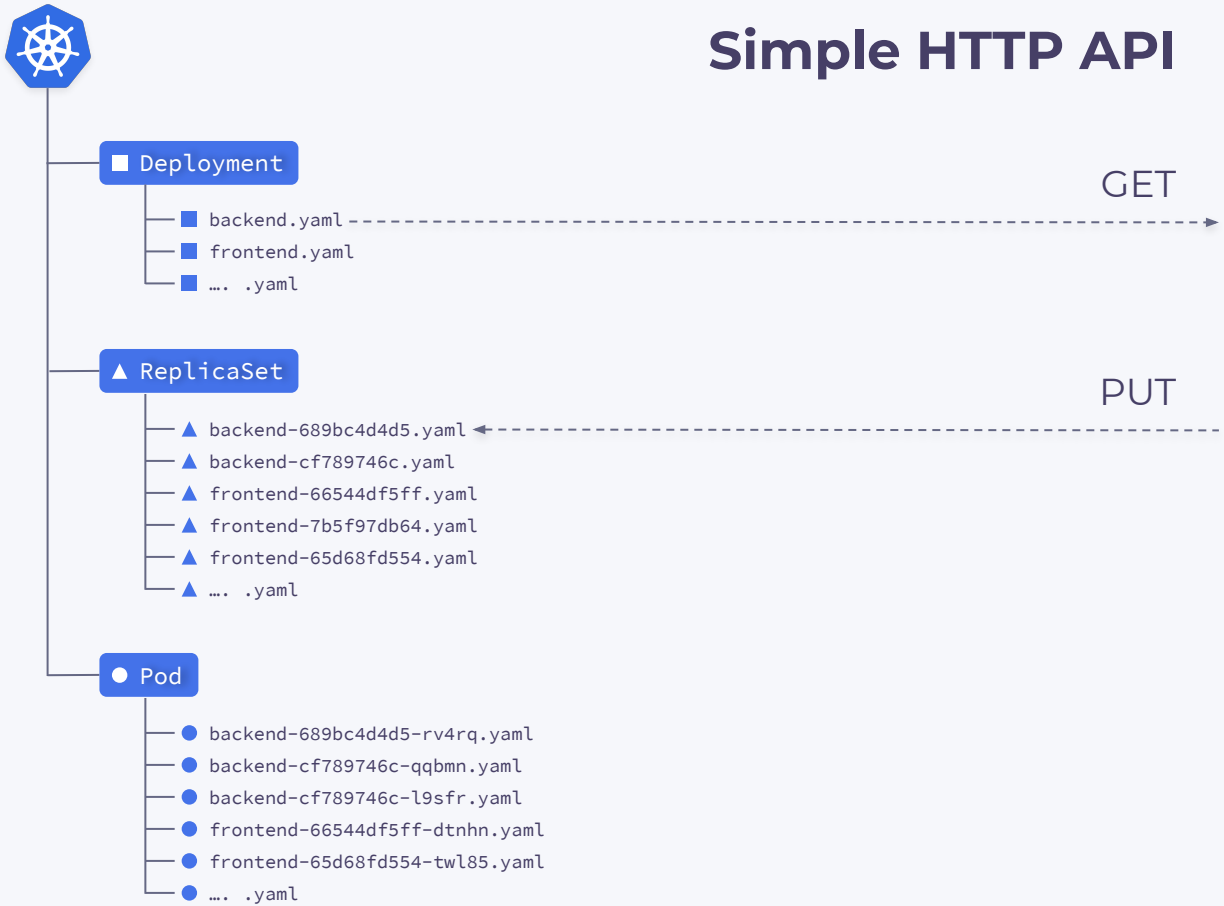
Andrey Klimentyev & Dmitry Stolyarov

Kubernetes API Basics

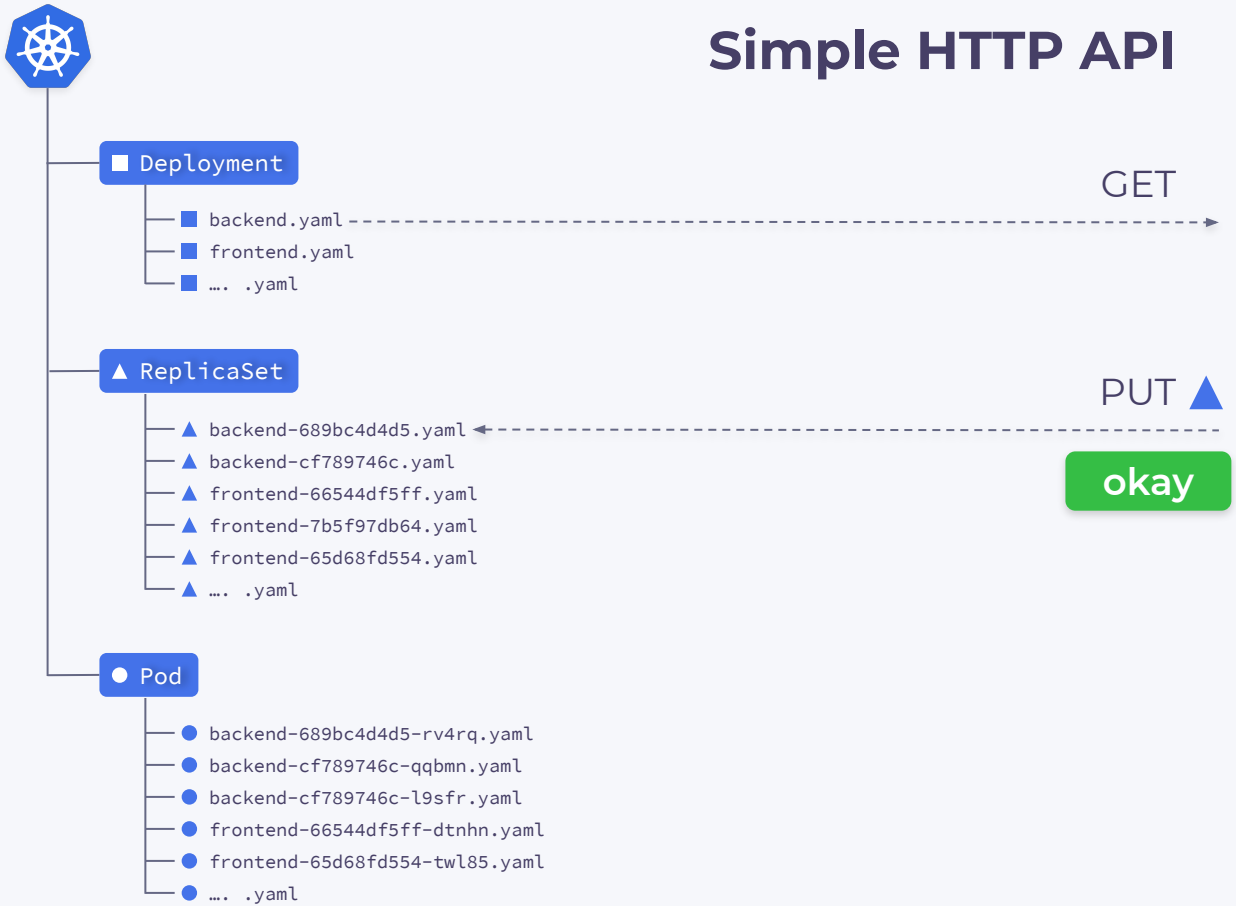
Simple HTTP API



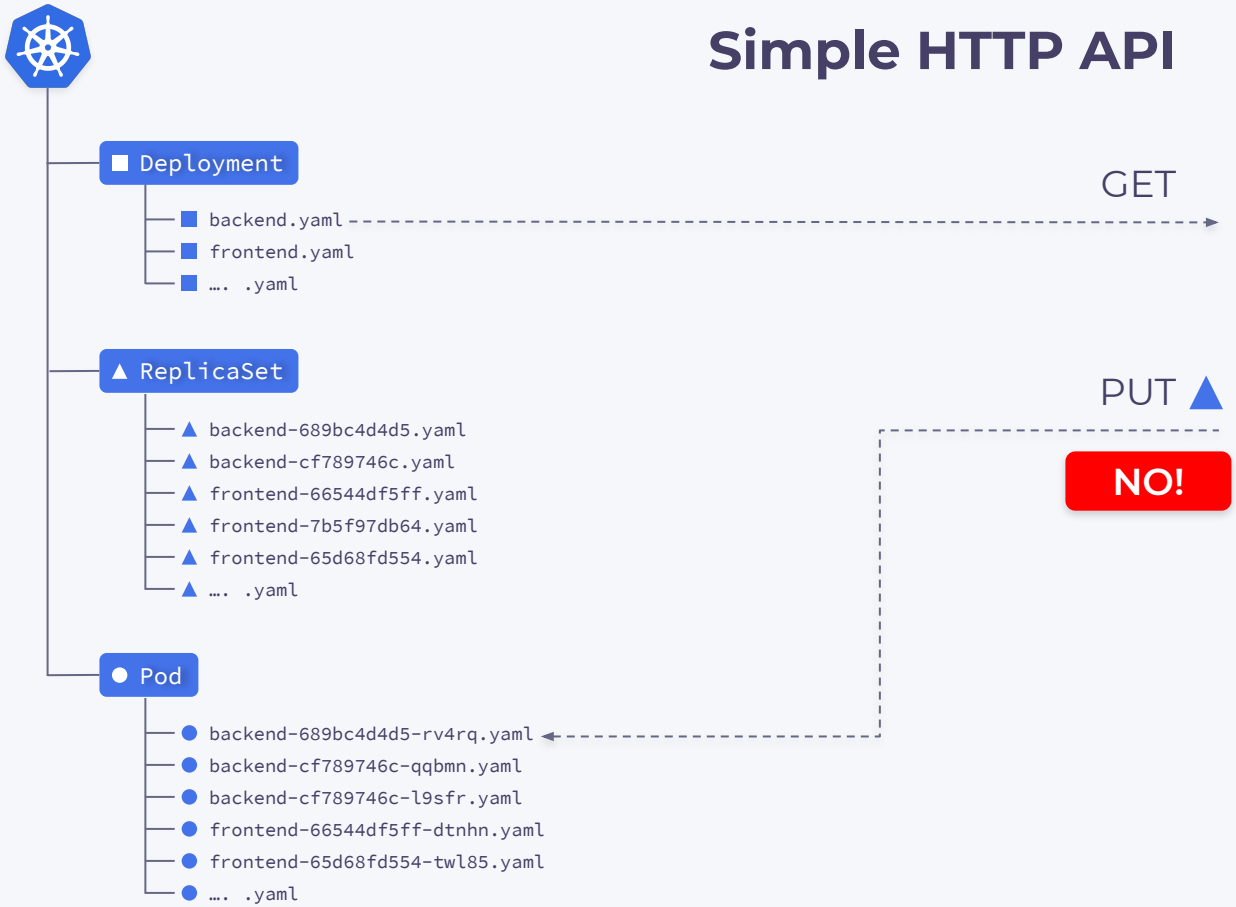
Simple HTTP API



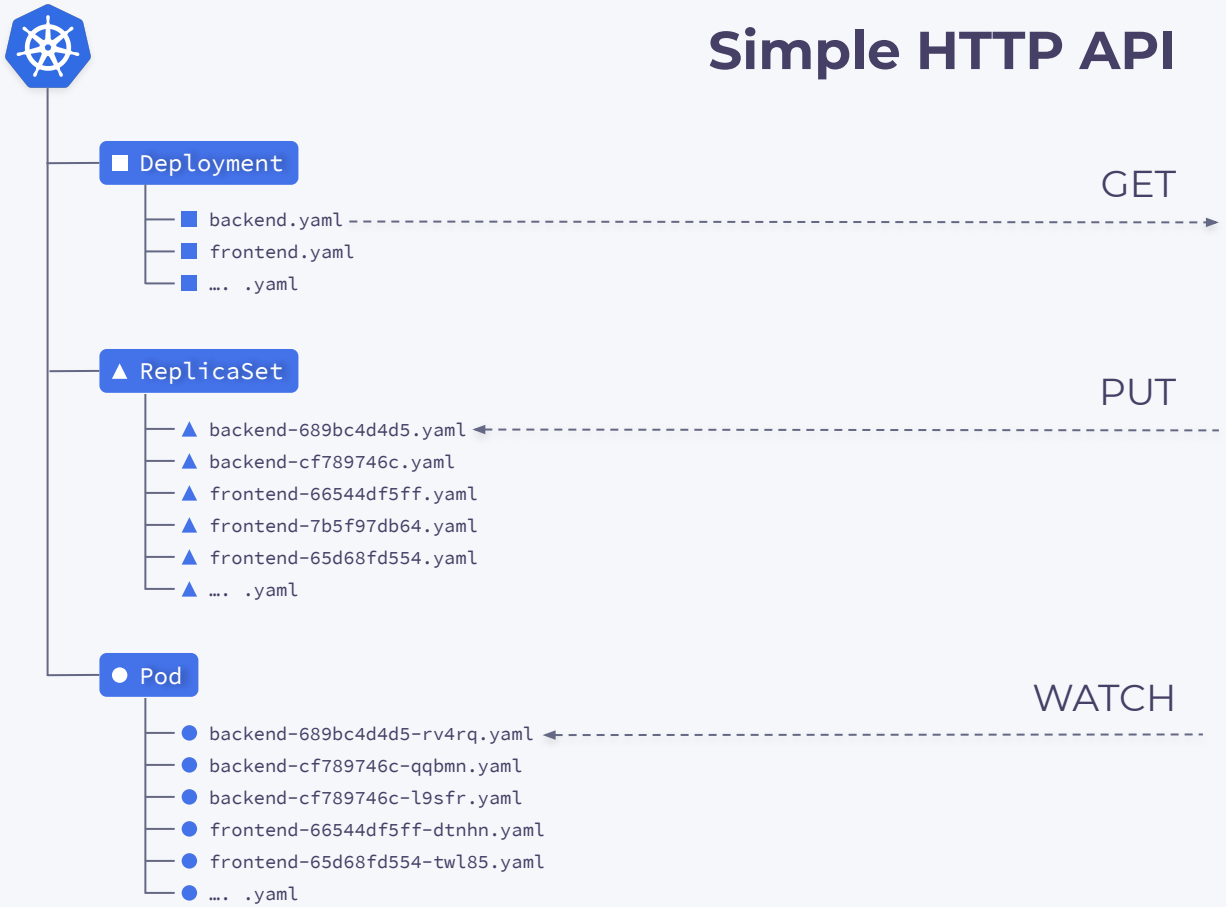
Simple HTTP API



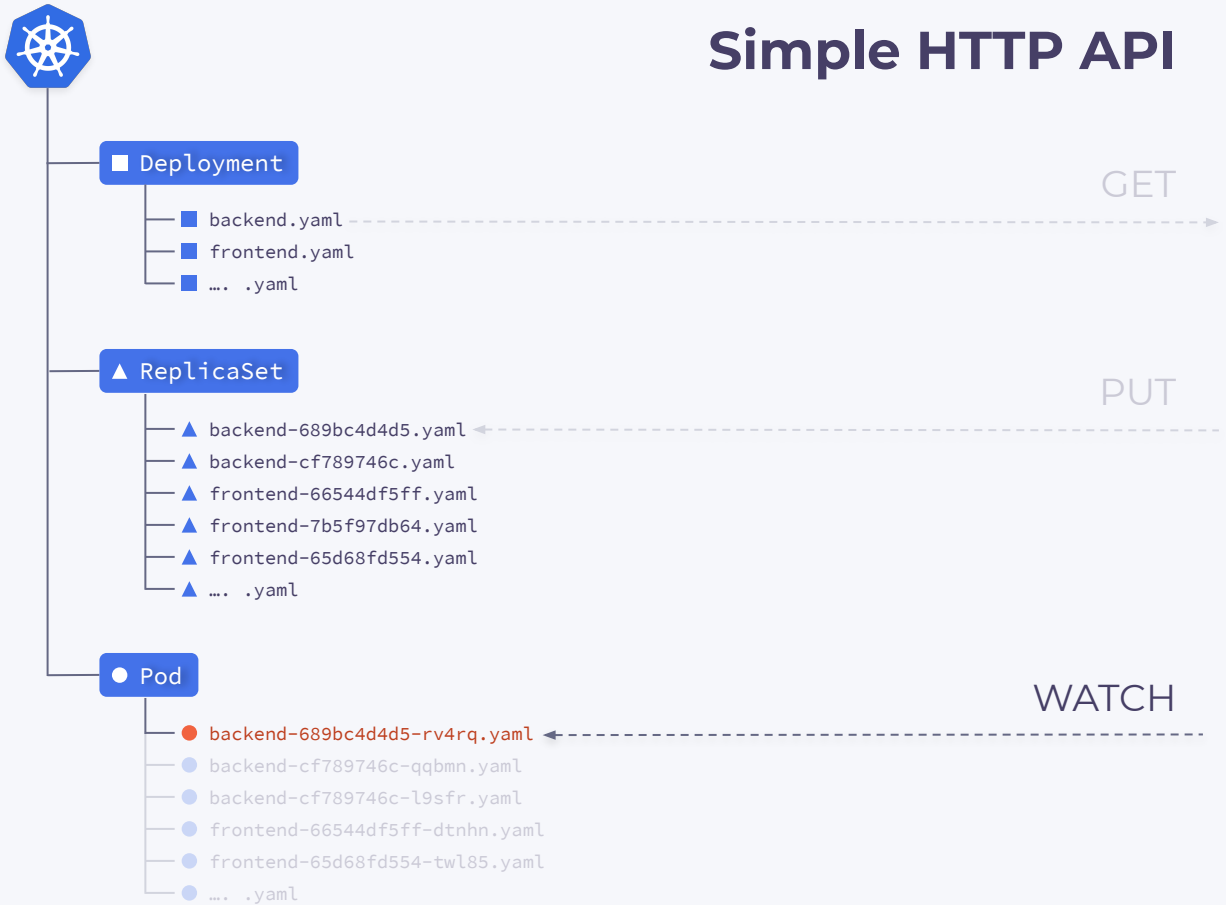
Simple HTTP API



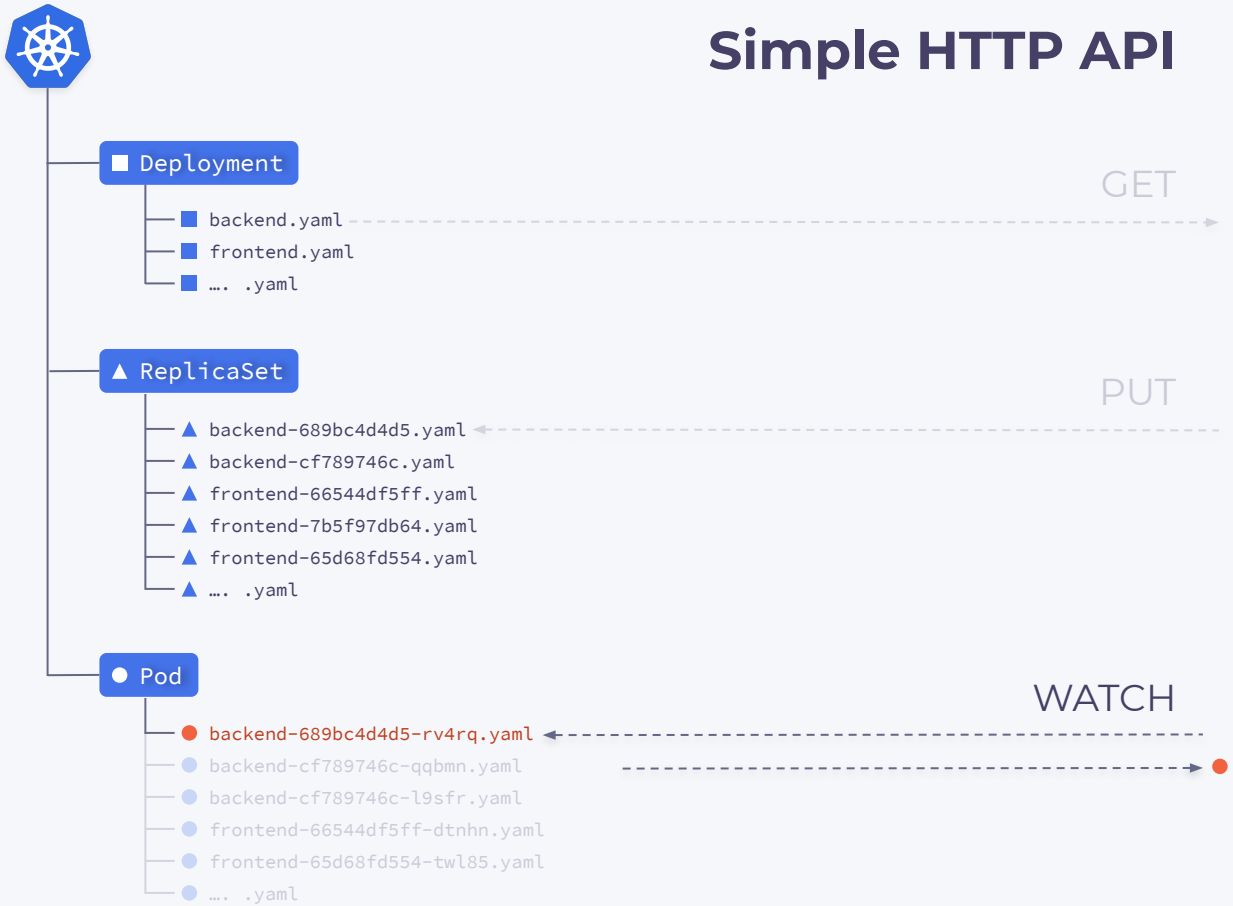
Simple HTTP API



Simple HTTP API

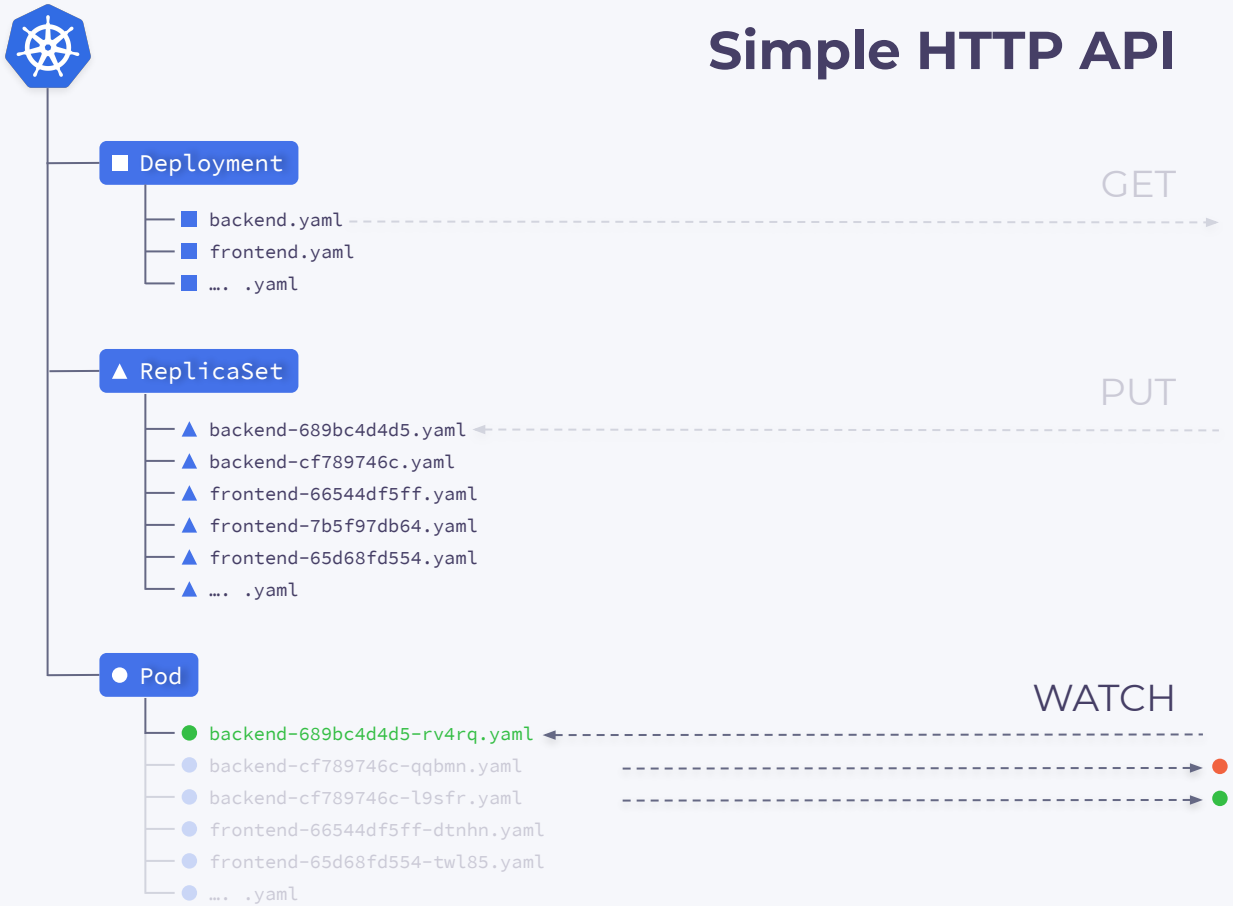


Simple HTTP API

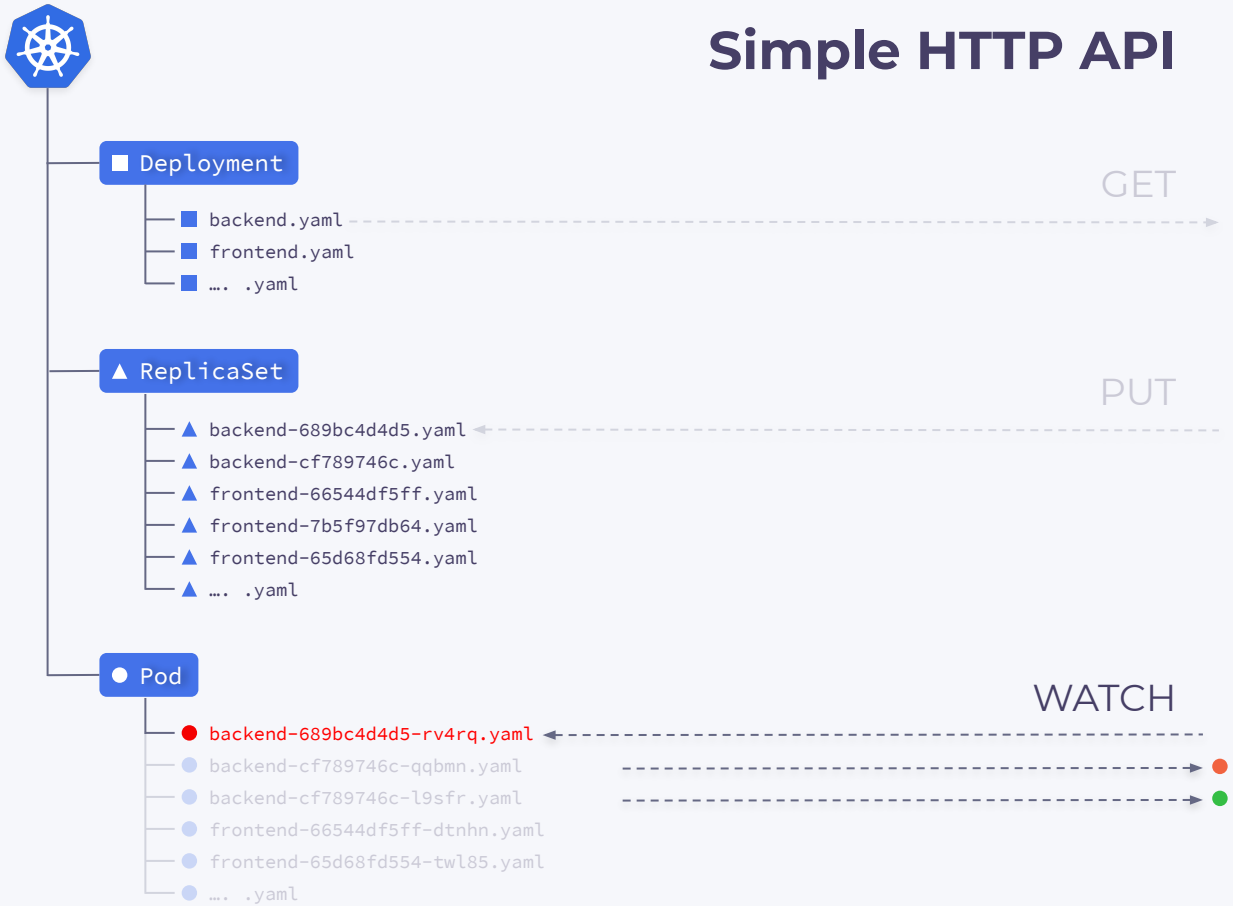


Simple HTTP API

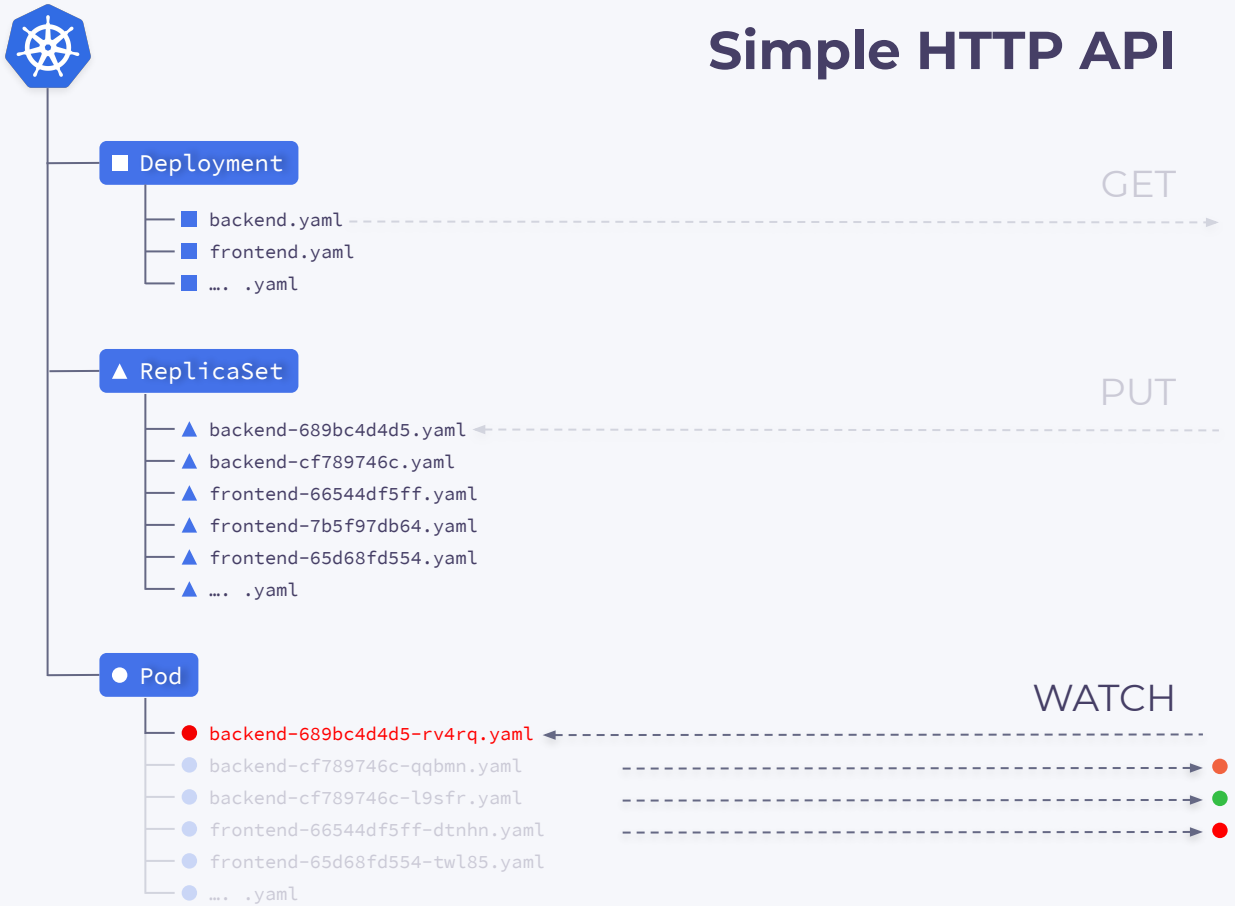
Simple HTTP API



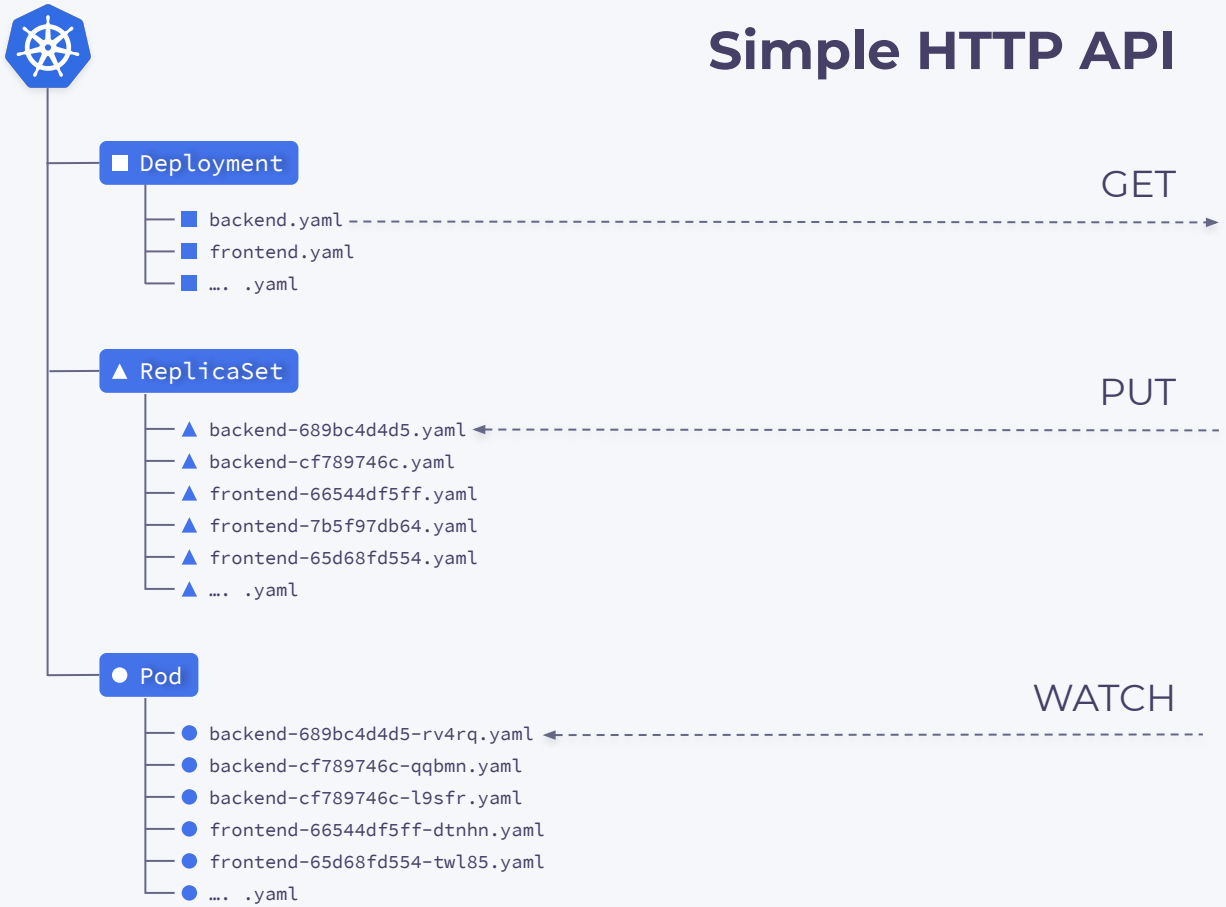
Simple HTTP API

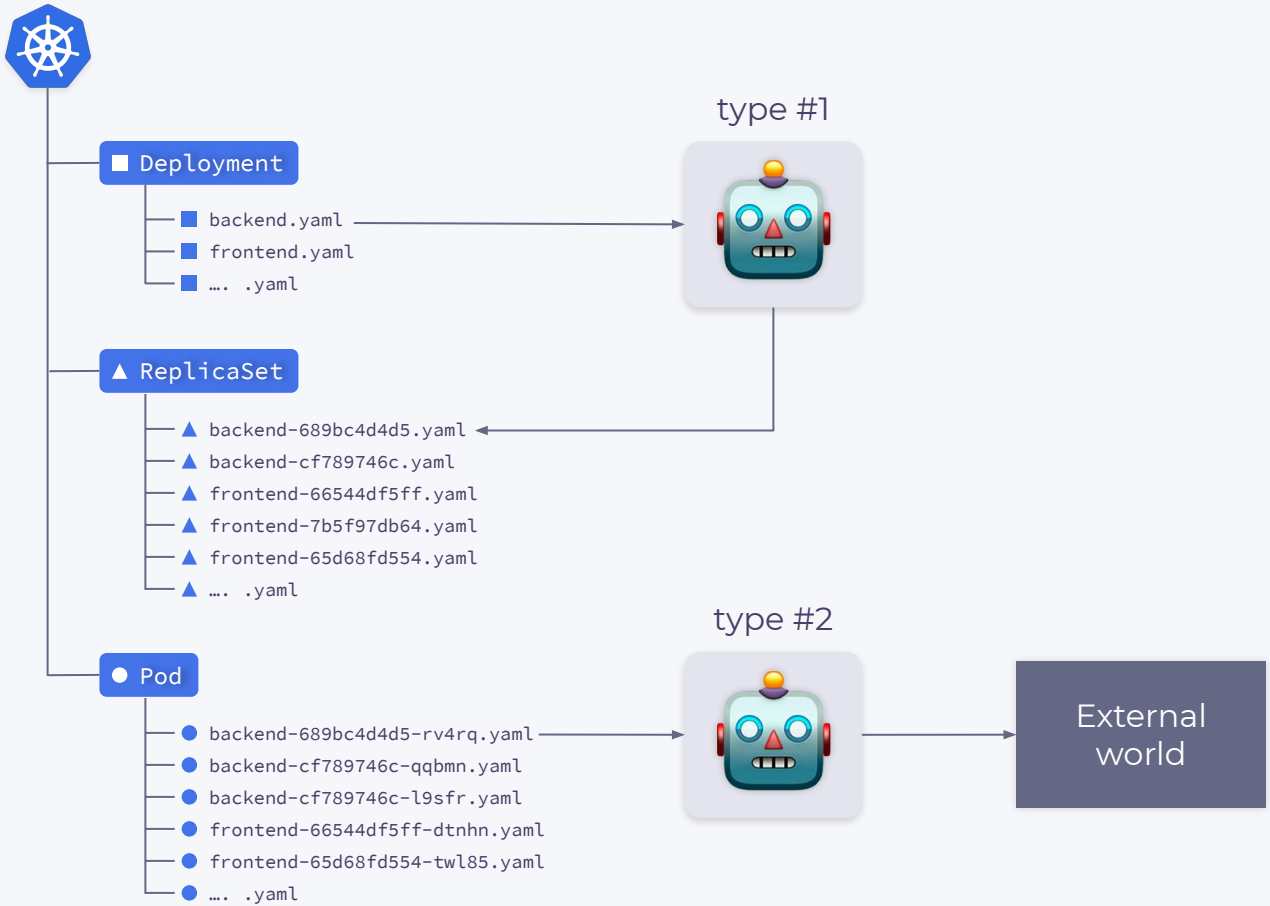


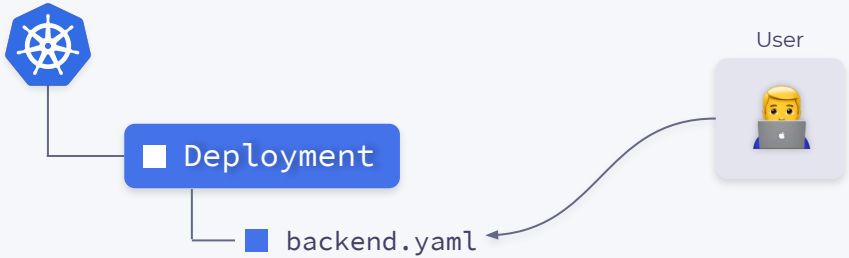
Simple HTTP API

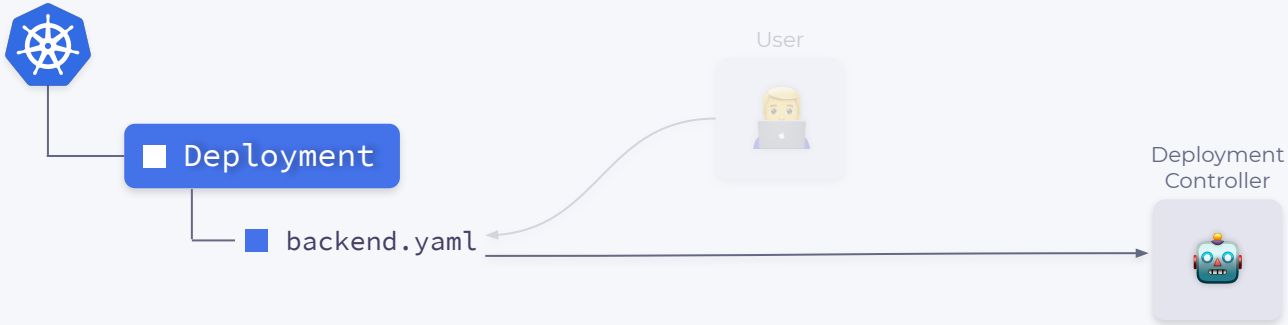


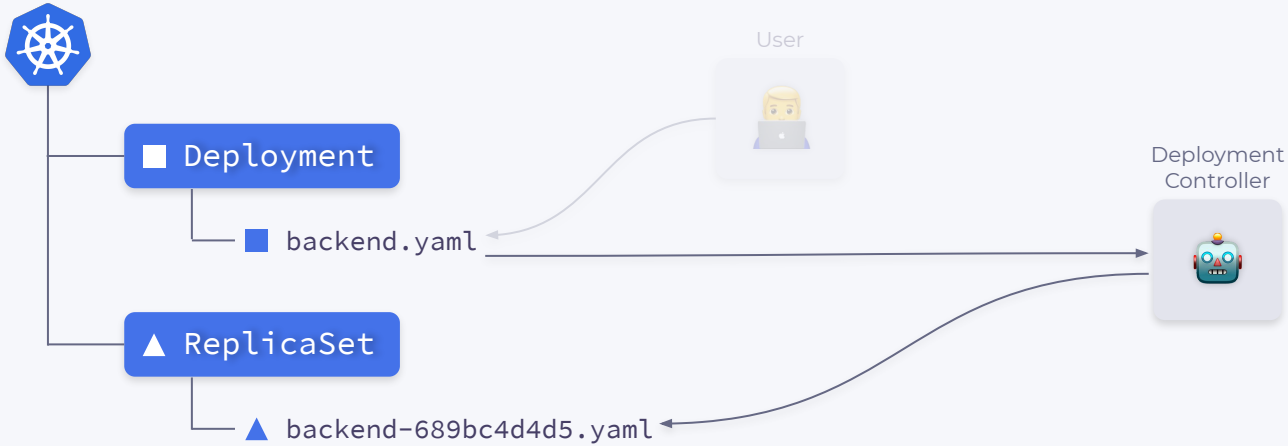
Simple HTTP API

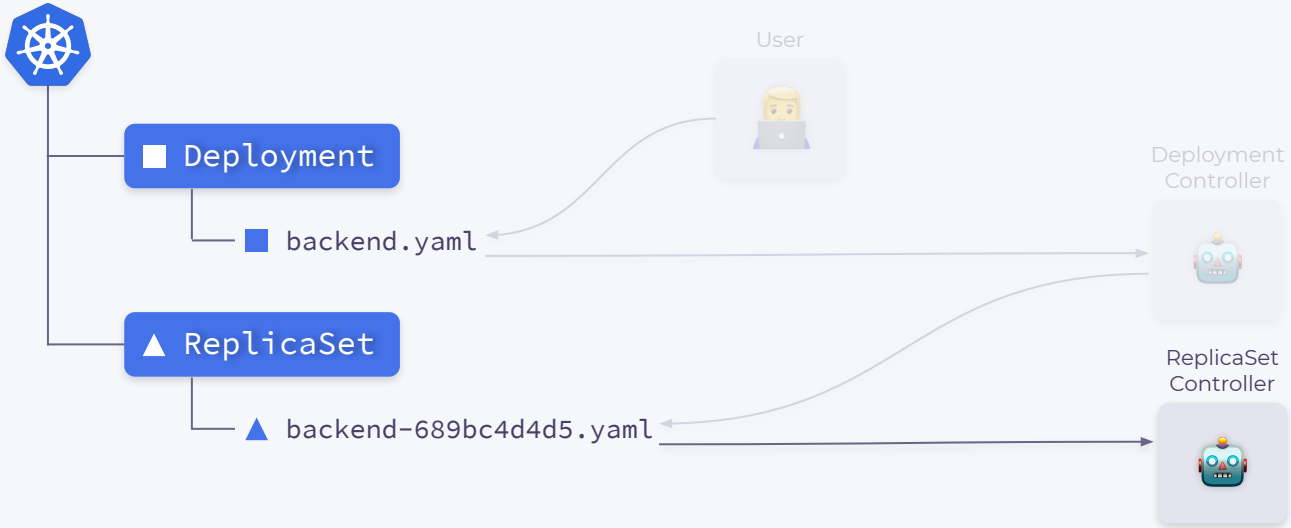


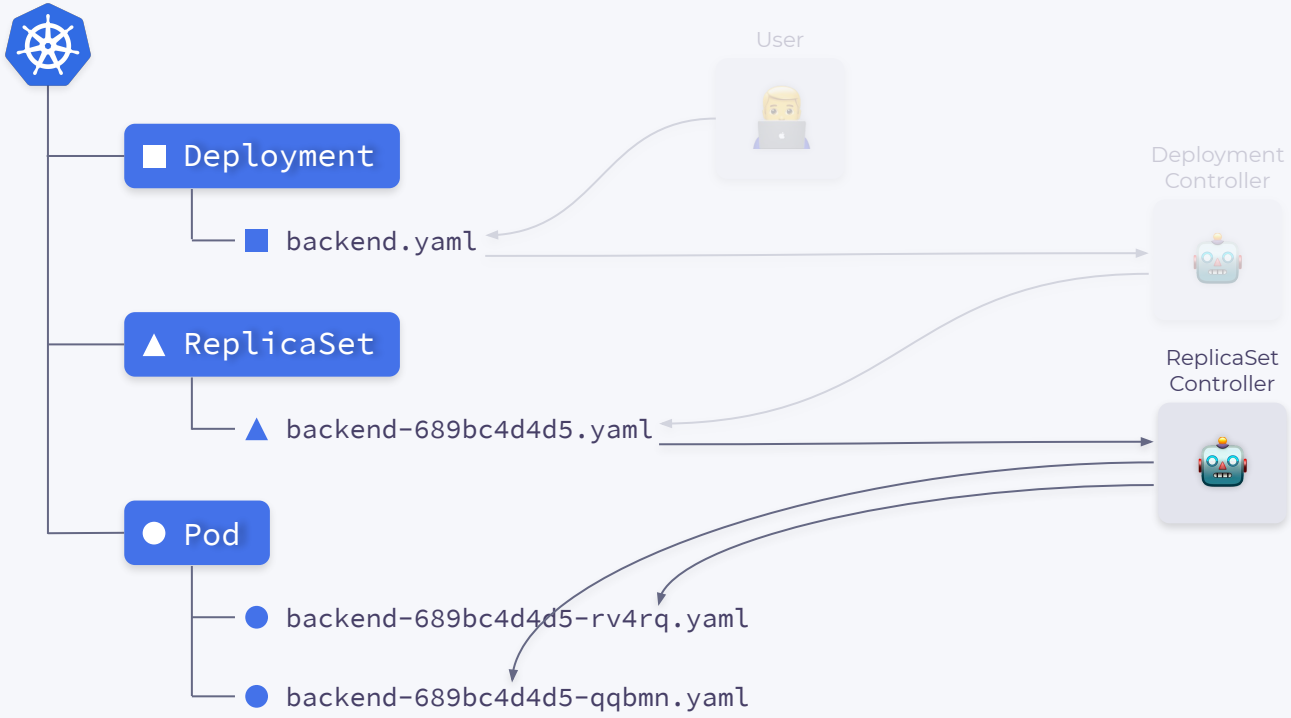

type #1
Deployment
backend.yaml
frontend.yaml
…. .yaml
ReplicaSet
backend-689bc4d4d5.yaml
backend-cf789746c.yaml
frontend-66544df5ff.yaml
frontend-7b5f97db64.yaml
frontend-65d68fd554.yaml
…. .yaml
type #2
Pod
backend-689bc4d4d5-rv4rq.yaml
backend-cf789746c-qqbmn.yaml
backend-cf789746c-l9sfr.yaml
frontend-66544df5ff-dtnhn.yaml
frontend-65d68fd554-twl85.yaml
…. .yaml
External world

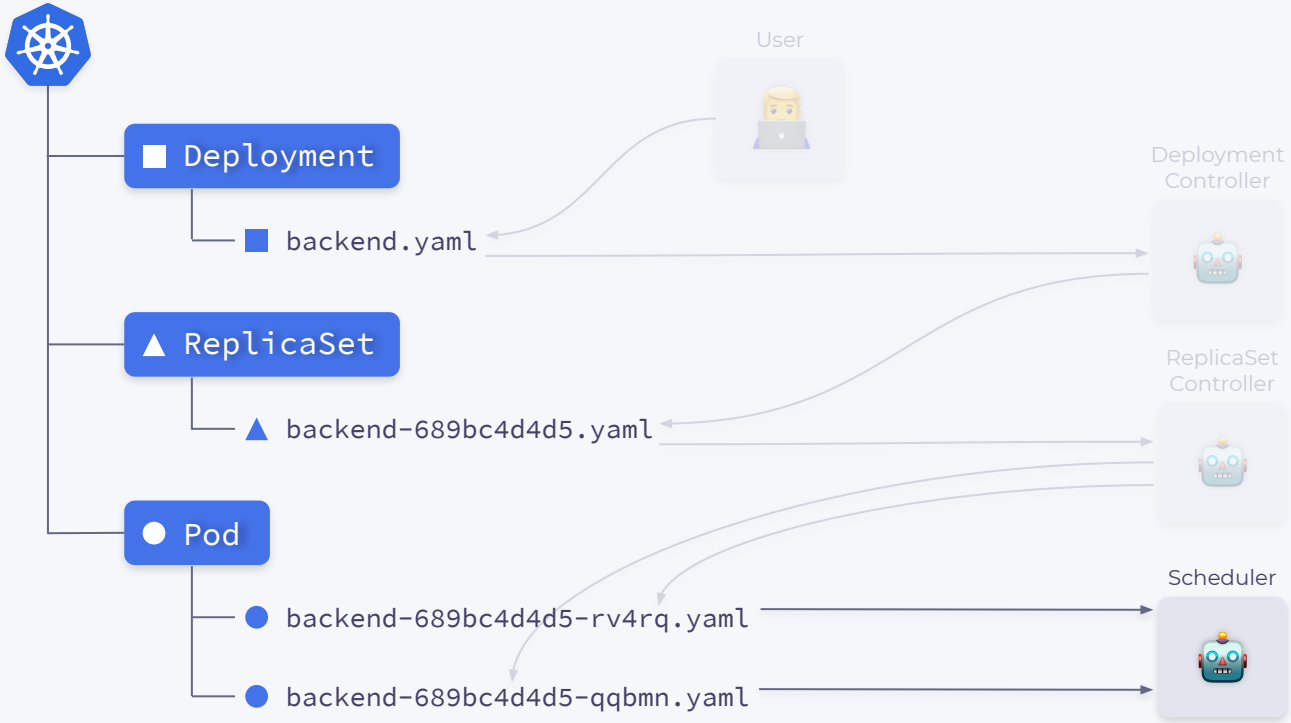

User
Deployment
backend.yaml

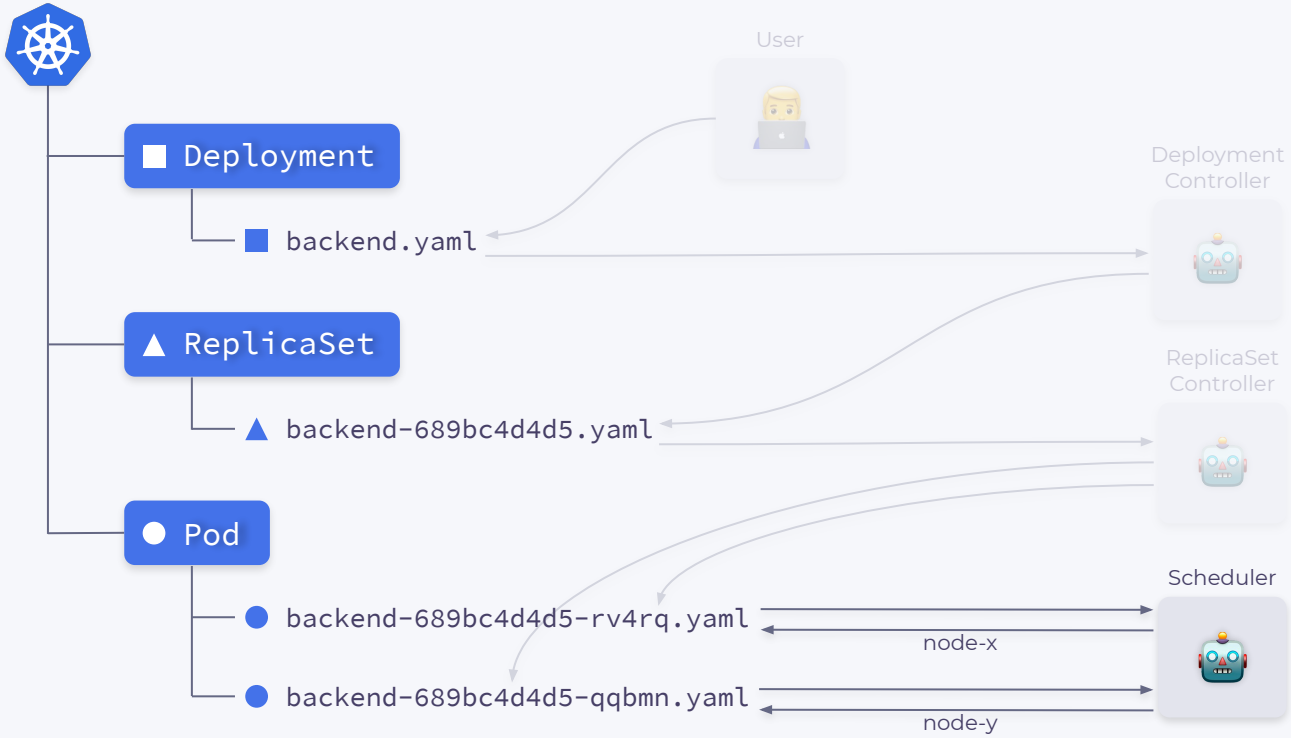

Deployment
backend.yaml
User
Deployment
Controller

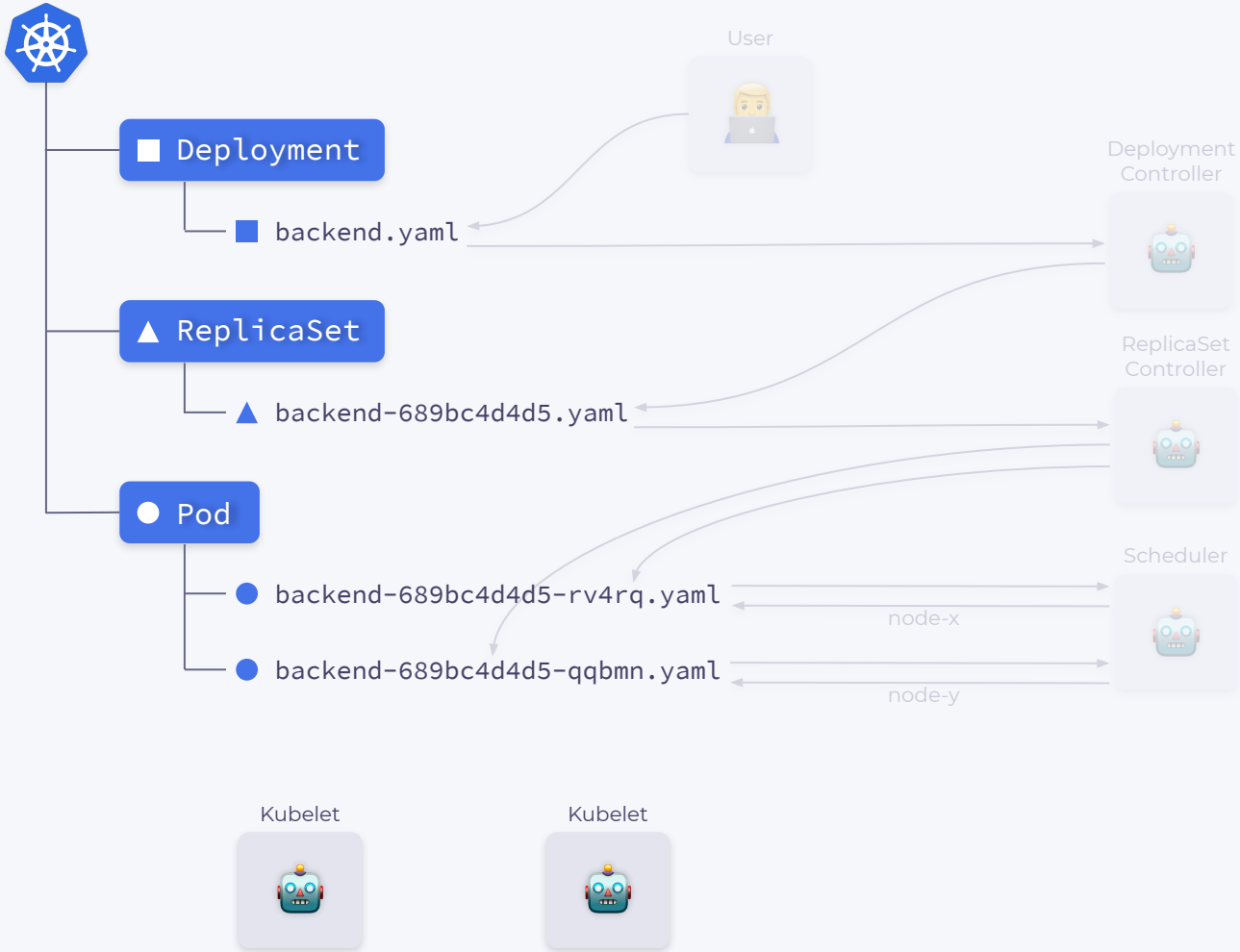

User
Deployment
backend.yaml
ReplicaSet
backend-689bc4d4d5.yaml
Deployment
Controller

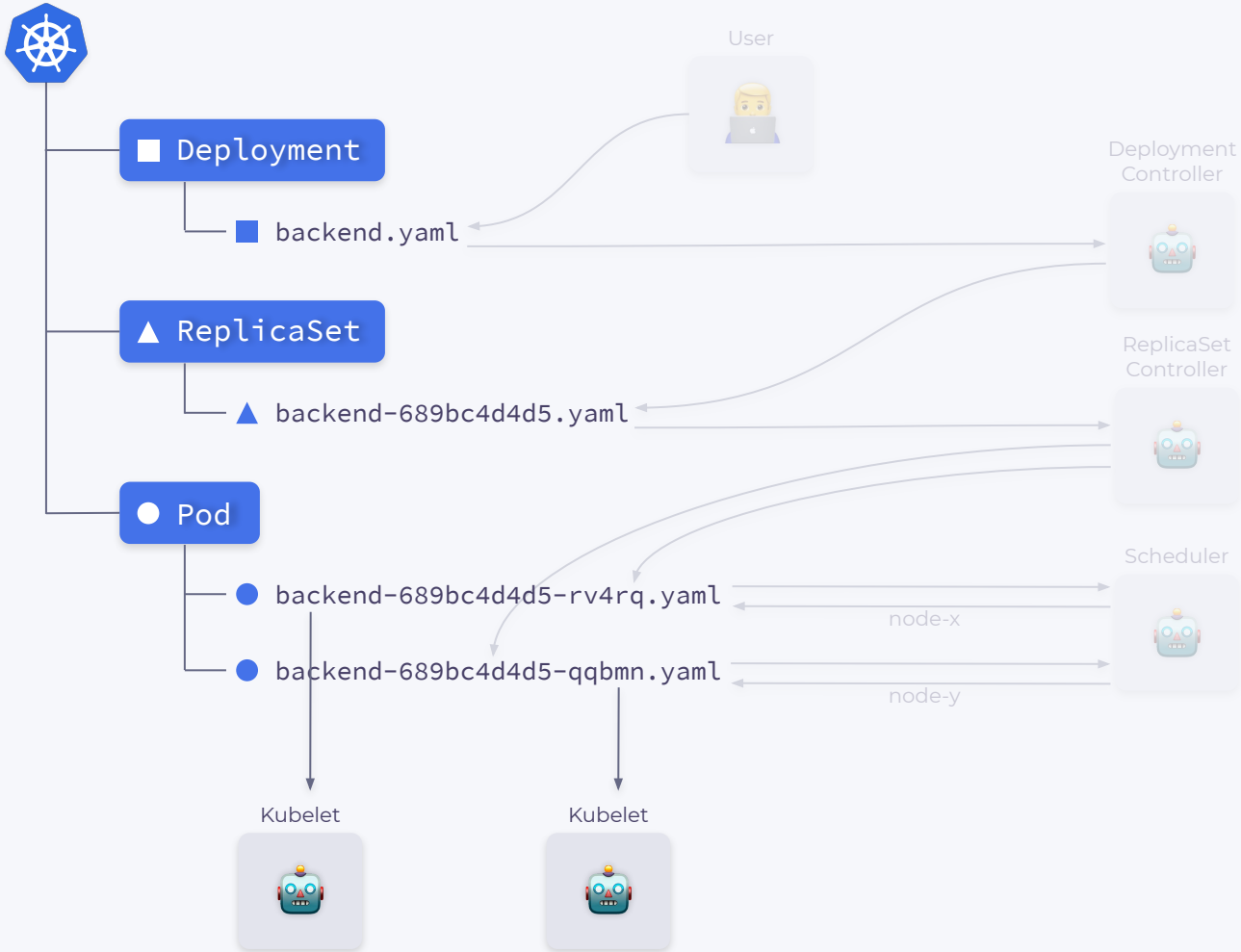

User
Deployment
backend.yaml
ReplicaSet
backend-689bc4d4d5.yaml
Deployment Controller
ReplicaSet Controller

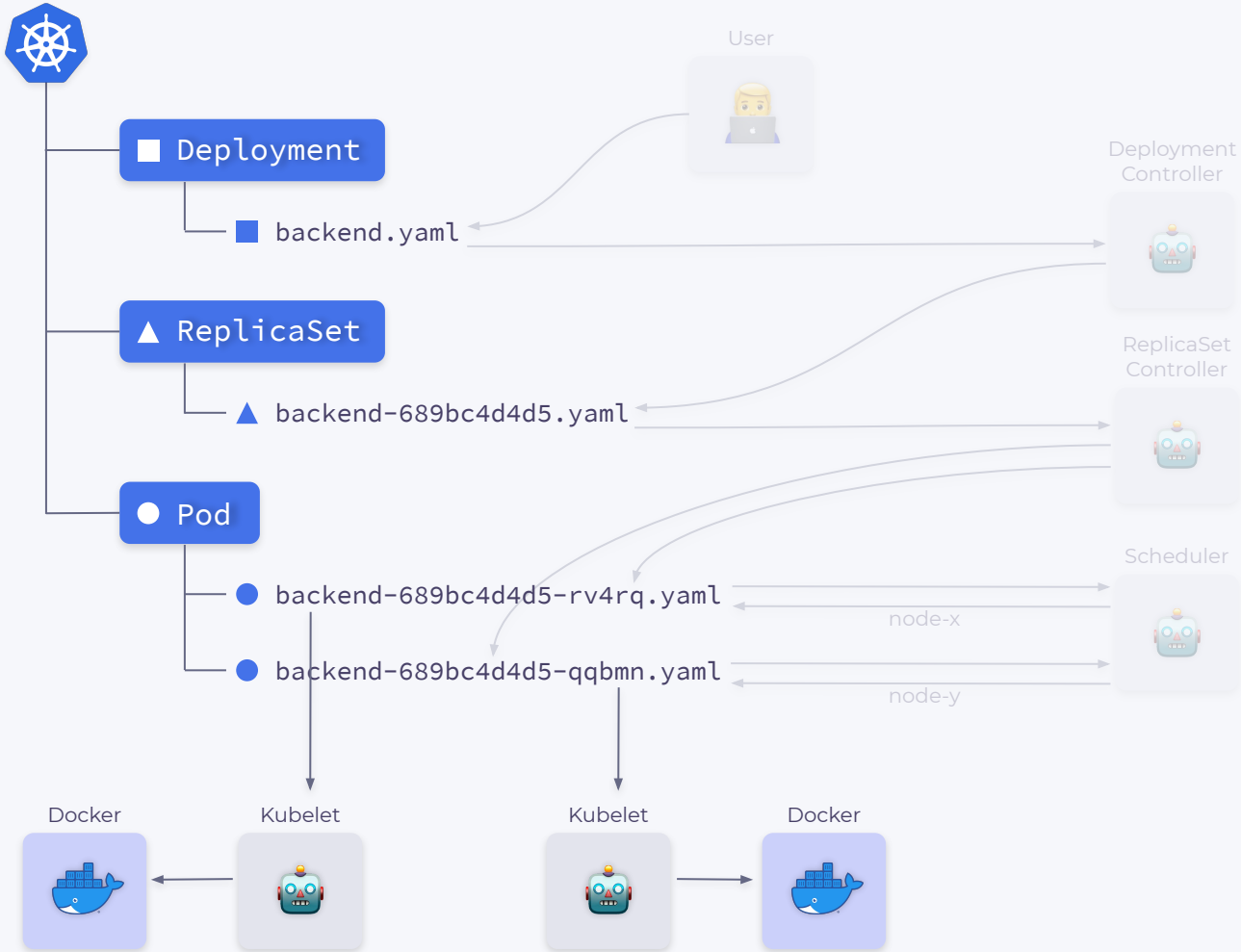

User
Deployment
backend.yaml
Deployment Controller
ReplicaSet
backend-689bc4d4d5.yaml
ReplicaSet Controller
Pod
backend-689bc4d4d5-rv4rq.yaml
backend-689bc4d4d5-qqbmn.yaml

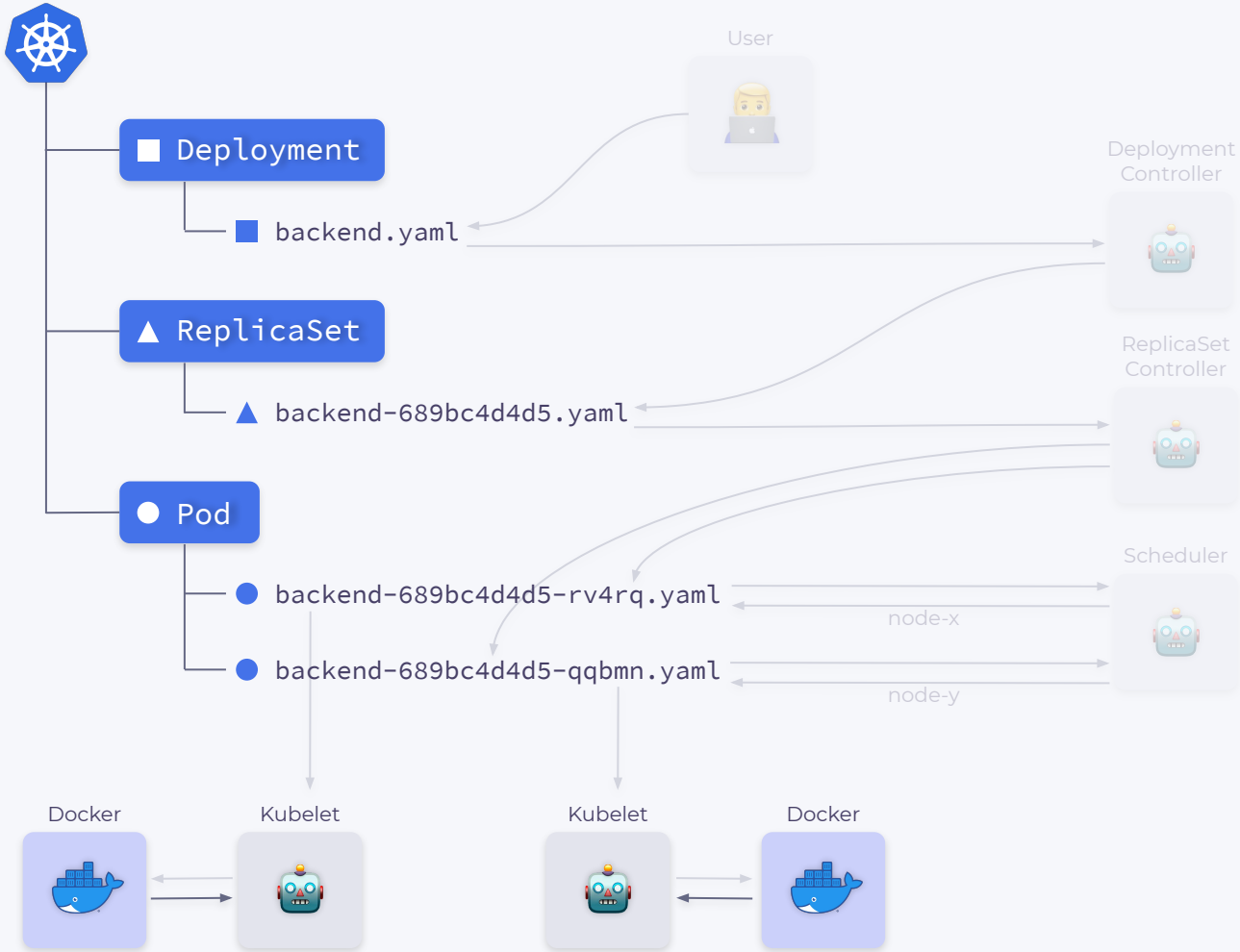

User
Deployment
backend.yaml
Deployment Controller
ReplicaSet
backend-689bc4d4d5.yaml
ReplicaSet Controller
Pod
Scheduler
backend-689bc4d4d5-rv4rq.yaml
backend-689bc4d4d5-qqbmn.yaml

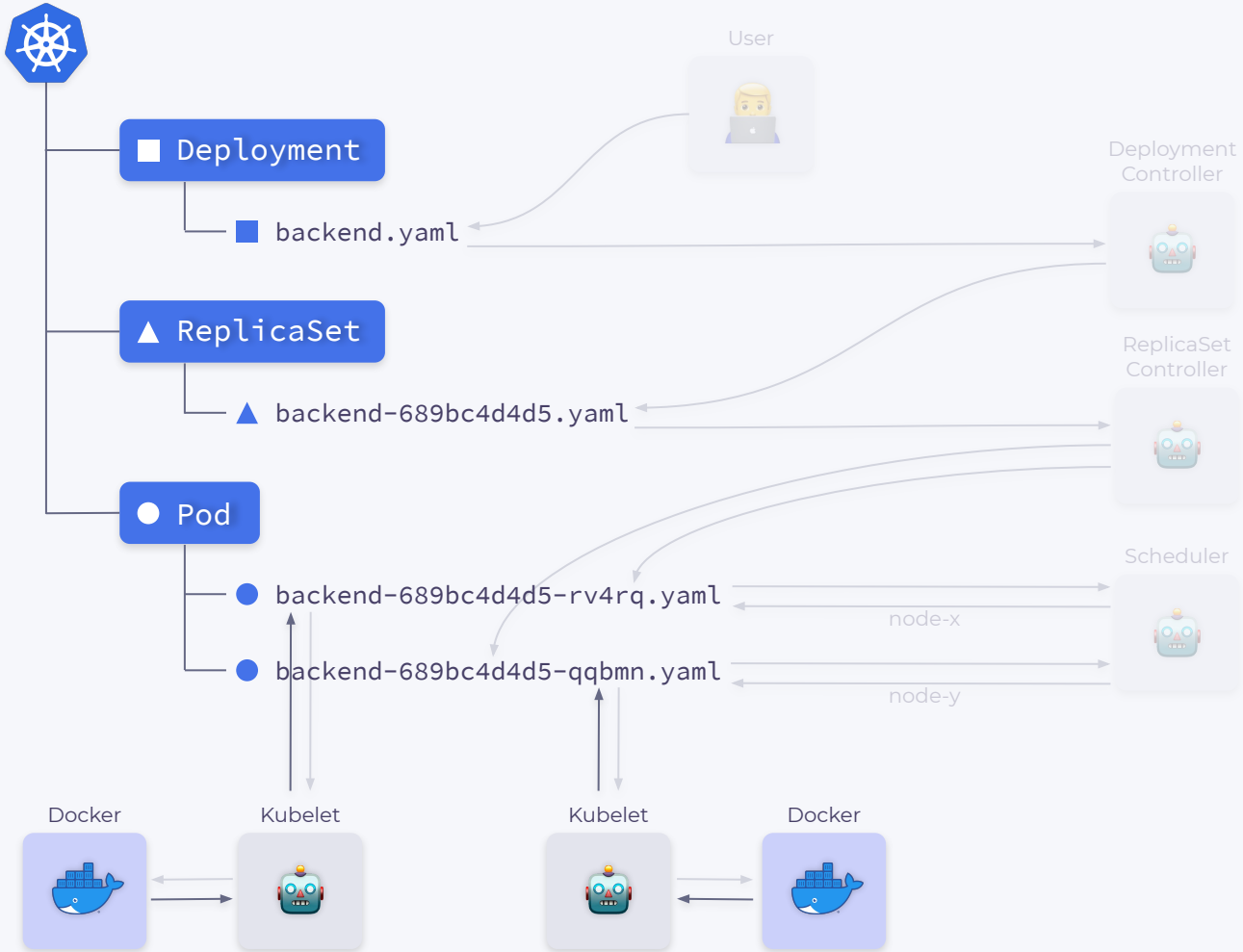

User
Deployment
backend.yaml
Deployment Controller
ReplicaSet
backend-689bc4d4d5.yaml
ReplicaSet Controller
Pod
Scheduler
backend-689bc4d4d5-rv4rq.yaml
node-x
backend-689bc4d4d5-qqbmn.yaml
node-y

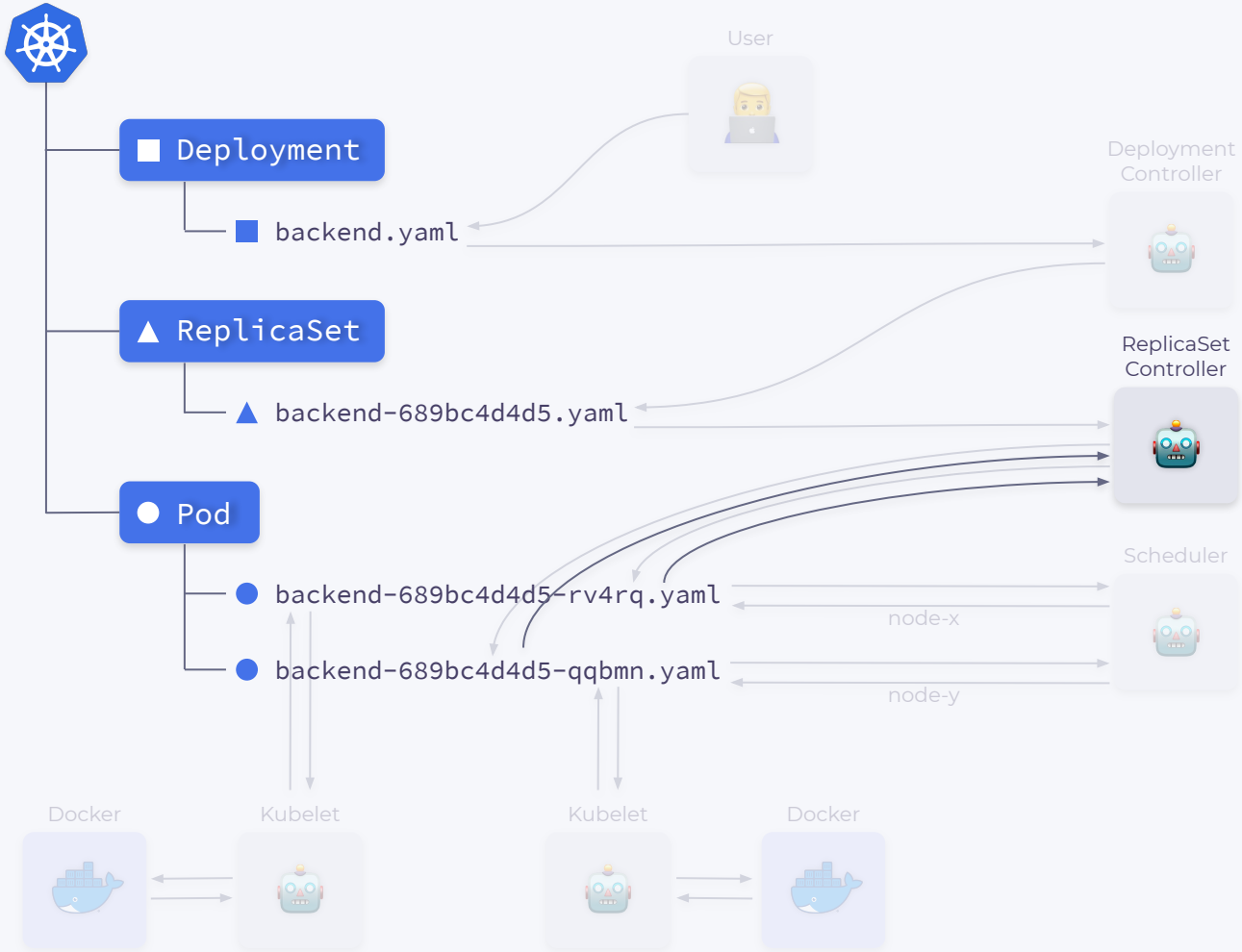

User
Deployment
backend.yaml
Deployment Controller
ReplicaSet
backend-689bc4d4d5.yaml
ReplicaSet Controller
Pod
backend-689bc4d4d5-rv4rq.yaml
Scheduler
node-x
backend-689bc4d4d5-qqbmn.yaml
node-y
Kubelet
Kubelet

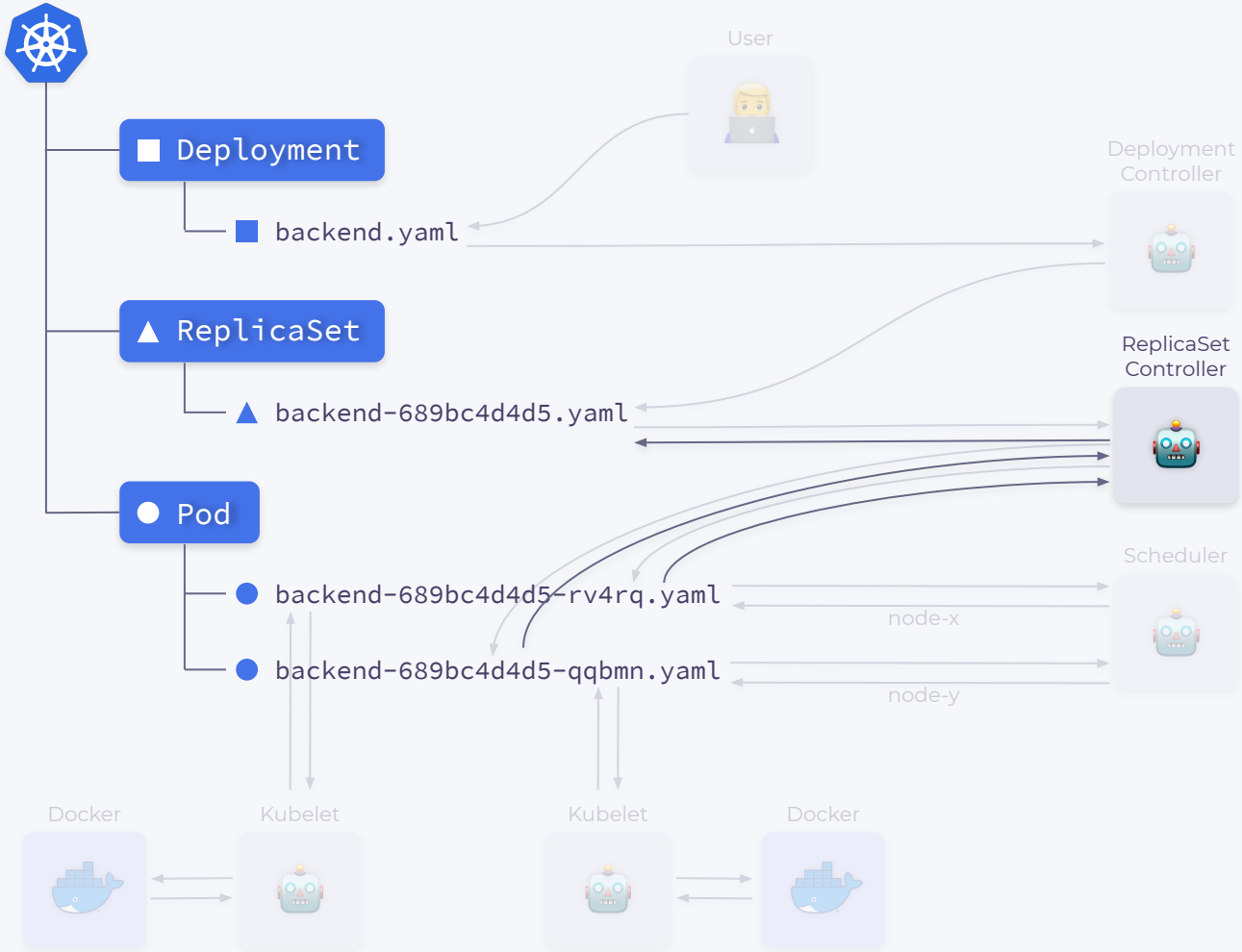

User
Deployment
backend.yaml
Deployment Controller
ReplicaSet
backend-689bc4d4d5.yaml
ReplicaSet Controller
Pod
backend-689bc4d4d5-rv4rq.yaml
backend-689bc4d4d5-qqbmn.yaml
Scheduler
node-x
node-y
Kubelet
Kubelet

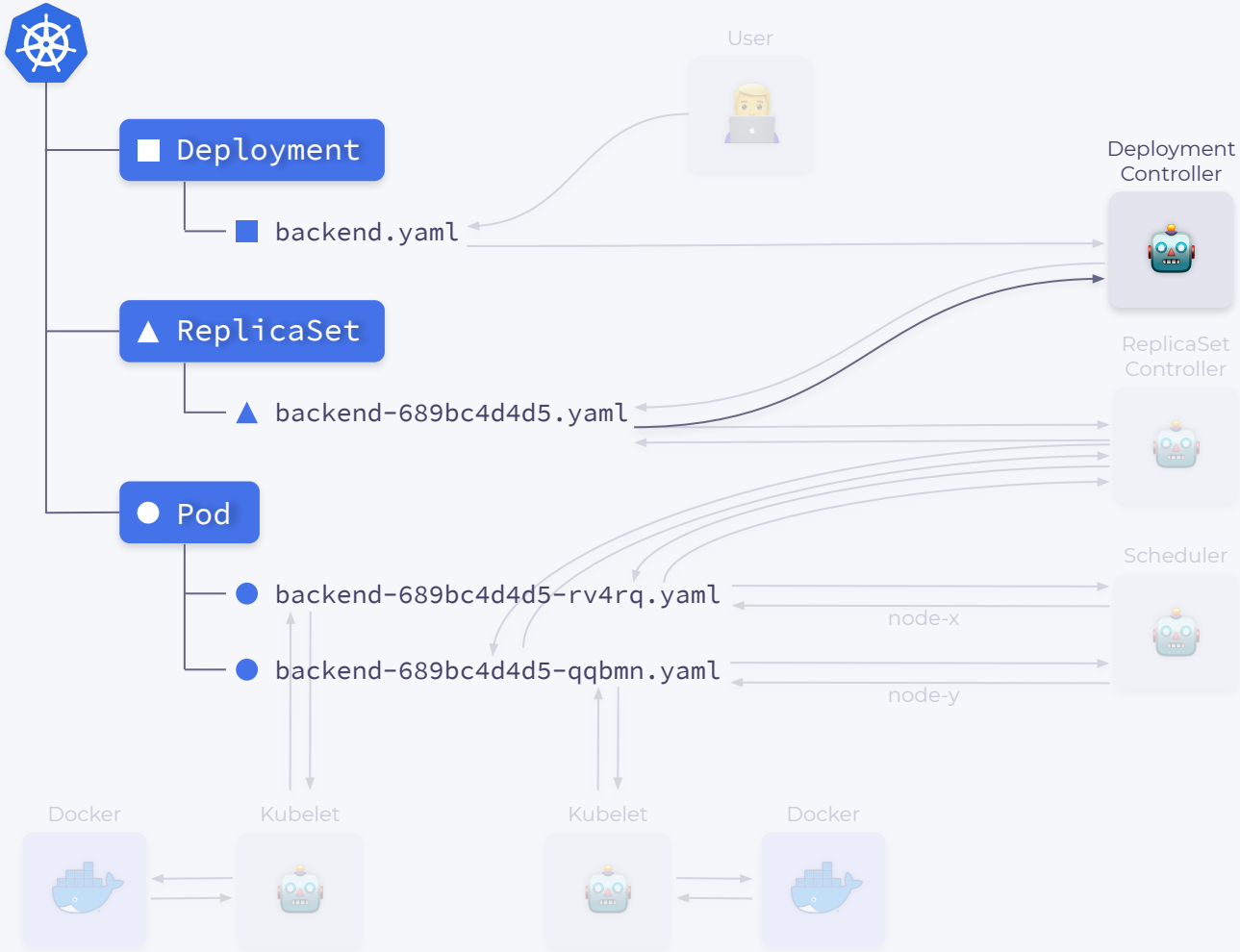

User
Deployment
backend.yaml
Deployment Controller
ReplicaSet
backend-689bc4d4d5.yaml
ReplicaSet Controller
Pod
backend-689bc4d4d5-rv4rq.yaml
backend-689bc4d4d5-qqbmn.yaml
Scheduler
node-x
node-y
Docker
Kubelet
Kubelet
Docker

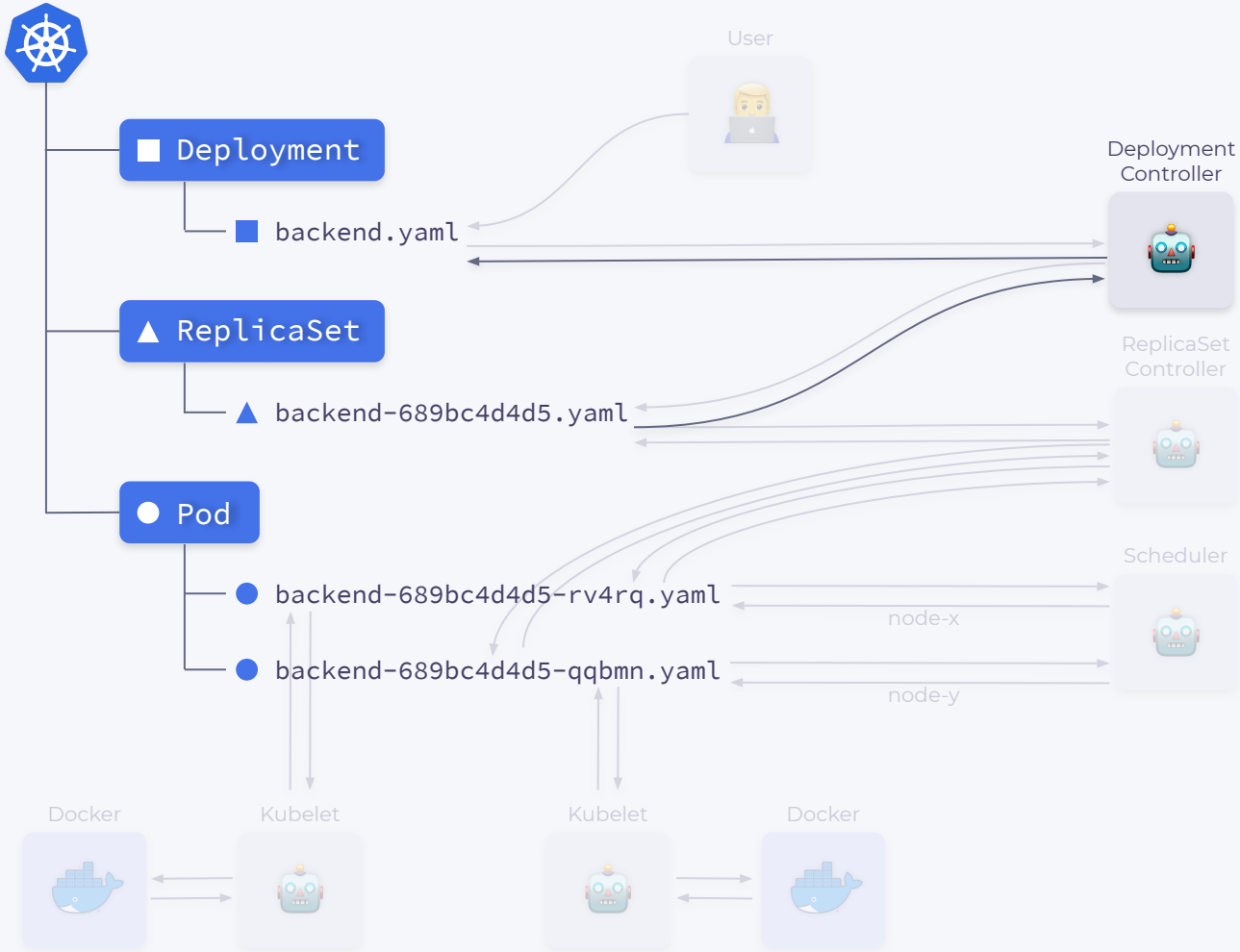

User
Deployment
backend.yaml
Deployment Controller
ReplicaSet
backend-689bc4d4d5.yaml
ReplicaSet Controller
Pod
backend-689bc4d4d5-rv4rq.yaml
Scheduler
node-x
backend-689bc4d4d5-qqbmn.yaml
node-y
Docker
Kubelet
Kubelet
Docker

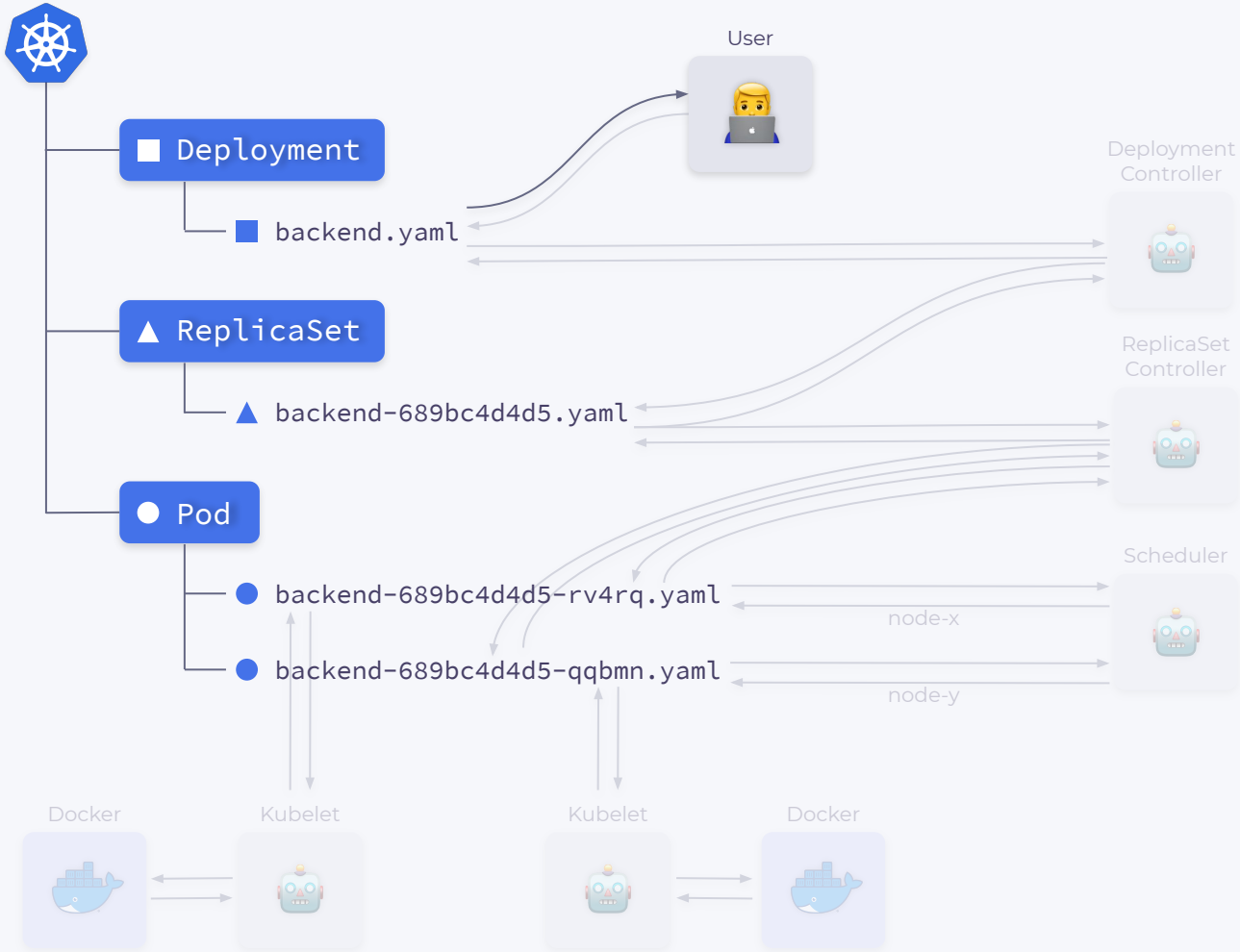

User
Deployment
backend.yaml
Deployment Controller
ReplicaSet
backend-689bc4d4d5.yaml
ReplicaSet Controller
Pod
backend-689bc4d4d5-rv4rq.yaml
Scheduler
node-x
backend-689bc4d4d5-qqbmn.yaml
node-y
Docker
Kubelet
Kubelet
Docker


User
Deployment
backend.yaml
Deployment Controller
ReplicaSet
backend-689bc4d4d5.yaml
ReplicaSet Controller
Pod
backend-689bc4d4d5-rv4rq.yaml
backend-689bc4d4d5-qqbmn.yaml
Scheduler
node-x
node-y
Docker
Kubelet
Kubelet
Docker


User
Deployment
backend.yaml
Deployment Controller
ReplicaSet
backend-689bc4d4d5.yaml
ReplicaSet Controller
Pod
backend-689bc4d4d5-rv4rq.yaml
backend-689bc4d4d5-qqbmn.yaml
Scheduler
node-x
node-y
Docker
Kubelet
Kubelet
Docker

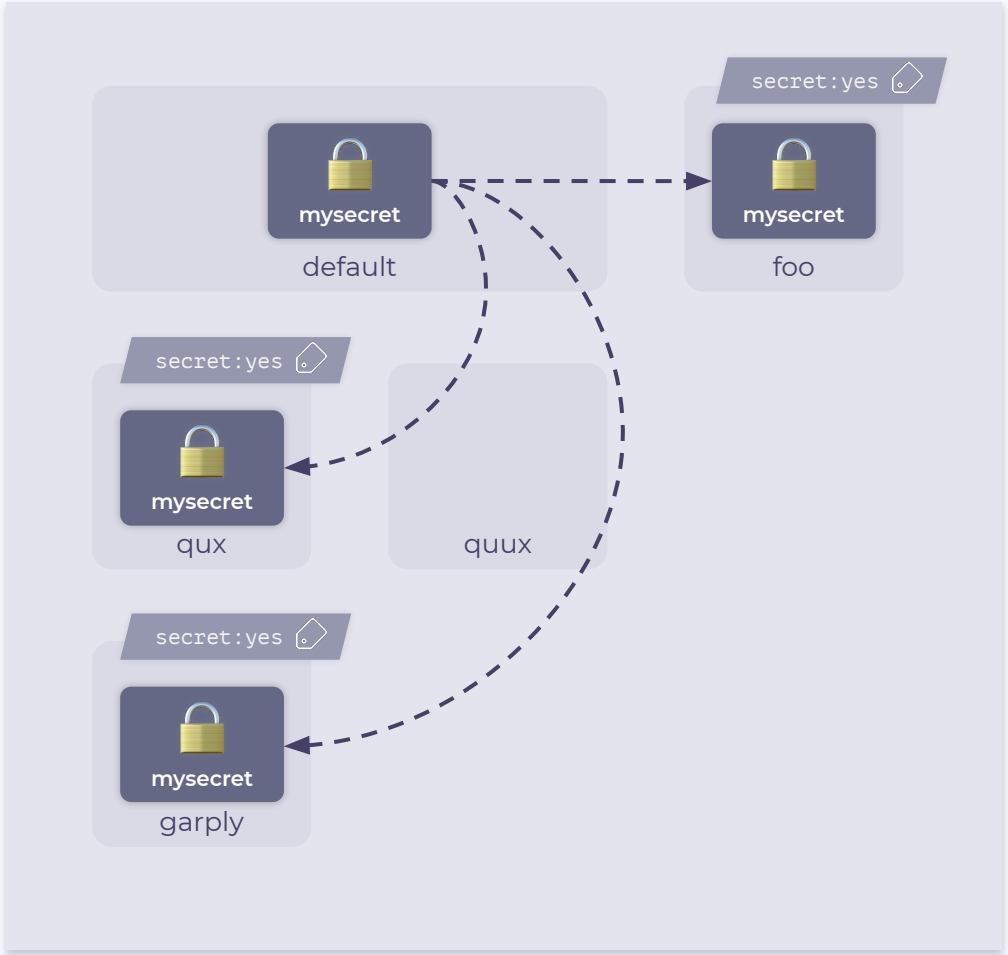

User
Deployment
backend.yaml
Deployment Controller
ReplicaSet
backend-689bc4d4d5.yaml
ReplicaSet Controller
Pod
backend-689bc4d4d5-rv4rq.yaml
Scheduler
node-x
backend-689bc4d4d5-qqbmn.yaml
node-y
Docker
Kubelet
Kubelet
Docker


User
Deployment
backend.yaml
Deployment Controller
ReplicaSet
backend-689bc4d4d5.yaml
ReplicaSet Controller
Pod
backend-689bc4d4d5-rv4rq.yaml
Scheduler
node-x
backend-689bc4d4d5-qqbmn.yaml
node-y
Docker
Kubelet
Kubelet
Docker


User
Deployment
backend.yaml
Deployment Controller
ReplicaSet
backend-689bc4d4d5.yaml
ReplicaSet Controller
Pod
backend-689bc4d4d5-rv4rq.yaml
backend-689bc4d4d5-qqbmn.yaml
Scheduler
node-x
node-y
Docker
Kubelet
Kubelet
Docker


kubernetes
mysecret
default
secret:yes
foo
secret:yes
qux
quux
secret:yes
garply

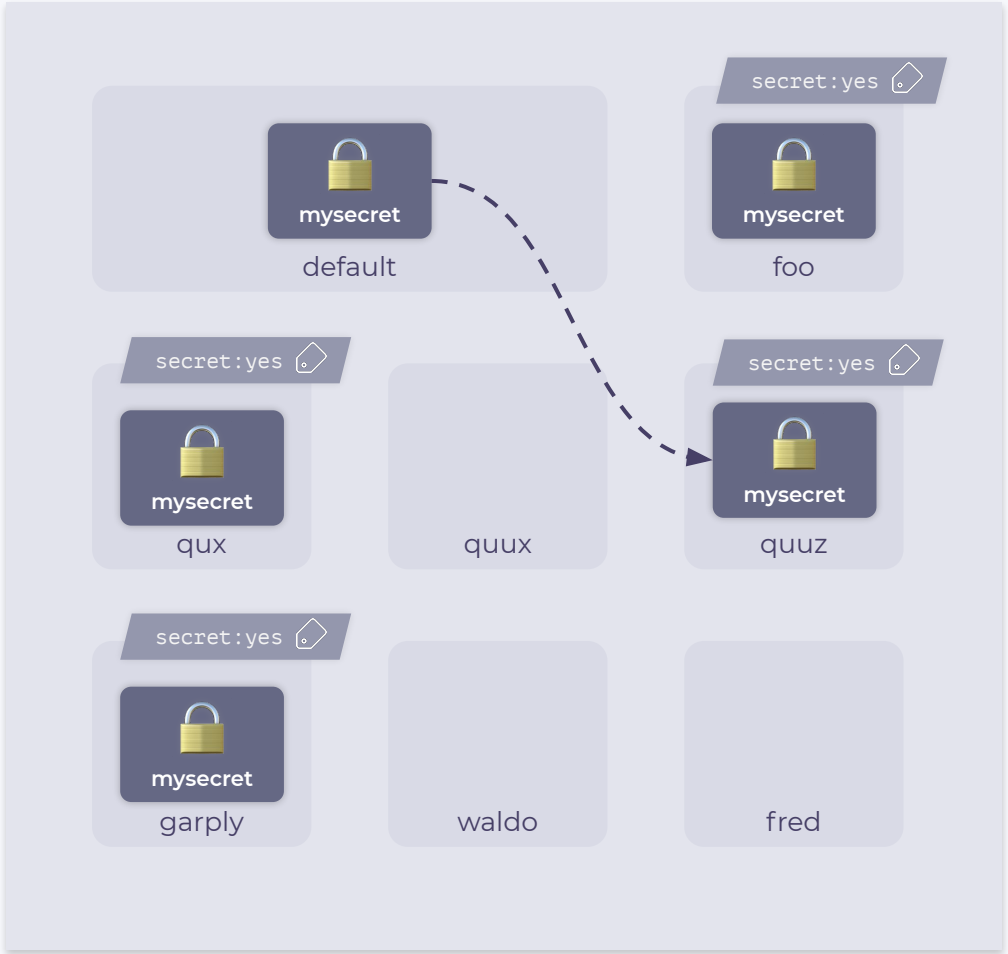
kubernetes

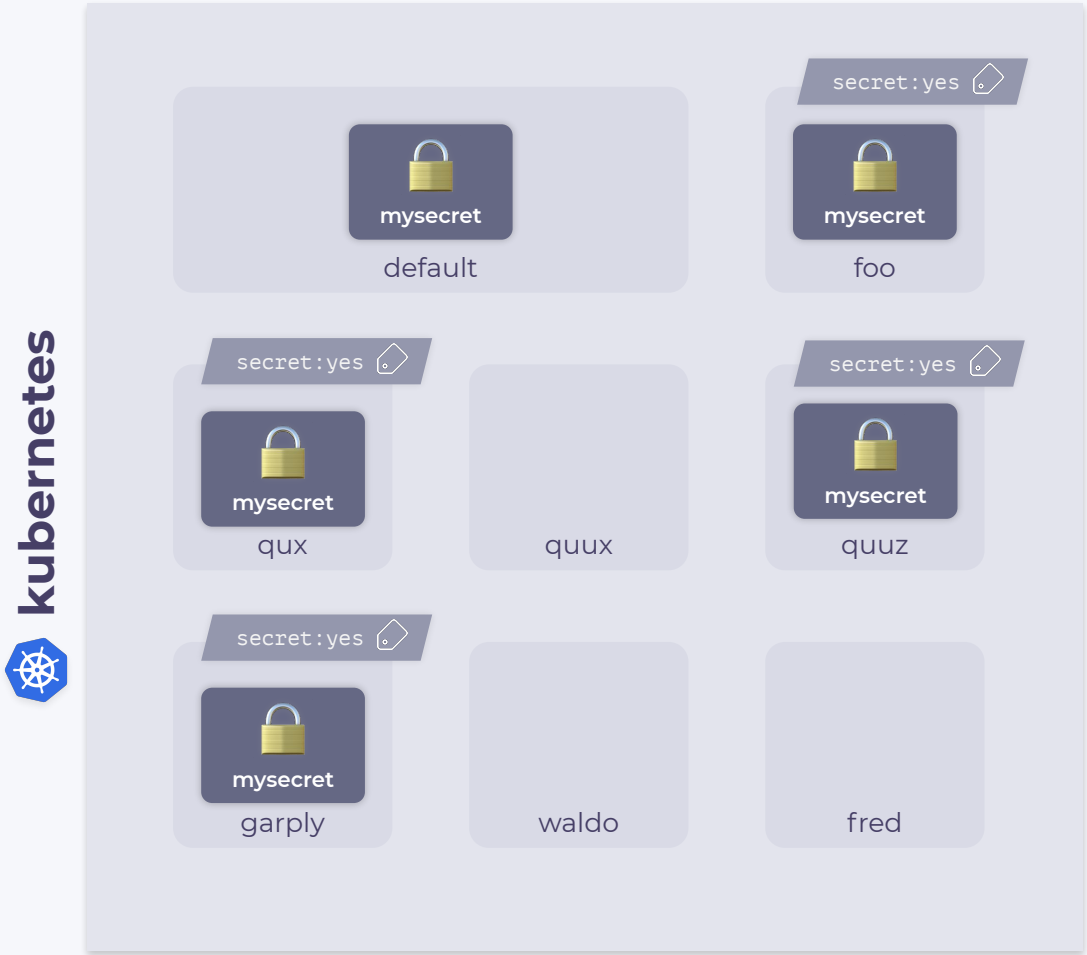

secret:yes
mysecret
default
mysecret
foo
secret:yes
mysecret
qux
quux
secret:yes
mysecret
garply

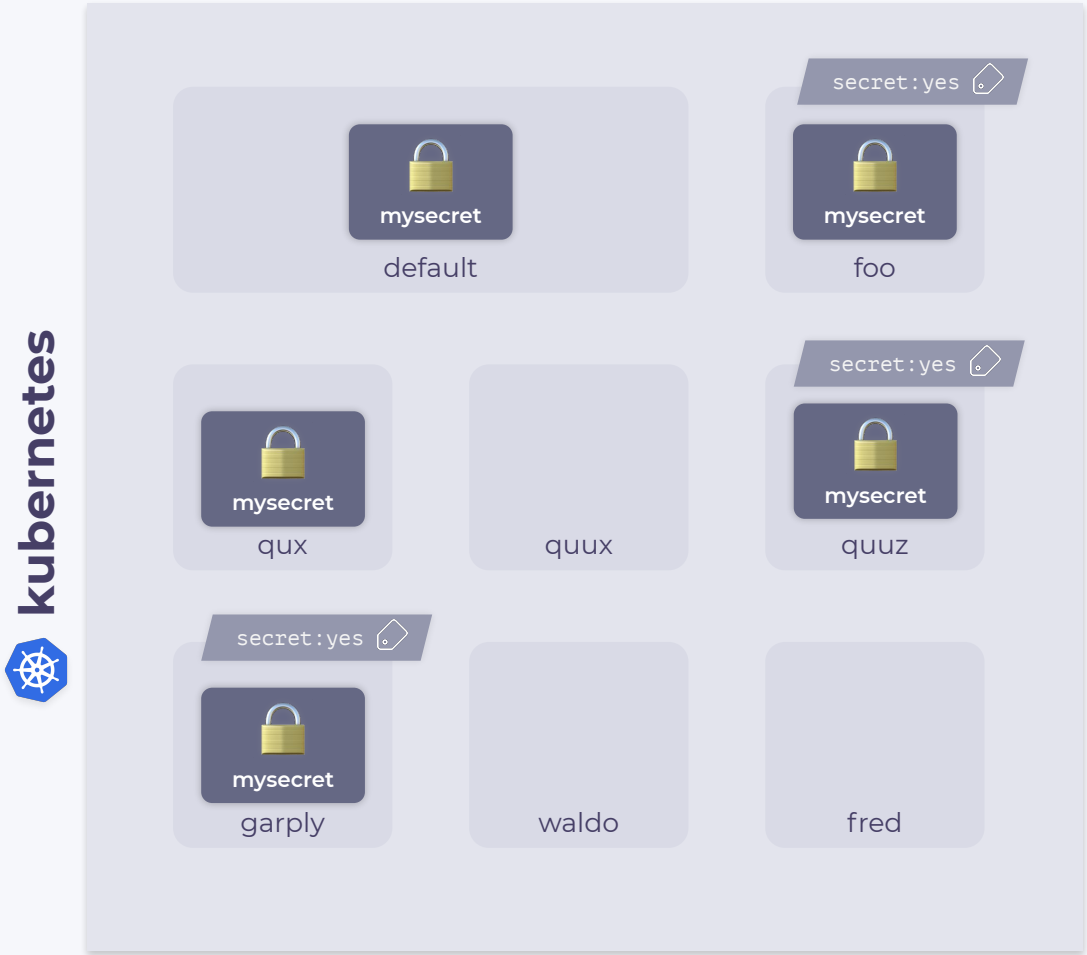

kubernetes
mysecret
default
secret:yes
mysecret
foo
secret:yes
mysecret
qux
quux
quuz
secret:yes
mysecret
garply
waldo
fred


kubernetes
mysecret
default
secret:yes
mysecret
foo
secret:yes
mysecret
qux
quux
secret:yes
quuz
secret:yes
mysecret
garply
waldo
fred

kubernetes


secret:yes
mysecret
default
mysecret
foo
secret:yes
mysecret
qux
quux
secret:yes
mysecret
quuz
secret:yes
mysecret
garply
waldo
fred

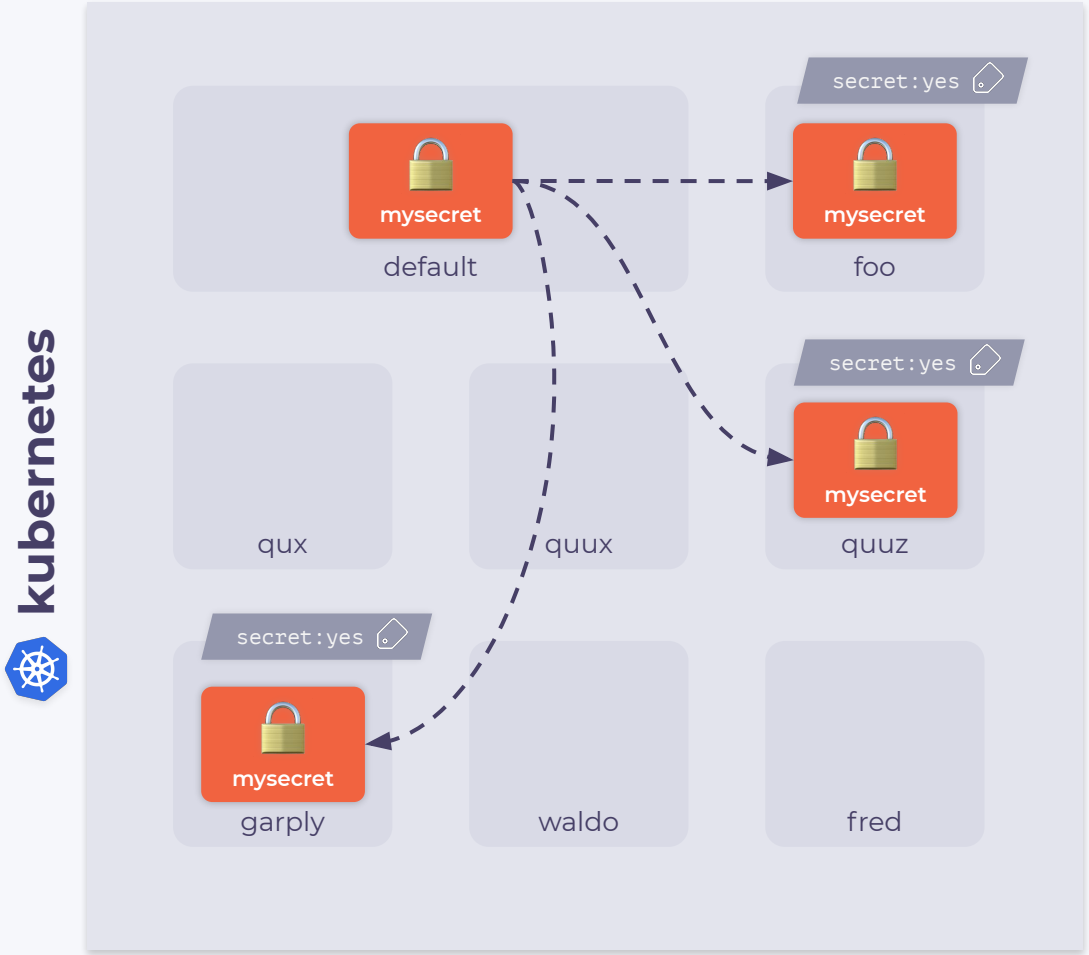

kubernetes
mysecret
default
secret:yes
mysecret
foo
secret:yes
mysecret
qux
quux
secret:yes
mysecret
quuz
secret:yes
mysecret
garply
waldo
fred


kubernetes
mysecret
default
secret:yes
mysecret
foo
mysecret
qux
quux
secret:yes
mysecret
quuz
secret:yes
mysecret
garply
waldo
fred


kubernetes
mysecret
default
secret:yes
mysecret
foo
qux
quux
secret:yes
mysecret
quuz
secret:yes
mysecret
garply
waldo
fred


kubernetes
mysecret
default
secret:yes
mysecret
foo
qux
quux
secret:yes
mysecret
quuz
secret:yes
mysecret
garply
waldo
fred

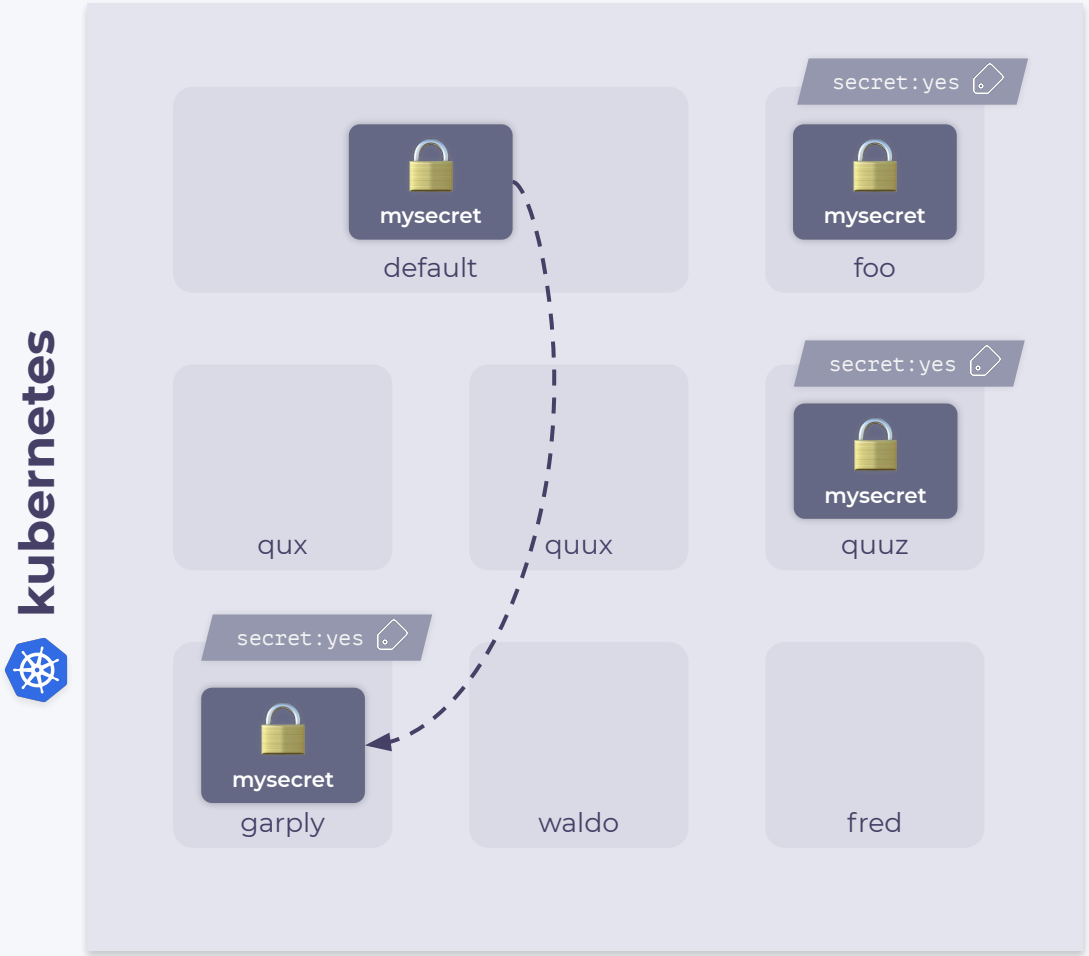

kubernetes
mysecret
default
secret:yes
mysecret
foo
qux
quux
secret:yes
mysecret
quuz
secret:yes
mysecret
garply
waldo
fred


kubernetes
secret:yes
mysecret
default
mysecret
foo
secret:yes
mysecret
quux
quuz
qux
secret:yes
mysecret
garply
waldo
fred


kubernetes
mysecret
default
secret:yes
mysecret
foo
qux
quux
secret:yes
mysecret
quuz
secret:yes
mysecret
garply
waldo
fred


kubernetes
secret:yes
mysecret
default
secret:yes
mysecret
foo
qux
quux
secret:yes
mysecret
quuz
secret:yes
garply
waldo
fred

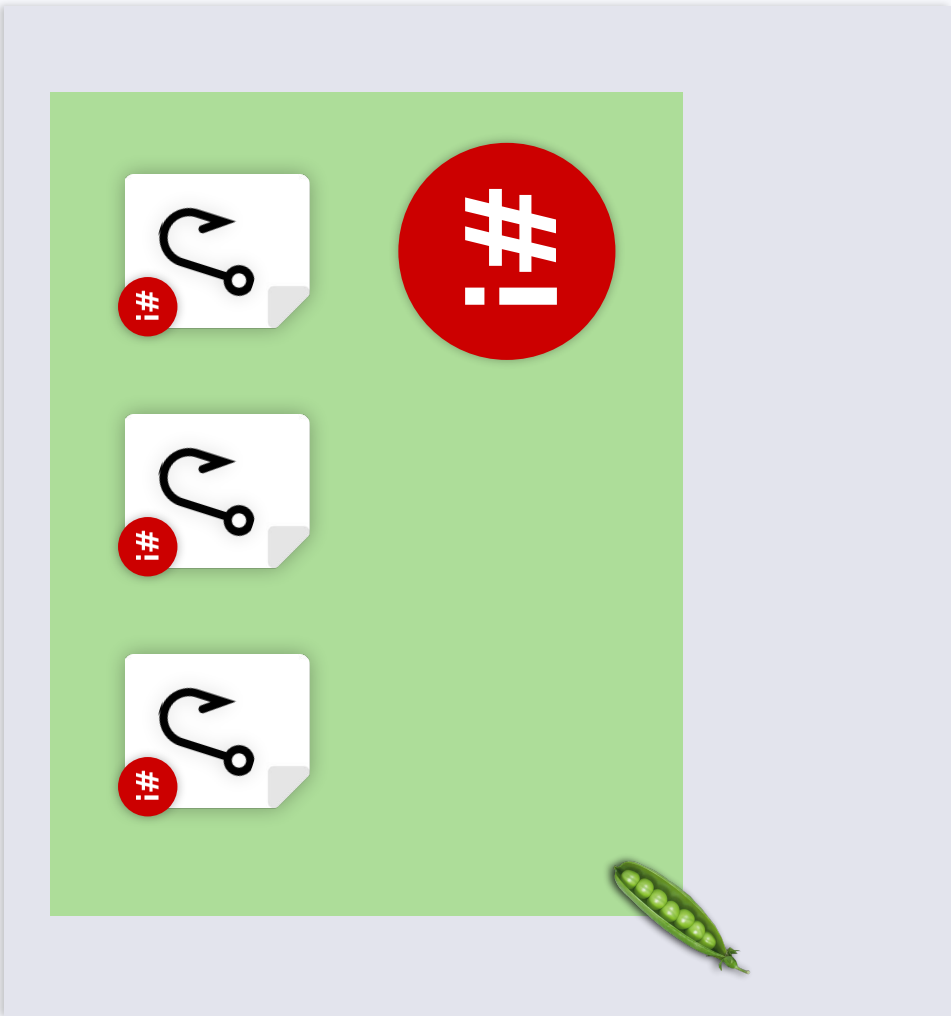

kubernetes
secret:yes
mysecret
default
mysecret
foo
qux
quux
secret:yes
mysecret
quuz
secret:yes
garply
waldo
fred


kubernetes
mysecret
default
secret:yes
mysecret
foo
qux
quux
secret:yes
mysecret
quuz
secret:yes
mysecret
garply
waldo
fred

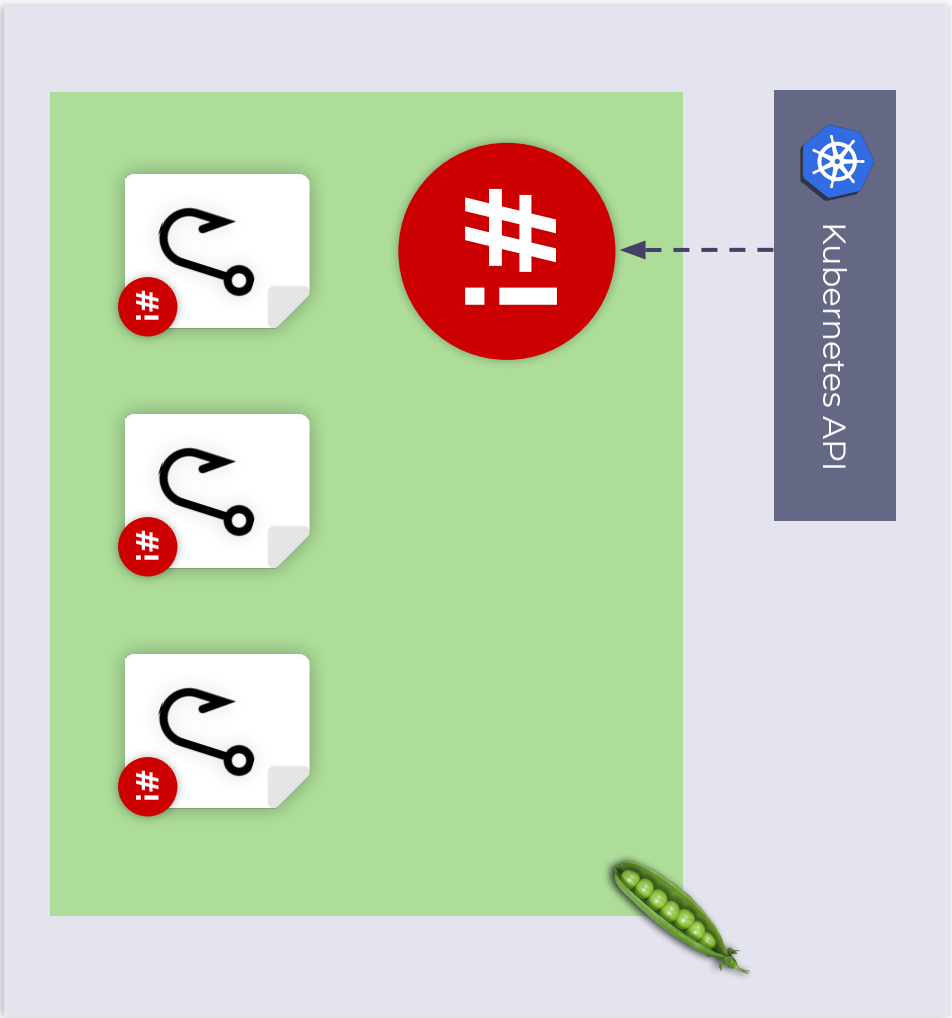
shell-operator


#!
/hooks/
foo.sh
bar.py
baz.rb

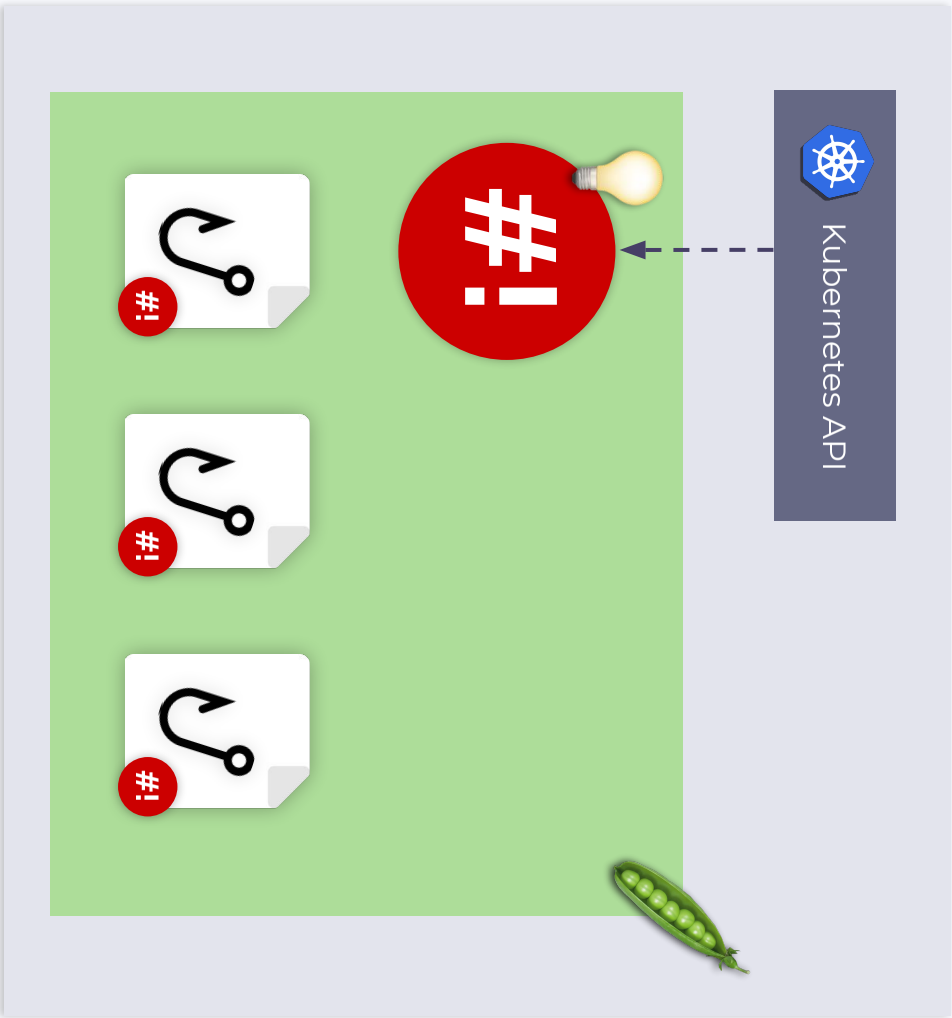
kubernetes

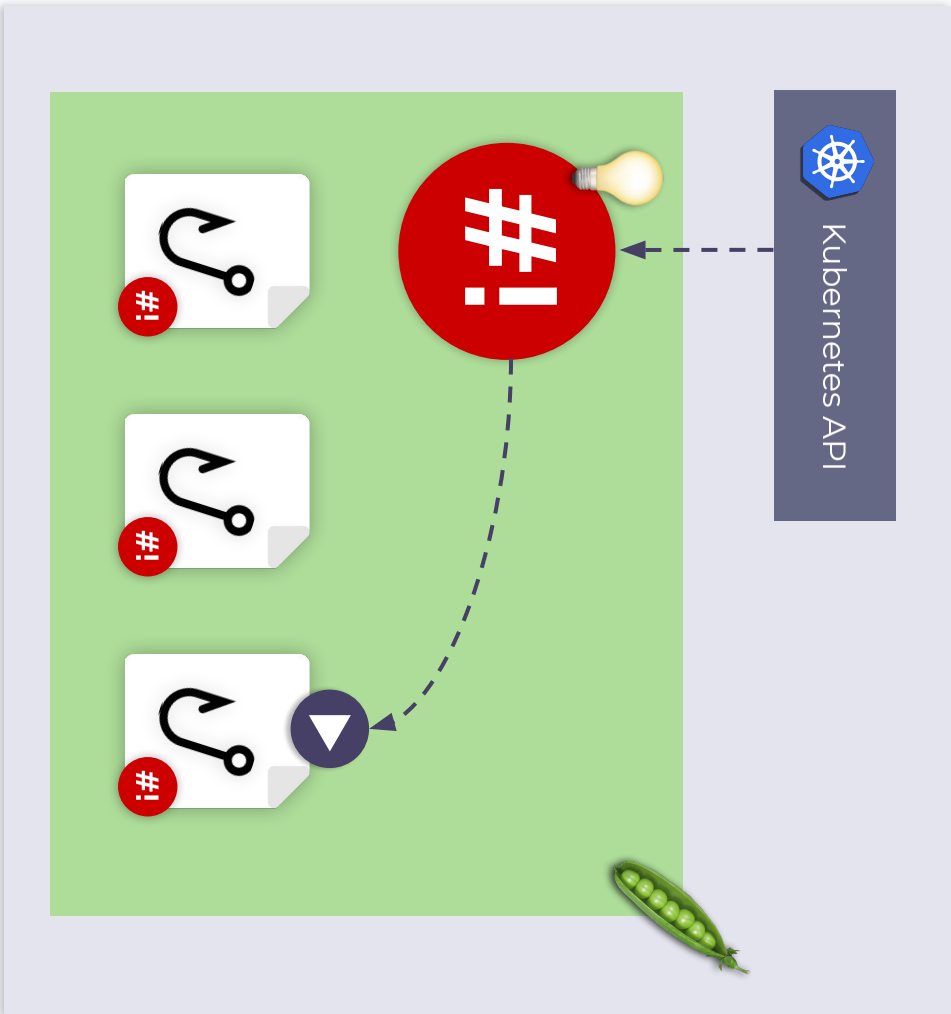


kubernetes



kubernetes




kubernetes
Kubernetes API

Configuration

once, on startup

1. Hook is executed with `--config` arg.
2. Hook should output YAML with its configuration.

Normal execution

many times, on events

1. Hook is executed with **no args**.
2. Hook receives **binding context**.

```
#!/bin/bash

source /shell_lib.sh

function __config__() {



}

function __main__() {

}

hook::run "$@"
```

```
#!/bin/bash

source /shell_lib.sh

function __config__() {
  cat << EOF
    configVersion: v1
    # BINDING CONFIGURATION
EOF
}


function __main__() {
  #THE LOGIC
}

hook::run "$@"
```

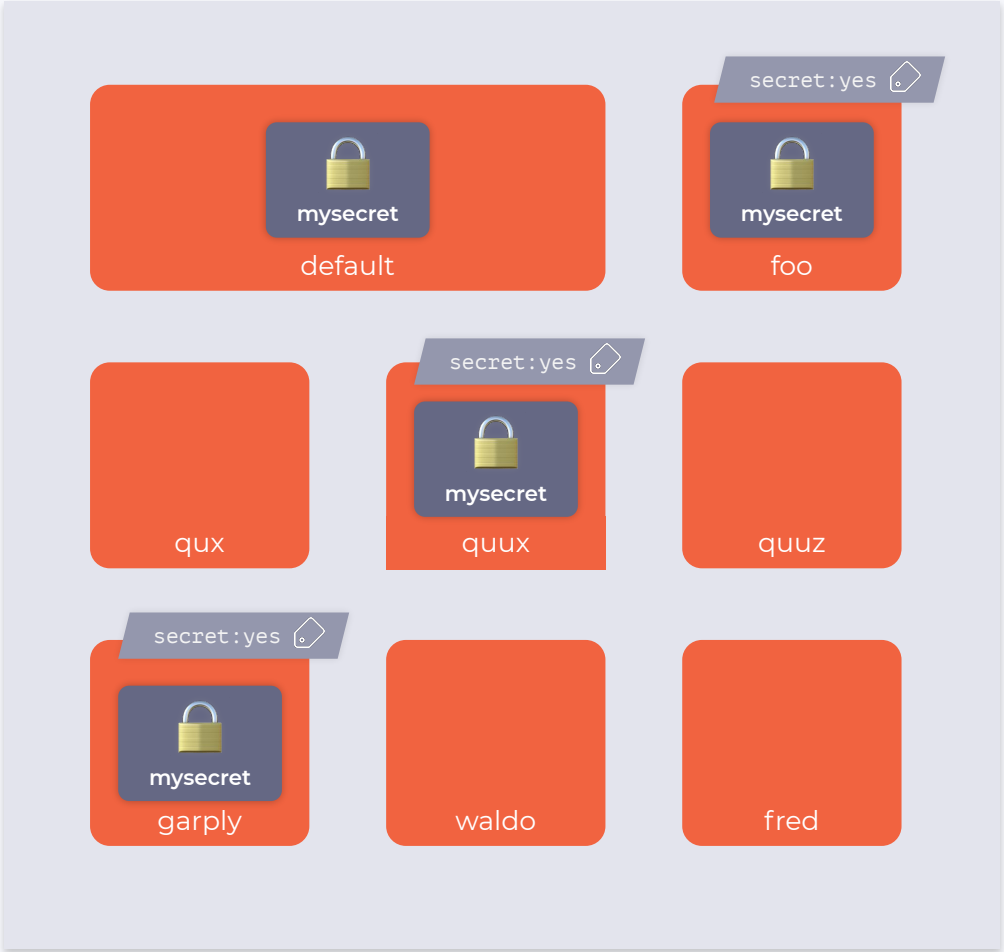

```
#!/bin/bash

source /shell_lib.sh

function __config__() {
  cat << EOF
    configVersion: v1
    # BINDING CONFIGURATION
EOF
}


function __main__() {
  #THE LOGIC
}

hook::run "$@"
```


```
#!/bin/bash

source /shell_lib.sh

function __config__() {
  cat << EOF
    configVersion: v1
    # BINDING CONFIGURATION
EOF
}


function __main__() {
  #THE LOGIC
}

hook::run "$@"
```



kubernetes
mysecret
default
secret:yes
mysecret
foo
qux
secret:yes
mysecret
quux
quuz
secret:yes
mysecret
garply
waldo
fred


kubernetes
mysecret
default
secret:yes
mysecret
foo
qux
secret:yes
mysecret
quux
quuz
secret:yes
mysecret
garply
waldo
fred


kubernetes
mysecret
default
secret:yes
mysecret
foo
qux
secret:yes
mysecret
quux
quuz
secret:yes
mysecret
garply
waldo
fred


kubernetes
mysecret
default
secret:yes
mysecret
foo
qux
secret:yes
mysecret
quux
quuz
secret:yes
mysecret
garply
waldo
fred


kubernetes
mysecret
default
secret:yes
mysecret
foo
qux
secret:yes
mysecret
quux
quuz
secret:yes
mysecret
garply
waldo
fred

Binding configuration

```
#!/bin/bash

source /shell_lib.sh

function __config__() {
  cat << EOF
    configVersion: v1
    # BINDING CONFIGURATION
EOF
}


function __main__() {

}

hook::run "$@"
```

Binding configuration

```
function __config__() {
  cat << EOF
    configVersion: v1
    # BINDING CONFIGURATION




EOF
}
```

Binding configuration

```
function __config__() {
  cat << EOF
    configVersion: v1
    kubernetes:
    - name: src_secret
      apiVersion: v1
      kind: Secret
      nameSelector:
        matchNames:
        - mysecret
      namespace:
        nameSelector:
          matchNames: ["default"]
      group: main
EOF
}
```

Binding configuration

```
function __config__() {
  cat << EOF
    configVersion: v1
    kubernetes:
    - name: src_secret
      apiVersion: v1
      kind: Secret
      nameSelector:
        matchNames:
        - mysecret
      namespace:
        nameSelector:
          matchNames: ["default"]
      group: main
EOF
}
```

Binding configuration

```
function __config__() {
  cat << EOF
    configVersion: v1
    kubernetes:
    - name: src_secret
      apiVersion: v1
      kind: Secret
      nameSelector:
        matchNames:
        - mysecret
      namespace:
        nameSelector:
          matchNames: ["default"]
      group: main
EOF
}
```

Binding configuration

```
function __config__() {
  cat << EOF
    configVersion: v1
    kubernetes:
    - name: src_secret
      apiVersion: v1
      kind: Secret
      nameSelector:
        matchNames:
        - mysecret
      namespace:
        nameSelector:
          matchNames: ["default"]
      group: main
EOF
}
```


Binding configuration

```
function __config__() {
  cat << EOF
    configVersion: v1
    kubernetes:
    - name: src_secret
      apiVersion: v1
      kind: Secret
      nameSelector:
        matchNames:
        - mysecret
      namespace:
        nameSelector:
          matchNames: ["default"]
      group: main
EOF
}
```

Binding configuration

```
function __config__() {
  cat << EOF
    configVersion: v1
    kubernetes:
    - name: src_secret
      apiVersion: v1
      kind: Secret
      nameSelector:
        matchNames:
        - mysecret
      namespace:
        nameSelector:
          matchNames: ["default"]
      group: main
EOF
}
```

Binding context

```
snapshots:
  src_secret:
  - object:
      apiVersion: v1
      kind: Secret
      metadata: { … }
      data: { … }
```

Binding configuration

```
function __config__() {
  cat << EOF
    configVersion: v1
    kubernetes:
    - name: src_secret
      apiVersion: v1
      kind: Secret
      nameSelector:
        matchNames:
        - mysecret
      namespace:
        nameSelector:
          matchNames: ["default"]
      group: main
EOF
}
```

Binding context

```
snapshots:
  src_secret:
  - object:
      apiVersion: v1
      kind: Secret
      metadata: { … }
      data: { … }
```

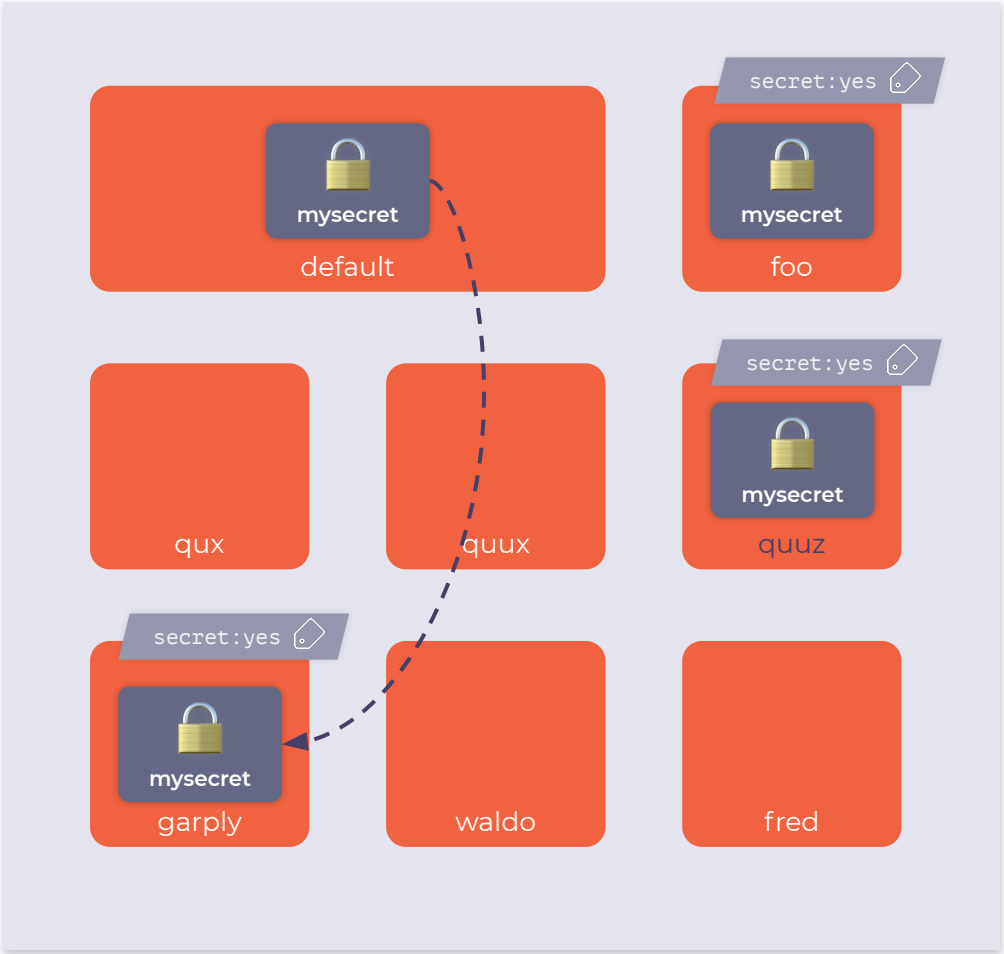

Binding configuration

```
function __config__() {
  cat << EOF
    configVersion: v1
    kubernetes:
    - name: src_secret
      apiVersion: v1
      kind: Secret
      nameSelector:
        matchNames:
        - mysecret
      namespace:
        nameSelector:
          matchNames: ["default"]
      group: main
EOF
}
```

Binding context

```
snapshots:
  src_secret:
  - object:
      apiVersion: v1
      kind: Secret
      metadata: { … }
      data: { … }
```


kubernetes


secret:yes
mysecret
default
mysecret
foo
qux
quux
secret:yes
mysecret
quuz
secret:yes
mysecret
garply
waldo
fred

Binding configuration

```
- name: namespaces
  group: main
  apiVersion: v1
  kind: Namespace
  jqFilter: |
    {
      namespace: .metadata.name,
      hasLabel: (
       .metadata.labels // {} |
         contains({"secret": "yes"})
      )
    }
  group: main
  keepFullObjectsInMemory: false
```


Binding configuration

```
- name: namespaces
  group: main
  apiVersion: v1
  kind: Namespace
  jqFilter: |
    {
      namespace: .metadata.name,
      hasLabel: (
       .metadata.labels // {} |
         contains({"secret": "yes"})
      )
    }
  group: main
  keepFullObjectsInMemory: false
```


Binding configuration

```
- name: namespaces
  group: main
  apiVersion: v1
  kind: Namespace
  jqFilter: |
    {
      namespace: .metadata.name,
      hasLabel: (
       .metadata.labels // {} |
         contains({"secret": "yes"})
      )
    }
  group: main
  keepFullObjectsInMemory: false
```


```
# kubectl get ns foo -o json | jq '{
>   "name": .metadata.name,
>   "hasLabel":   (
        .metadata.labels // {} |
          contains({"secret": "yes"})
      )
> }'
{
  "name": "kube-system",
  "hasLabel": true
}
```

Binding configuration

```
- name: namespaces
  group: main
  apiVersion: v1
  kind: Namespace
  jqFilter: |
    {
      namespace: .metadata.name,
      hasLabel: (
       .metadata.labels // {} |
         contains({"secret": "yes"})
      )
    }
  group: main
  keepFullObjectsInMemory: false
```


Binding configuration

```
- name: namespaces
  group: main
  apiVersion: v1
  kind: Namespace
  jqFilter: |
    {
      namespace: .metadata.name,
      hasLabel: (
       .metadata.labels // {} |
         contains({"secret": "yes"})
      )
    }
  group: main
  keepFullObjectsInMemory: false
```

Binding context

```
snapshots:
  namespaces:
  - filterResult:
      namespace: foo
      hasLabel: true
  - filterResult:
      namespace: bar
      hasLabel: false
  - filterResult:
      namespace: baz
      hasLabel: false
  - filterResult:
      namespace: quz
      hasLabel: false
  - ...
```


Binding configuration

```
- name: namespaces
  group: main
  apiVersion: v1
  kind: Namespace
  jqFilter: |
    {
      namespace: .metadata.name,
      hasLabel: (
       .metadata.labels // {} |
         contains({"secret": "yes"})
      )
    }
  group: main
  keepFullObjectsInMemory: false
```

Binding context

```
snapshots:
  namespaces:
  - filterResult:
      namespace: foo
      hasLabel: true
  - filterResult:
      namespace: bar
      hasLabel: false
  - filterResult:
      namespace: baz
      hasLabel: false
  - filterResult:
      namespace: quz
      hasLabel: false
  - ...
```


Binding configuration

```
- name: namespaces
  group: main
  apiVersion: v1
  kind: Namespace
  jqFilter: |
    {
      namespace: .metadata.name,
      hasLabel: (
        .metadata.labels // {} |
          contains({"secret": "yes"})
      )
    }
  group: main
  keepFullObjectsInMemory: false
```

Binding context

```
snapshots:
  namespaces:
  - filterResult:
      namespace: foo
      hasLabel: true
  - filterResult:
      namespace: bar
      hasLabel: false
  - filterResult:
      namespace: baz
      hasLabel: false
  - filterResult:
      namespace: quz
      hasLabel: false
  - ...
```


Binding configuration

```
- name: namespaces
  group: main
  apiVersion: v1
  kind: Namespace
  jqFilter: |
    {
      namespace: .metadata.name,
      hasLabel: (
       .metadata.labels // {} |
         contains({"secret": "yes"})
      )
    }
  group: main
  keepFullObjectsInMemory: false
```

Binding context

```
snapshots:
  namespaces:
  - filterResult:
      namespace: foo
      hasLabel: true



  - filterResult:
      namespace: bar
      hasLabel: false
  - filterResult:
      namespace: baz
      hasLabel: false
  - ...
```


Binding configuration

```
- name: namespaces
  group: main
  apiVersion: v1
  kind: Namespace
  jqFilter: |
    {
      namespace: .metadata.name,
      hasLabel: (
       .metadata.labels // {} |
         contains({"secret": "yes"})
      )
    }
  group: main
  keepFullObjectsInMemory: false
```

Binding context

```
snapshots:
  namespaces:
  - filterResult:
      namespace: foo
      hasLabel: true
    object:
      apiVersion: v1
      kind: Namespace
      metadata: { … }
  - filterResult:
      namespace: bar
      hasLabel: false
  - filterResult:
      namespace: baz
      hasLabel: false
  - ...
```


Binding configuration

```
- name: namespaces
  group: main
  apiVersion: v1
  kind: Namespace
  jqFilter: |
    {
      namespace: .metadata.name,
      hasLabel: (
       .metadata.labels // {} |
         contains({"secret": "yes"})
      )
    }
  group: main
  keepFullObjectsInMemory: false
```

Binding context

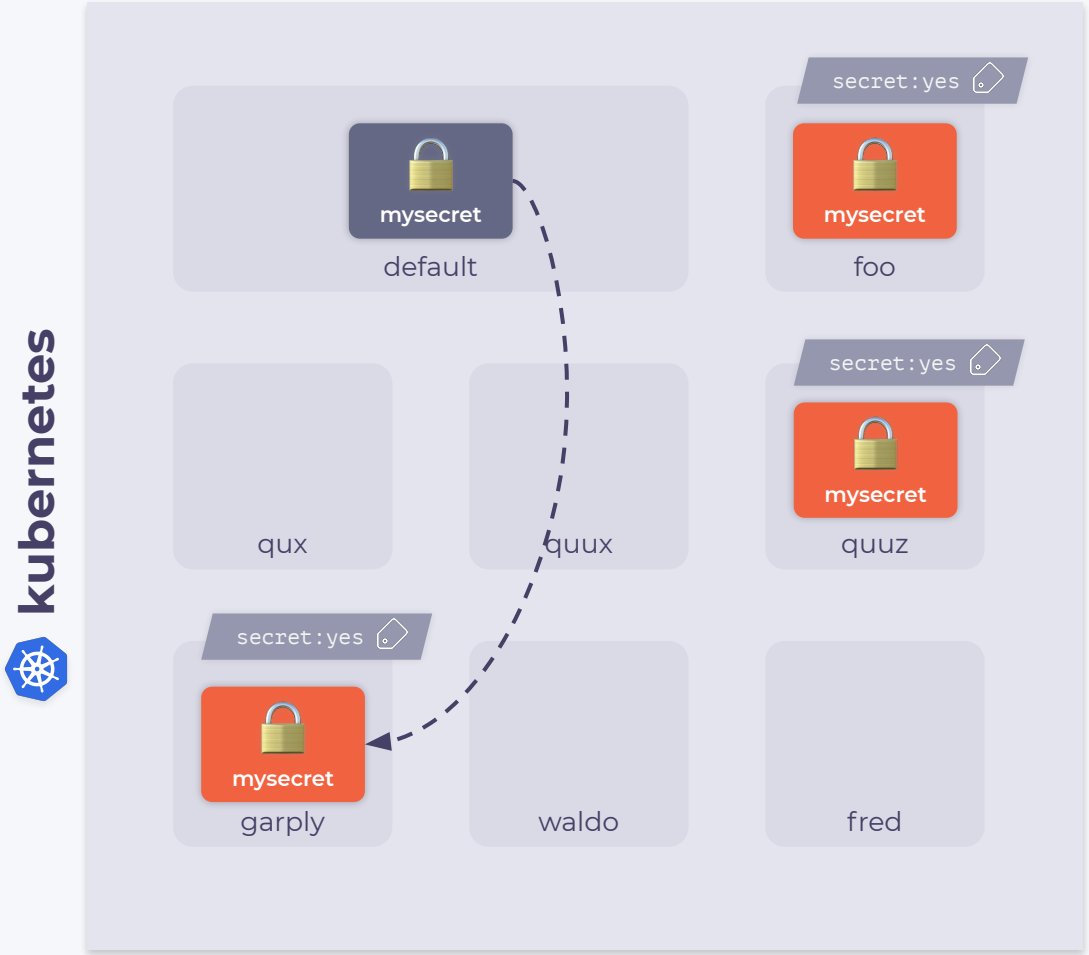
```
snapshots:
  namespaces:
  - filterResult:
      namespace: foo
      hasLabel: true
    object:
      apiVersion: v1
      kind: Namespace
      metadata: { … }
  - filterResult:
      namespace: bar
      hasLabel: false
  - filterResult:
      namespace: baz
      hasLabel: false
  - ...
```


Binding configuration

```
- name: namespaces
  group: main
  apiVersion: v1
  kind: Namespace
  jqFilter: |
    {
      namespace: .metadata.name,
      hasLabel: (
       .metadata.labels // {} |
         contains({"secret": "yes"})
      )
    }
  group: main
  keepFullObjectsInMemory: false
```

Binding context

```
snapshots:
  namespaces:
  - filterResult:
      namespace: foo
      hasLabel: true
      object:
        apiVersion: v1
        kind: Namespace
        metadata: { … }
  - filterResult:
      namespace: bar
      hasLabel: false
  - filterResult:
      namespace: baz
      hasLabel: false
  - ...
```


kubernetes
mysecret
default
secret:yes
mysecret
foo
qux
quux
secret:yes
mysecret
quuz
secret:yes
mysecret
garply
waldo
fred

Binding configuration

```
- name: dst_secrets
  apiVersion: v1
  kind: Secret
  labelSelector:
    matchLabels:
      managed-secret: "yes"
  jqFilter: |
    {
      "namespace":
        .metadata.namespace,
      "resourceVersion":
        .metadata.annotations.resourceVersion
    }
  group: main
  keepFullObjectsInMemory: false
```


Binding configuration

```
- name: dst_secrets
  apiVersion: v1
  kind: Secret
  labelSelector:
    matchLabels:
      managed-secret: "yes"
  jqFilter: |
    {
      "namespace":
        .metadata.namespace,
      "resourceVersion":
        .metadata.annotations.resourceVersion
    }
  group: main
  keepFullObjectsInMemory: false
```

Binding configuration

```
- name: dst_secrets
  apiVersion: v1
  kind: Secret
  labelSelector:
    matchLabels:
      managed-secret: "yes"
  jqFilter: |
    {
      "namespace":
        .metadata.namespace,
      "resourceVersion":
        .metadata.annotations.resourceVersion
    }
  group: main
  keepFullObjectsInMemory: false
```

Binding configuration

```
- name: dst_secrets
  apiVersion: v1
  kind: Secret
  labelSelector:
    matchLabels:
      managed-secret: "yes"
  jqFilter: |
    {
      "namespace":
        .metadata.namespace,
      "resourceVersion":
        .metadata.annotations.resourceVersion
    }
  group: main
  keepFullObjectsInMemory: false
```

Binding context

```
snapshots:
  dst_secrets:
  - filterResult:
      namespace: foo
      resourceVersion: 123456
```

Binding configuration

```
- name: dst_secrets
  apiVersion: v1
  kind: Secret
  labelSelector:
    matchLabels:
      managed-secret: "yes"
  jqFilter: |
    {
      "namespace":
        .metadata.namespace,
      "resourceVersion":
        .metadata.annotations.resourceVersion
    }
  group: main
  keepFullObjectsInMemory: false
```

Binding context

```
snapshots:
  dst_secrets:
  - filterResult:
      namespace: foo
      resourceVersion: 123456
```

```
src_secret:
- object:
    apiVersion: v1
    kind: Secret
    metadata: { … }
    spec: { … }
```

```
dst_secrets:
- filterResult:
    namespace: foo
    resourceVersion: 123456
- filterResult:
    namespace: corge
    resourceVersion: 123456
- filterResult:
    namespace: xyzzy
    resourceVersion: 123456
```

```
namespaces:
- filterResult:
    namespace: foo
    hasLabel: true
- filterResult:
    namespace: bar
    hasLabel: false
- filterResult:
    namespace: baz
    hasLabel: false
- filterResult:
    namespace: quz
    hasLabel: false
- ...
```


```
src_secret:
- object:
    apiVersion: v1
    kind: Secret
    metadata: { … }
    spec: { … }
```

```
dst_secrets:
- filterResult:
    namespace: foo
    resourceVersion: 123456
- filterResult:
    namespace: corge
    resourceVersion: 123456
- filterResult:
    namespace: xyzzy
    resourceVersion: 123456
```

```
namespaces:
- filterResult:
    namespace: foo
    hasLabel: true
- filterResult:
    namespace: bar
    hasLabel: false
- filterResult:
    namespace: baz
    hasLabel: false
- filterResult:
    namespace: quz
    hasLabel: false
- ...
```

```
src_secret:
- object:
    apiVersion: v1
    kind: Secret
    metadata: { … }
    spec: { … }
```

```
dst_secrets:
- filterResult:
    namespace: foo
    resourceVersion: 123456
- filterResult:
    namespace: corge
    resourceVersion: 123456
- filterResult:
    namespace: xyzzy
    resourceVersion: 123456
```

```
namespaces:
- filterResult:
    namespace: foo
    hasLabel: true
- filterResult:
    namespace: bar
    hasLabel: false
- filterResult:
    namespace: baz
    hasLabel: false
- filterResult:
    namespace: quz
    hasLabel: false
- ...
```


src_secret:
- object:
apiVersion: v1
kind: Secret
metadata: { … }
spec: { … }
dst_secrets:
- filterResult:
namespace: foo
resourceVersion: 123456
- filterResult:
namespace: corge
resourceVersion: 123456
- filterResult:
namespace: xyzzy
resourceVersion: 123456
namespaces:
- filterResult:
namespace: foo
hasLabel: true
- filterResult:
namespace: bar
hasLabel: false
- filterResult:
namespace: baz
hasLabel: false
- filterResult:
namespace: quz
hasLabel: false
- ...

```
src_secret:
- object:
    apiVersion: v1
    kind: Secret
    metadata: { … }
    spec: { … }
```

```
dst_secrets:
- filterResult:
    namespace: foo
    resourceVersion: 123456
- filterResult:
    namespace: corge
    resourceVersion: 123456
- filterResult:
    namespace: xyzzy
    resourceVersion: 123456
```

```
namespaces:
- filterResult:
    namespace: foo
    hasLabel: true
- filterResult:
    namespace: bar
    hasLabel: false
- filterResult:
    namespace: baz
    hasLabel: false
- filterResult:
    namespace: quz
    hasLabel: false
- ...
```

Is `dst` in sync with `src`?

yes do nothing **no** kubectl create or replace

```
src_secret:
- object:
    apiVersion: v1
    kind: Secret
    metadata: { … }
    spec: { … }
```

```
dst_secrets:
- filterResult:
    namespace: foo
    resourceVersion: 123456
- filterResult:
    namespace: corge
    resourceVersion: 123456
- filterResult:
    namespace: xyzzy
    resourceVersion: 123456
```

```
namespaces:
- filterResult:
    namespace: foo
    hasLabel: true
- filterResult:
    namespace: bar
    hasLabel: false
- filterResult:
    namespace: baz
    hasLabel: false
- filterResult:
    namespace: quz
    hasLabel: false
- ...
```

```
src_secret:
- object:
    apiVersion: v1
    kind: Secret
    metadata: { … }
    spec: { … }
```

```
dst_secrets:
- filterResult:
    namespace: foo
    resourceVersion: 123456
- filterResult:
    namespace: corge
    resourceVersion: 123456
- filterResult:
    namespace: xyzzy
    resourceVersion: 123456
```

```
namespaces:
- filterResult:
    namespace: foo
    hasLabel: true
- filterResult:
    namespace: bar
    hasLabel: false
- filterResult:
    namespace: baz
    hasLabel: false
- filterResult:
    namespace: quz
    hasLabel: false
- ...
```

```
src_secret:
- object:
    apiVersion: v1
    kind: Secret
    metadata: { … }
    spec: { … }
```

```
dst_secrets:
- filterResult:
    namespace: foo
    resourceVersion: 123456
- filterResult:
    namespace: corge
    resourceVersion: 123456
- filterResult:
    namespace: xyzzy
    resourceVersion: 123456
```

```
namespaces:
- filterResult:
    namespace: foo
    hasLabel: true
- filterResult:
    namespace: bar
    hasLabel: false
- filterResult:
    namespace: baz
    hasLabel: false
- filterResult:
    namespace: quz
    hasLabel: false
- ...
```

```
src_secret:
- object:
    apiVersion: v1
    kind: Secret
    metadata: { … }
    spec: { … }
```

```
dst_secrets:
- filterResult:
    namespace: foo
    resourceVersion: 123456
- filterResult:
    namespace: corge
    resourceVersion: 123456
- filterResult:
    namespace: xyzzy
    resourceVersion: 123456
```

```
namespaces:
- filterResult:
    namespace: foo
    hasLabel: true
- filterResult:
    namespace: bar
    hasLabel: false
- filterResult:
    namespace: baz
    hasLabel: false
- filterResult:
    namespace: quz
    hasLabel: false
- ...
```

Does `dst` exist?

yes kubectl delete **no** do nothing

```
function __main__() {




}
```

```
function __main__() {
  for i in $(seq 0 "$(context::jq -r '(.snapshots.namespaces | length) - 1')"); do
    ns_name="$(context::jq -r '.snapshots.namespaces['"$i"'].filterResult.name')"

    if context::jq -e '.snapshots.namespaces['"$i"'].filterResult.hasLabel'; then
      sync_secret "$ns_name"
    else
      delete_secret "$ns_name"
    fi
  done
}
```


```
function __main__() {
  for i in $(seq 0 "$(context::jq -r '(.snapshots.namespaces | length) - 1')"); do
    ns_name="$(context::jq -r '.snapshots.namespaces['"$i"'].filterResult.name')"

    if context::jq -e '.snapshots.namespaces['"$i"'].filterResult.hasLabel'; then
      sync_secret "$ns_name"
    else
      delete_secret "$ns_name"
    fi
  done
}
```


```
function __main__() {
  for i in $(seq 0 "$(context::jq -r '(.snapshots.namespaces | length) - 1')"); do
    ns_name="$(context::jq -r '.snapshots.namespaces['"$i"'].filterResult.name')"

    if context::jq -e '.snapshots.namespaces['"$i"'].filterResult.hasLabel'; then
      sync_secret "$ns_name"
    else
      delete_secret "$ns_name"
    fi
  done
}
```


```
function __main__() {
  for i in $(seq 0 "$(context::jq -r '(.snapshots.namespaces | length) - 1')"); do
    ns_name="$(context::jq -r '.snapshots.namespaces['"$i"'].filterResult.name')"

    if context::jq -e '.snapshots.namespaces['"$i"'].filterResult.hasLabel'; then
      sync_secret "$ns_name"
    else
      delete_secret "$ns_name"
    fi
  done
}
```

```
function delete_secret() {
  if context::jq -e --arg ns "$1" 'select(.snapshots…namespace == $ns)' ; then
    kubectl -n "$1" delete secret "mysecret"
  fi
}
```


```
function delete_secret() {
  if context::jq -e --arg ns "$1" 'select(.snapshots…namespace == $ns)' ; then
    kubectl -n "$1" delete secret "mysecret"
  fi
}
```


```
function delete_secret() {
  if context::jq -e --arg ns "$1" 'select(.snapshots…namespace == $ns)' ; then
    kubectl -n "$1" delete secret "mysecret"
  fi
}
```


```
function delete_secret() {
  if context::jq -e --arg ns "$1" 'select(.snapshots…namespace == $ns)' ; then
    kubectl -n "$1" delete secret "mysecret"
  fi
}

function sync_secret() {
  src_resource_version="$(…)"
  dst_secret_resource_version="$(…)"

  if [ "$src_resource_version" != "$dst_resource_version" ] ; then
    new_secret="$(context::jq -r '.snapshots.src_secret[0].object | …)"

    kubectl -n "$1" replace -f <(echo "$new_secret") ||
      kubectl -n "$1" create -f  <(echo "$new_secret")
  fi
}
```


```
function delete_secret() {
  if context::jq -e --arg ns "$1" 'select(.snapshots…namespace == $ns)' ; then
    kubectl -n "$1" delete secret "mysecret"
  fi
}

function sync_secret() {
  src_resource_version="$(…)"
  dst_secret_resource_version="$(…)"

  if [ "$src_resource_version" != "$dst_resource_version" ] ; then
    new_secret="$(context::jq -r '.snapshots.src_secret[0].object | …)"

    kubectl -n "$1" replace -f <(echo "$new_secret") ||
      kubectl -n "$1" create -f  <(echo "$new_secret")
  fi
}
```

```
function delete_secret() {
  if context::jq -e --arg ns "$1" 'select(.snapshots…namespace == $ns)' ; then
    kubectl -n "$1" delete secret "mysecret"
  fi
}

function sync_secret() {
  src_resource_version="$(…)"
  dst_secret_resource_version="$(…)"

  if [ "$src_resource_version" != "$dst_resource_version" ] ; then
    new_secret="$(context::jq -r '.snapshots.src_secret[0].object | …)"

    kubectl -n "$1" replace -f <(echo "$new_secret") ||
      kubectl -n "$1" create -f  <(echo "$new_secret")
  fi
}
```

```
function delete_secret() {
  if context::jq -e --arg ns "$1" 'select(.snapshots…namespace == $ns)' ; then
    kubectl -n "$1" delete secret "mysecret"
  fi
}

function sync_secret() {
  src_resource_version="$(…)"
  dst_secret_resource_version="$(…)"

  if [ "$src_resource_version" != "$dst_resource_version" ] ; then
    new_secret="$(context::jq -r '.snapshots.src_secret[0].object | …)"

    kubectl -n "$1" replace -f <(echo "$new_secret") ||
      kubectl -n "$1" create -f  <(echo "$new_secret")
  fi
}
```


```
function delete_secret() {
  if context::jq -e --arg ns "$1" 'select(.snapshots…namespace == $ns)' ; then
    kubectl -n "$1" delete secret "mysecret"
  fi
}

function sync_secret() {
  src_resource_version="$(…)"
  dst_secret_resource_version="$(…)"

  if [ "$src_resource_version" != "$dst_resource_version" ] ; then
    new_secret="$(context::jq -r '.snapshots.src_secret[0].object | …)"

    kubectl -n "$1" replace -f <(echo "$new_secret") ||
      kubectl -n "$1" create -f  <(echo "$new_secret")
  fi
}
```

```
function delete_secret() {
  if context::jq -e --arg ns "$1" 'select(.snapshots…namespace == $ns)' ; then
    kubectl -n "$1" delete secret "mysecret"
  fi
}

function sync_secret() {
  src_resource_version="$(…)"
  dst_secret_resource_version="$(…)"

  if [ "$src_resource_version" != "$dst_resource_version" ] ; then
    new_secret="$(context::jq -r '.snapshots.src_secret[0].object | …)"

    kubectl -n "$1" replace -f <(echo "$new_secret") ||
      kubectl -n "$1" create -f  <(echo "$new_secret")
  fi
}
```

```
function delete_secret() {
  if context::jq -e --arg ns "$1" 'select(.snapshots…namespace == $ns)' ; then
    kubectl -n "$1" delete secret "mysecret"
  fi
}

function sync_secret() {
  src_resource_version="$(…)"
  dst_secret_resource_version="$(…)"

  if [ "$src_resource_version" != "$dst_resource_version" ] ; then
    new_secret="$(context::jq -r '.snapshots.src_secret[0].object | …)"

    kubectl -n "$1" replace -f <(echo "$new_secret") ||
      kubectl -n "$1" create -f  <(echo "$new_secret")
  fi
}
```


```
function delete_secret() {
  if context::jq -e --arg ns "$1" 'select(.snapshots…namespace == $ns)' ; then
    kubectl -n "$1" delete secret "mysecret"
  fi
}

function sync_secret() {
  src_resource_version="$(…)"
  dst_secret_resource_version="$(…)"

  if [ "$src_resource_version" != "$dst_resource_version" ] ; then
    new_secret="$(context::jq -r '.snapshots.src_secret[0].object | …)"

    kubectl -n "$1" replace -f <(echo "$new_secret") ||
      kubectl -n "$1" create -f  <(echo "$new_secret")
  fi
}
```

```
function delete_secret() {
  if context::jq -e --arg ns "$1" 'select(.snapshots…namespace == $ns)' ; then
    kubectl -n "$1" delete secret "mysecret"
  fi
}

function sync_secret() {
  src_resource_version="$(…)"
  dst_secret_resource_version="$(…)"

  if [ "$src_resource_version" != "$dst_resource_version" ] ; then
    new_secret="$(context::jq -r '.snapshots.src_secret[0].object | …)"

    kubectl -n "$1" replace -f <(echo "$new_secret") ||
      kubectl -n "$1" create -f  <(echo "$new_secret")
  fi
}
```

github.com/flant/examples/tree/master/2020/08-kubecon

35 lines in yaml

```
configVersion: v1
kubernetes:
- name: src_secret
  apiVersion: v1
  kind: Secret
  nameSelector:
    matchNames:
      mysecret
  namespace:
    nameSelector:
      matchNames: ["default"]
  group: main
- name: dst_secrets
  apiVersion: v1
  kind: Secret
  labelSelector:
    matchLabels:
      managed-secret: "yes"
  jqFilter: |
    {
      "namespace": .metadata.namespace,
      "resourceVersion": .annotations."resourceVersion"
    }
  group: main
  keepFullObjectsInMemory: false
- name: namespaces
  group: main
  apiVersion: v1
  kind: Namespace
  jqFilter: |
    {
      name: .metadata.name,
      hasLabel: (.metadata.labels // {} | contains({"secret": "yes"}))
    }
  group: main
  keepFullObjectsInMemory: false
```

+

20 lines in bash

```
function __main__() {
  for i in $(seq 0 "$(context::jq -r '(.snapshots.namespaces | length) - 1')");
do
    ns_name="$(context::jq -r '.snapshots.namespaces["$i"].filterResult.name')"

    if context::jq -e '.snapshots.namespaces["$i"].filterResult.hasLabel'; then
      sync_secret "$ns_name"
    else
      delete_secret "$ns_name"
    fi
  done
}


function delete_secret() {
  if context::jq -e --arg ns "$1" 'select(.snapshots_namespace == $ns)' ; then
    kubectl -n "$1" delete secret "mysecret"
  fi
}

function sync_secret() {
  src_resource_version="$(…)"
  dst_secret_resource_version="$(…)"

  if [ "$src_resource_version" != "$dst_resource_version" ] ; then
    new_secret="$(context::jq -r '.snapshots.src_secret[0].object | …)"

    kubectl -n "$1" replace -f <(echo "$new_secret") ||
      kubectl -n "$1" create -f  <(echo "$new_secret")
  fi
}
```

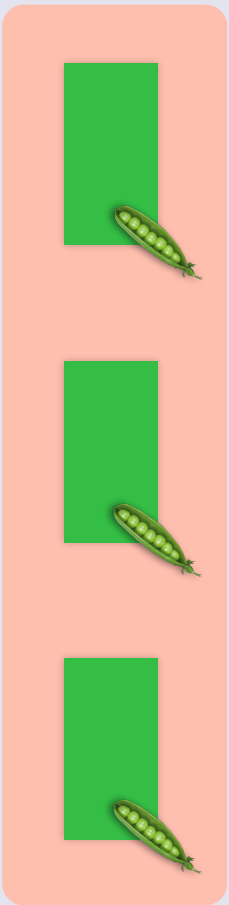
+

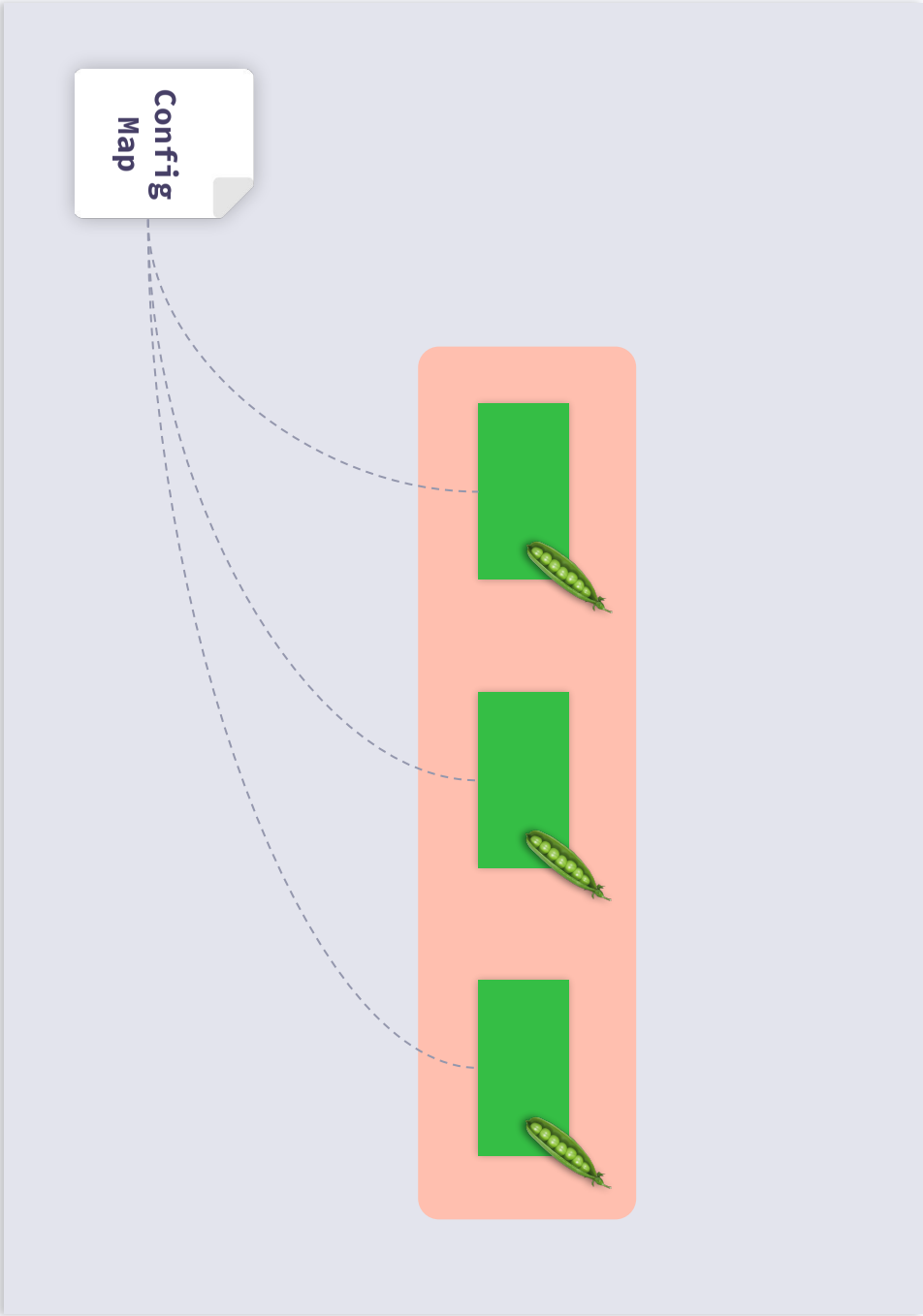
shell-operator

Go? Bash! Meet the Shell-operator

FLANT

kubernetes



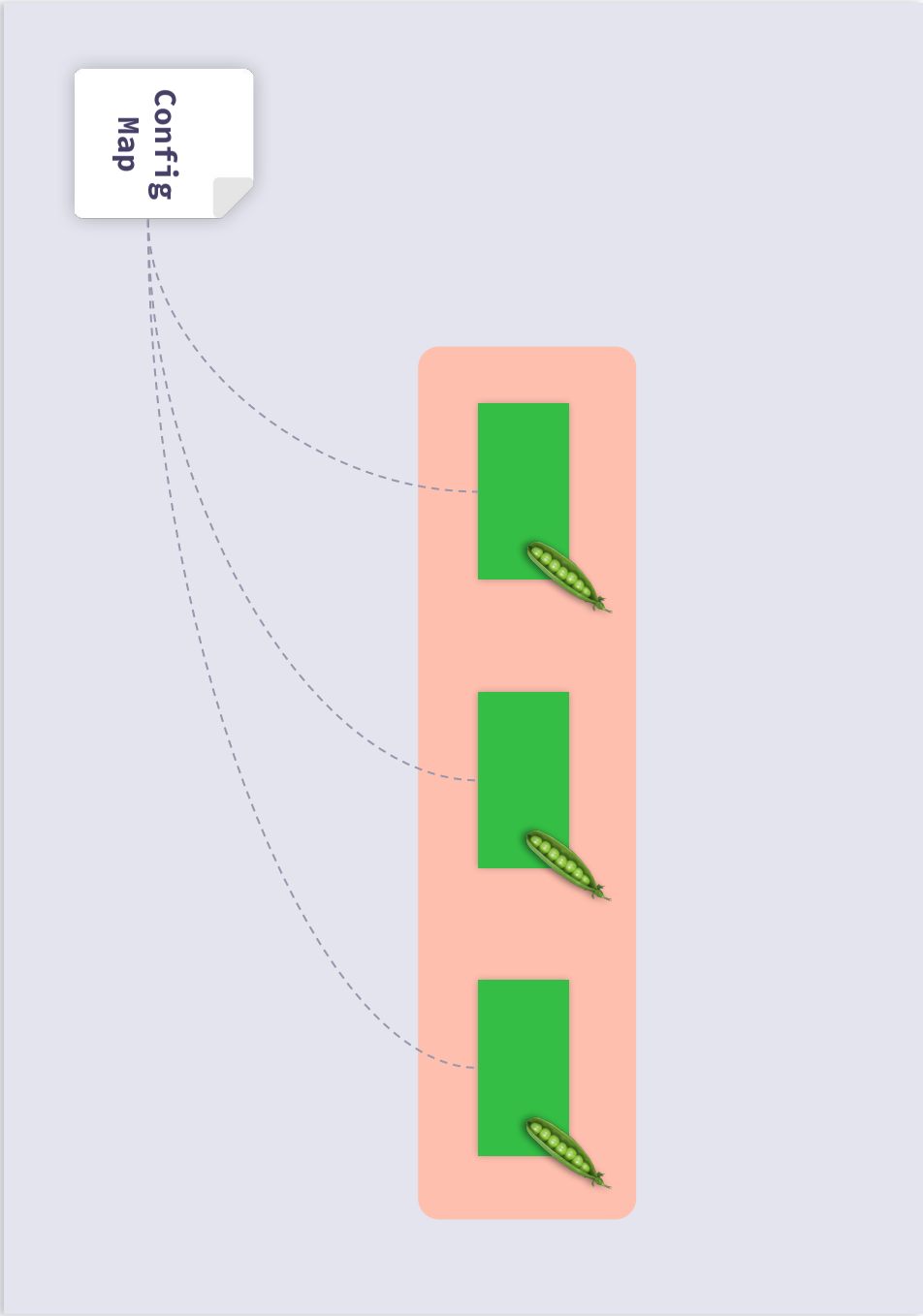

Config
Map

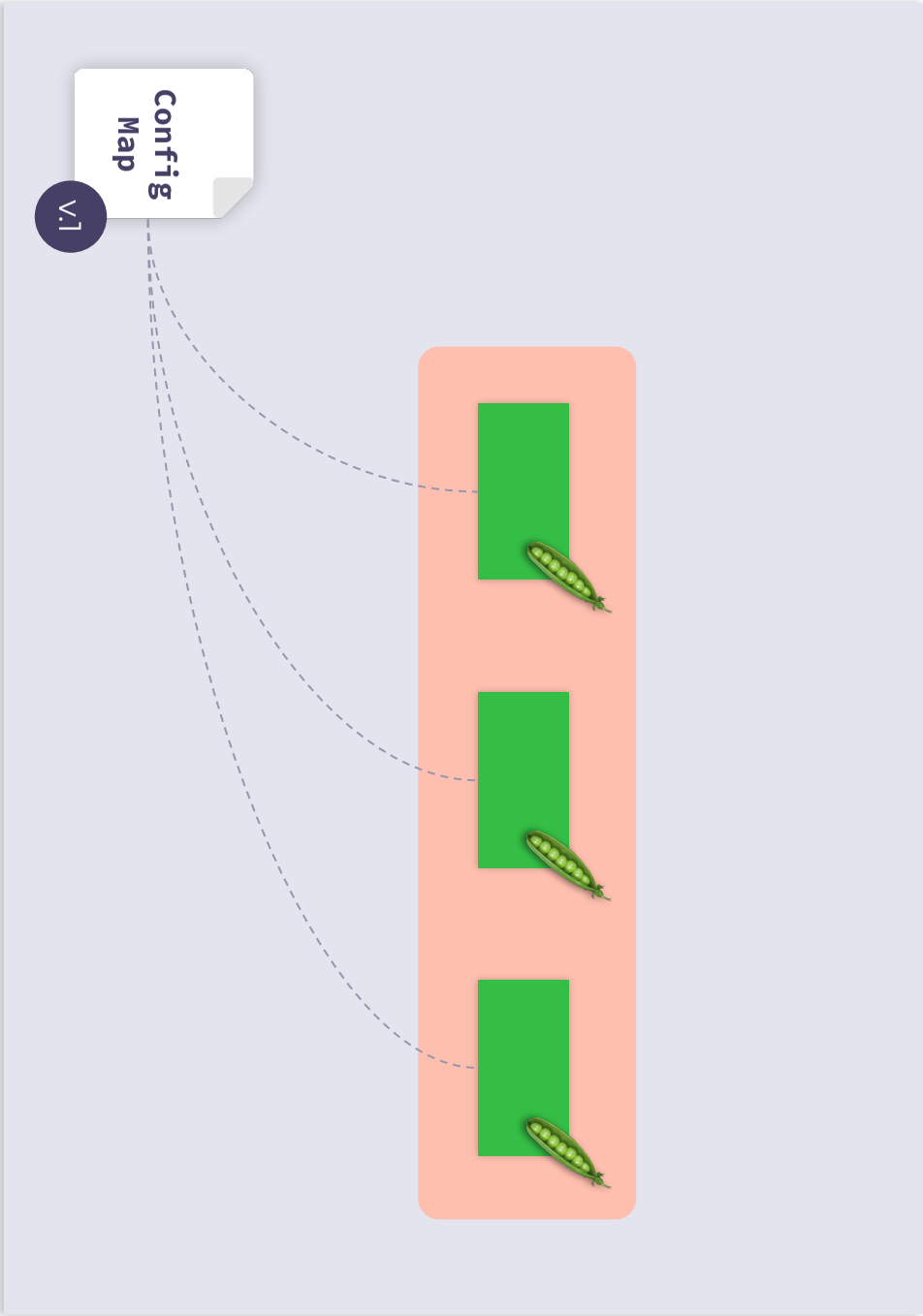
ENV

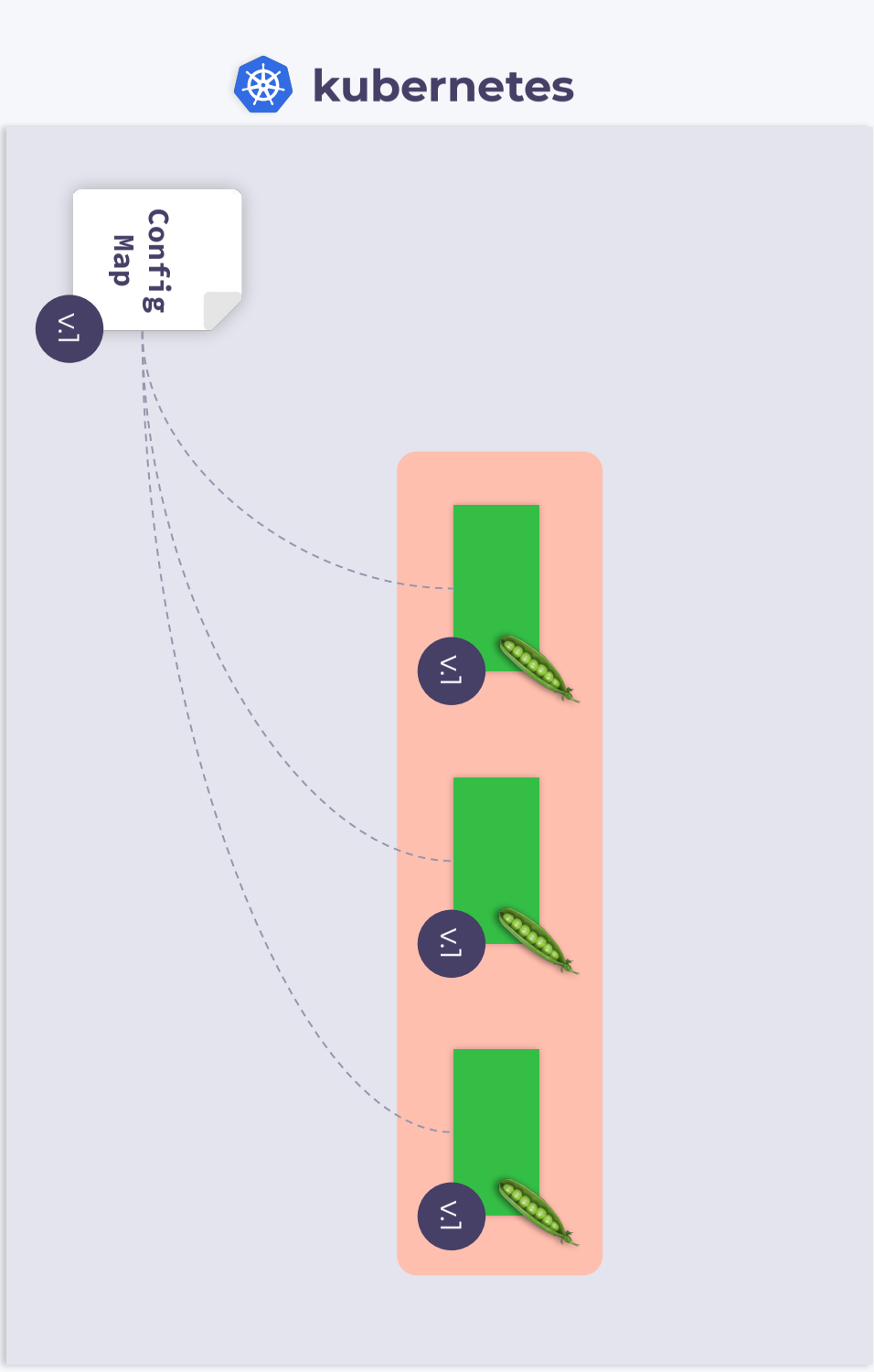
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: foo
spec:
  template:
    spec:
      containers:
        - name: bar
          image: registry.example.com/myapp:v1.0.0
          env:
          - name: DB_USER
            valueFrom:
              configMapKeyRef:
                name: foo
                key: db_user
          - name: DB_PASSWORD
            valueFrom:
              configMapKeyRef:
                name: foo
                key: db_password
```

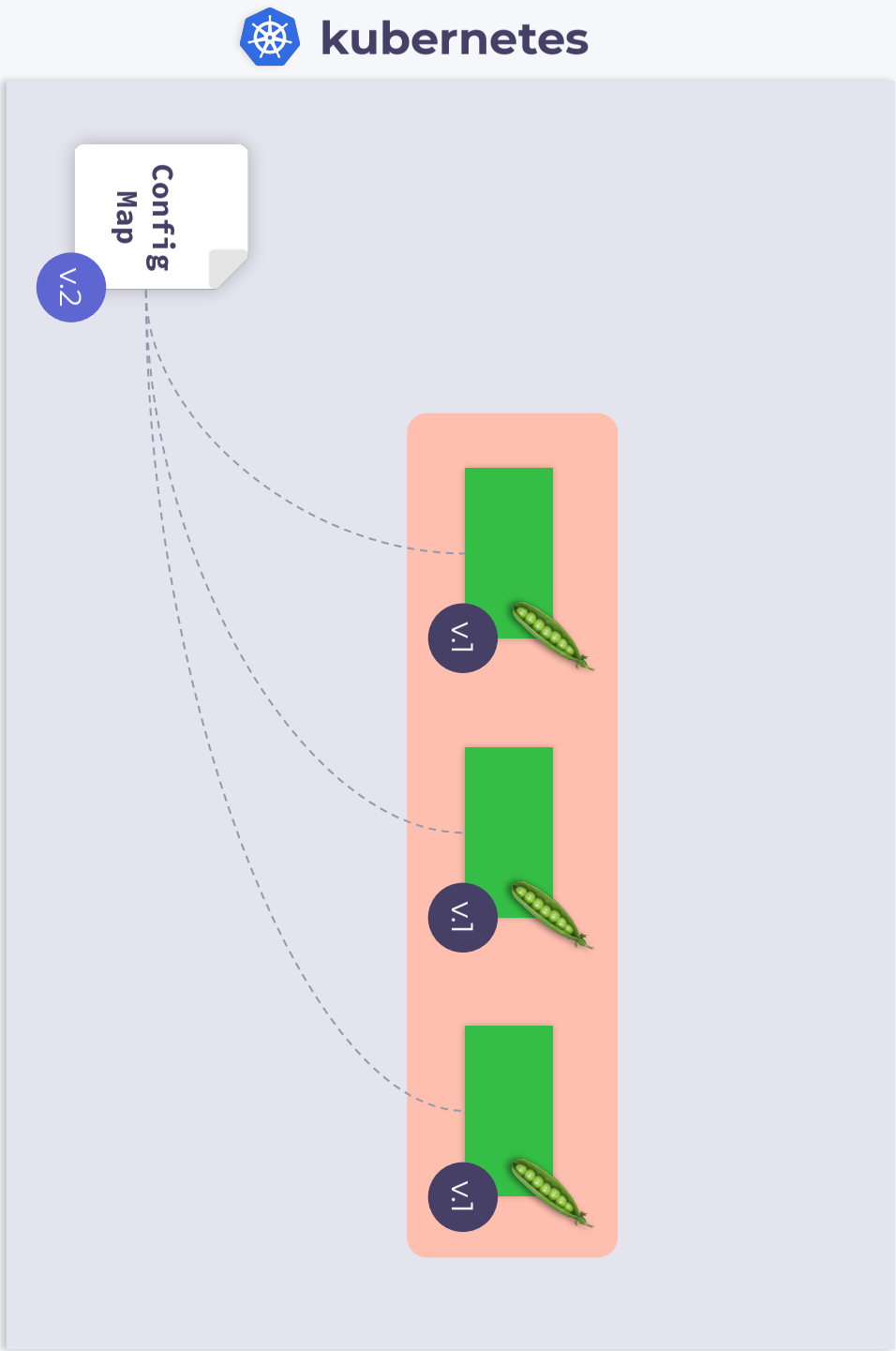
Mount

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: foo
spec:
  template:
    spec:
      containers:
        - name: bar
          image: registry.example.com/myapp:v1.0.0
          volumeMounts:
          - mountPath: /etc/config
            name: config
      volumes:
      - name: config
        configMap:
          name: foo
          items:
          - key: config.yaml
             path: config.yaml
```


Config
Map


Config Map
v.1

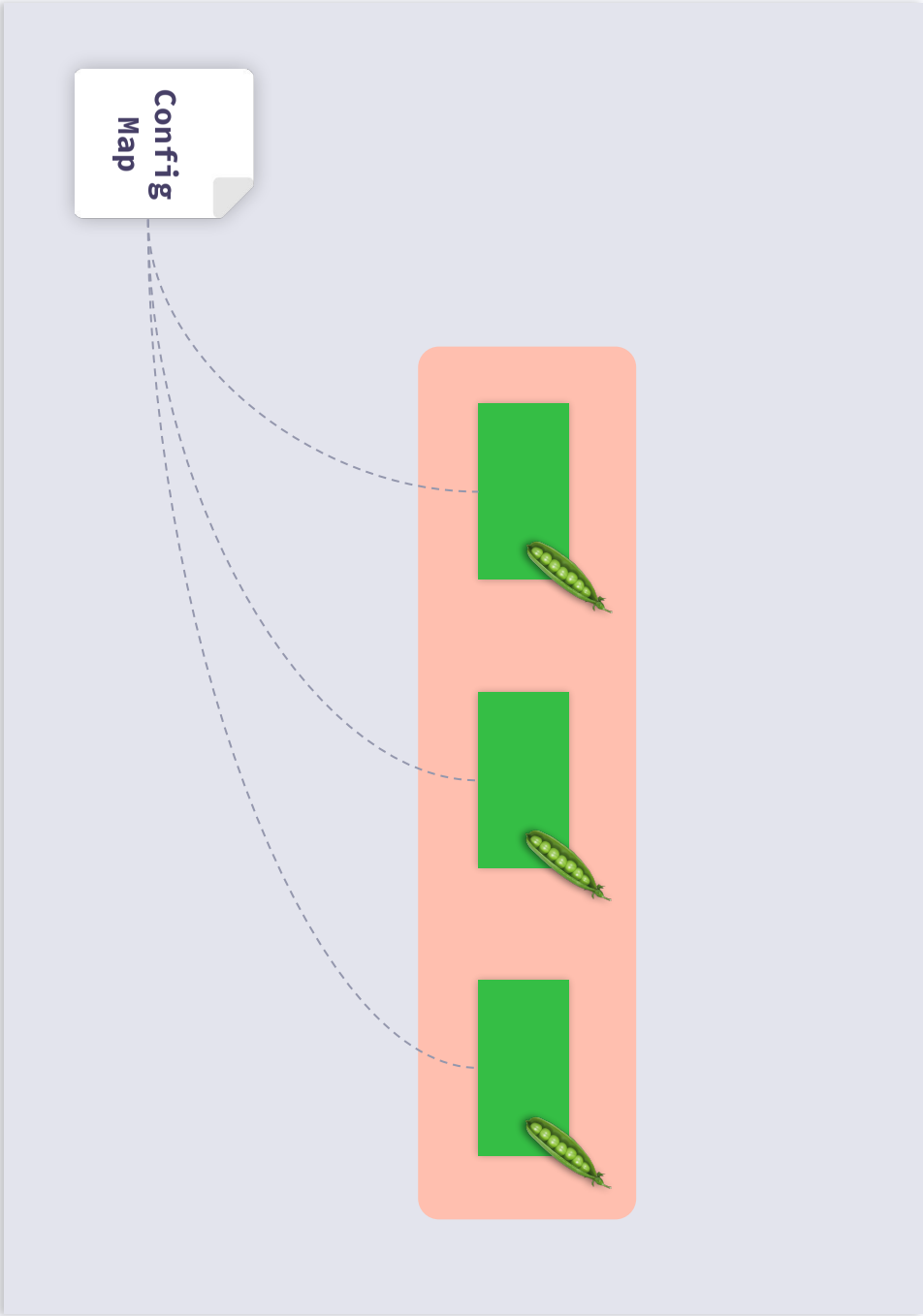

kubernetes
Config Map
v.1
v.1
v.1
v.1

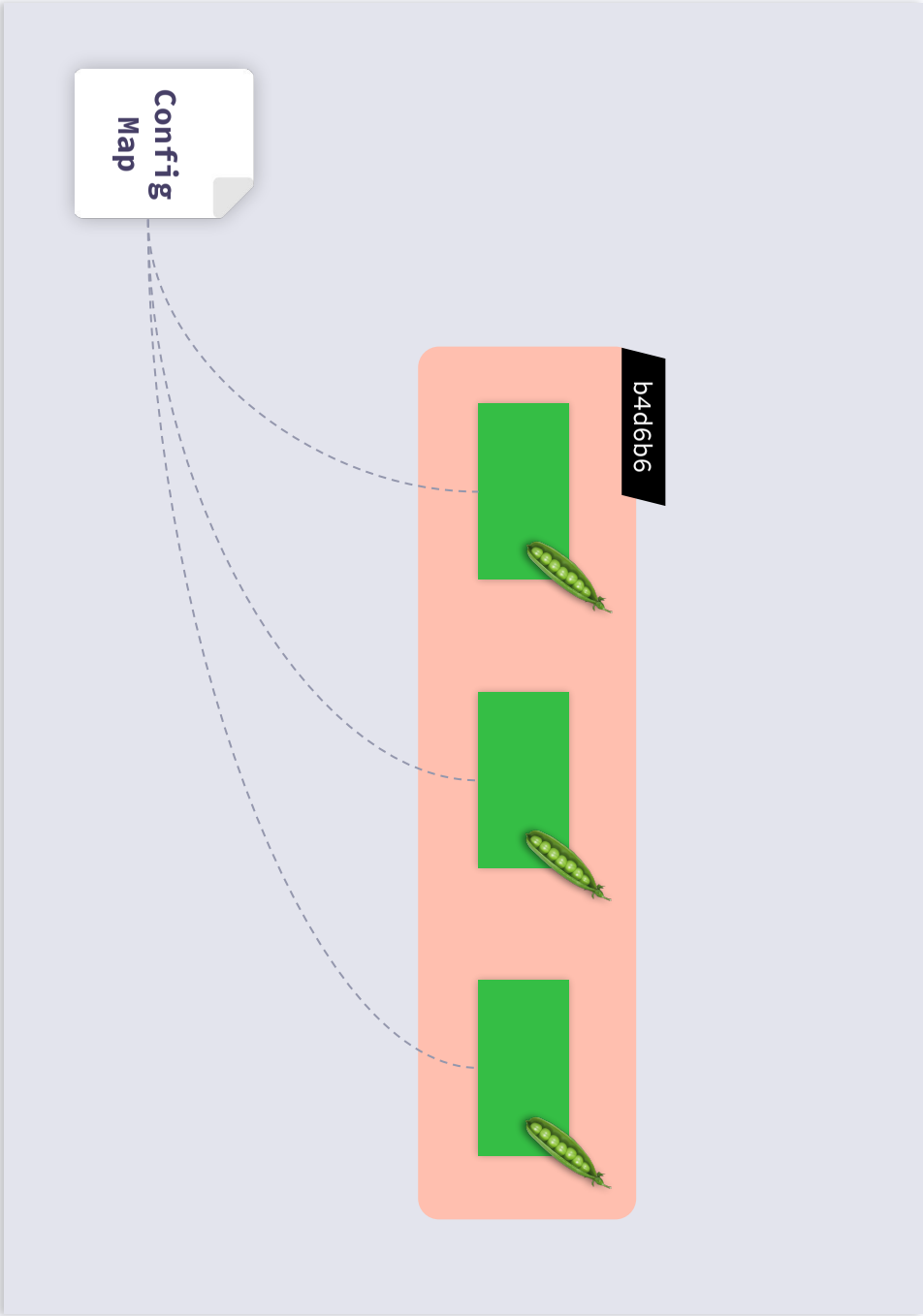

kubernetes
Config Map
v.2
v.1
v.1
v.1

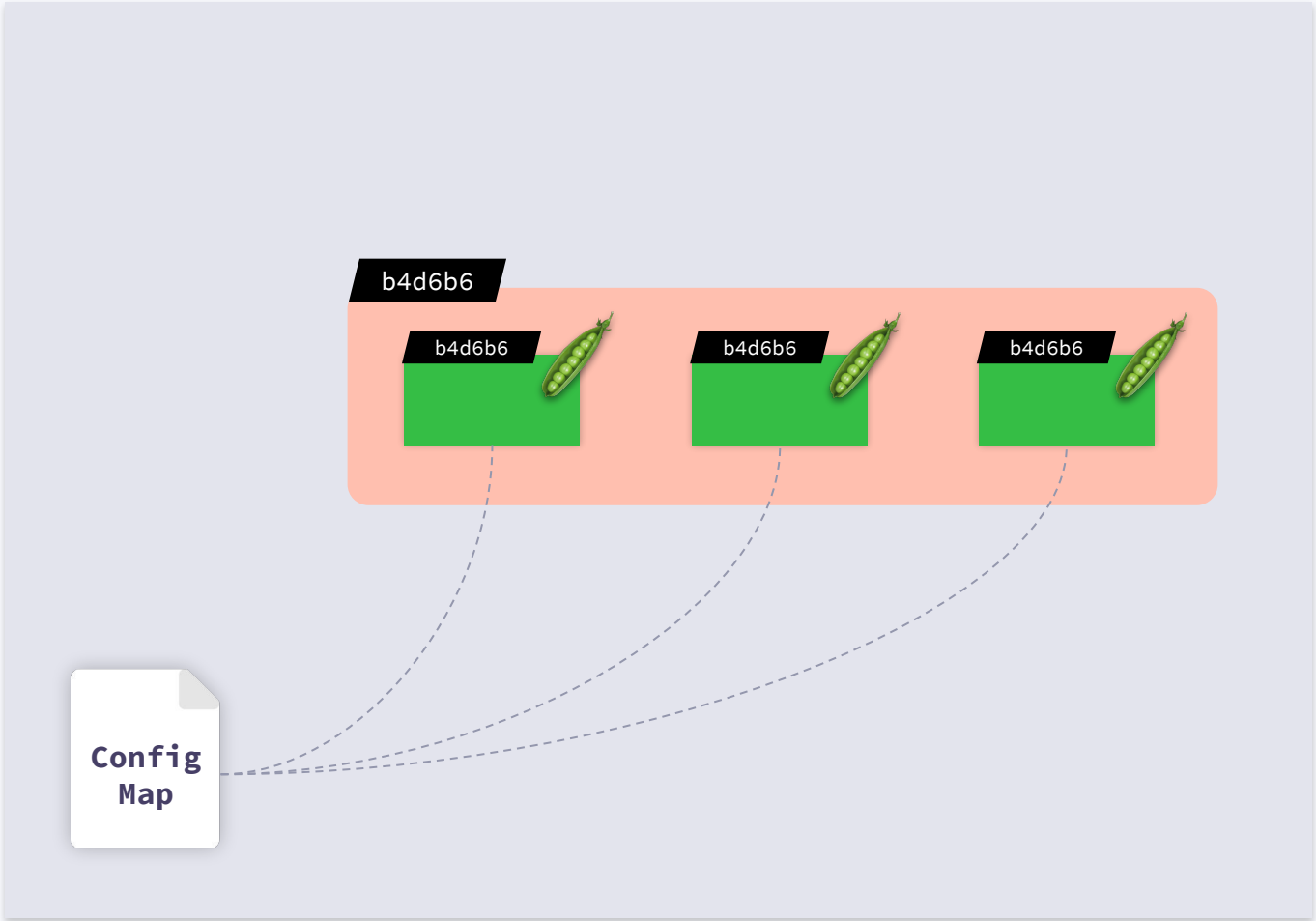
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: foo
spec:
  template:
    spec:
      containers:
        - name: bar
          image: registry.example.com/myapp:v1.0.0
          env:
          - name: DB_USER
            valueFrom:
              configMapKeyRef:
                name: foo
                key: db_user
          - name: DB_PASSWORD
            valueFrom:
              configMapKeyRef:
                name: foo
                key: db_password
```

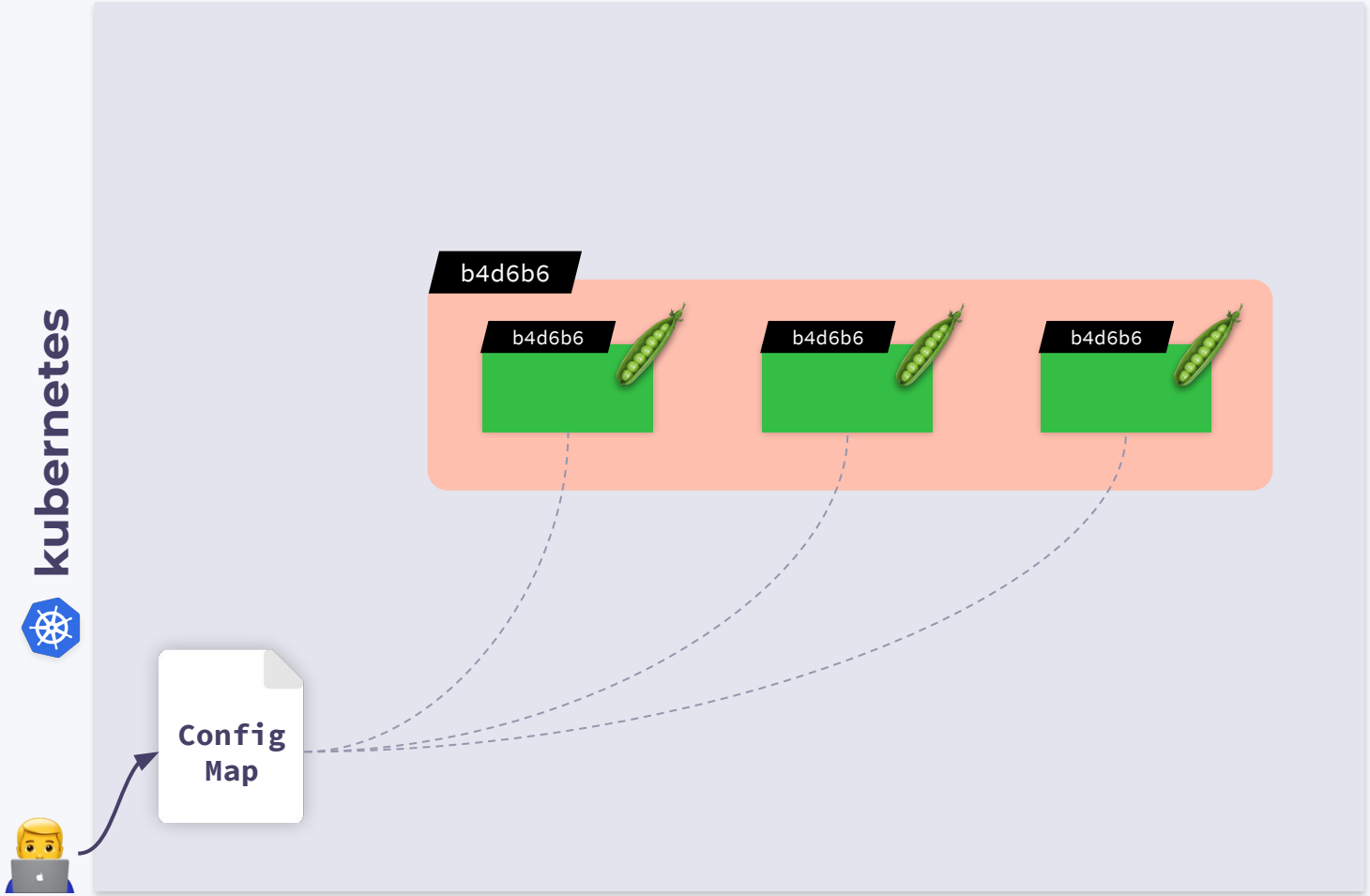
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: foo
spec:
  template:



    spec:
      containers:
        - name: bar
          image: registry.example.com/myapp:v1.0.0
          env:
          - name: DB_USER
            valueFrom:
              configMapKeyRef:
                name: foo
                key: db_user
          - name: DB_PASSWORD
            valueFrom:
              configMapKeyRef:
                name: foo
                key: db_password
```

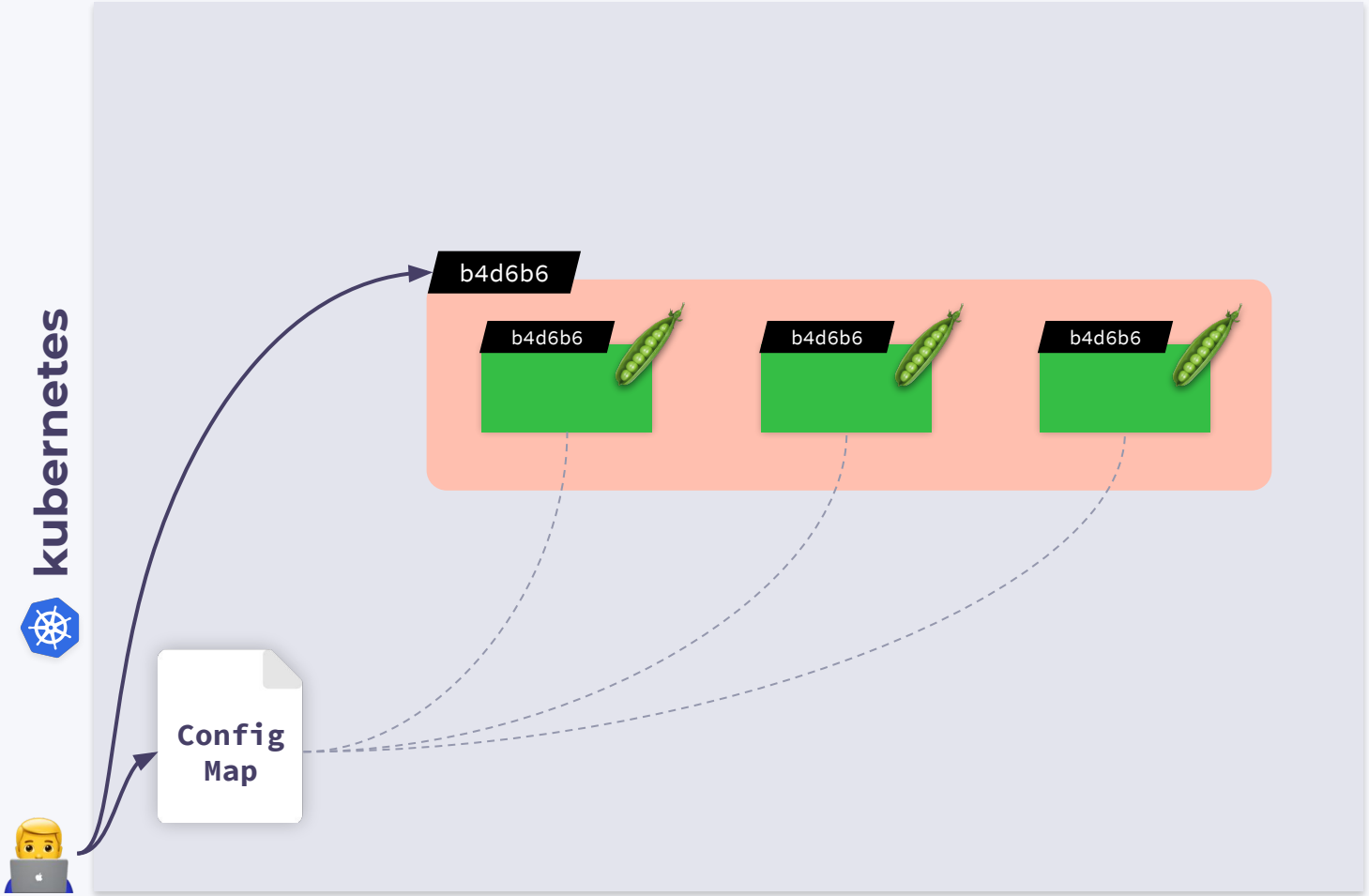
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: foo
spec:
  template:
    metadata:
      annotations:
        config/checksum: b4d6b660fffb798dcaca5b8a
    spec:
      containers:
        - name: bar
          image: registry.example.com/myapp:v1.0.0
          env:
          - name: DB_USER
            valueFrom:
              configMapKeyRef:
                name: foo
                key: db_user
          - name: DB_PASSWORD
            valueFrom:
              configMapKeyRef:
                name: foo
                key: db_password
```

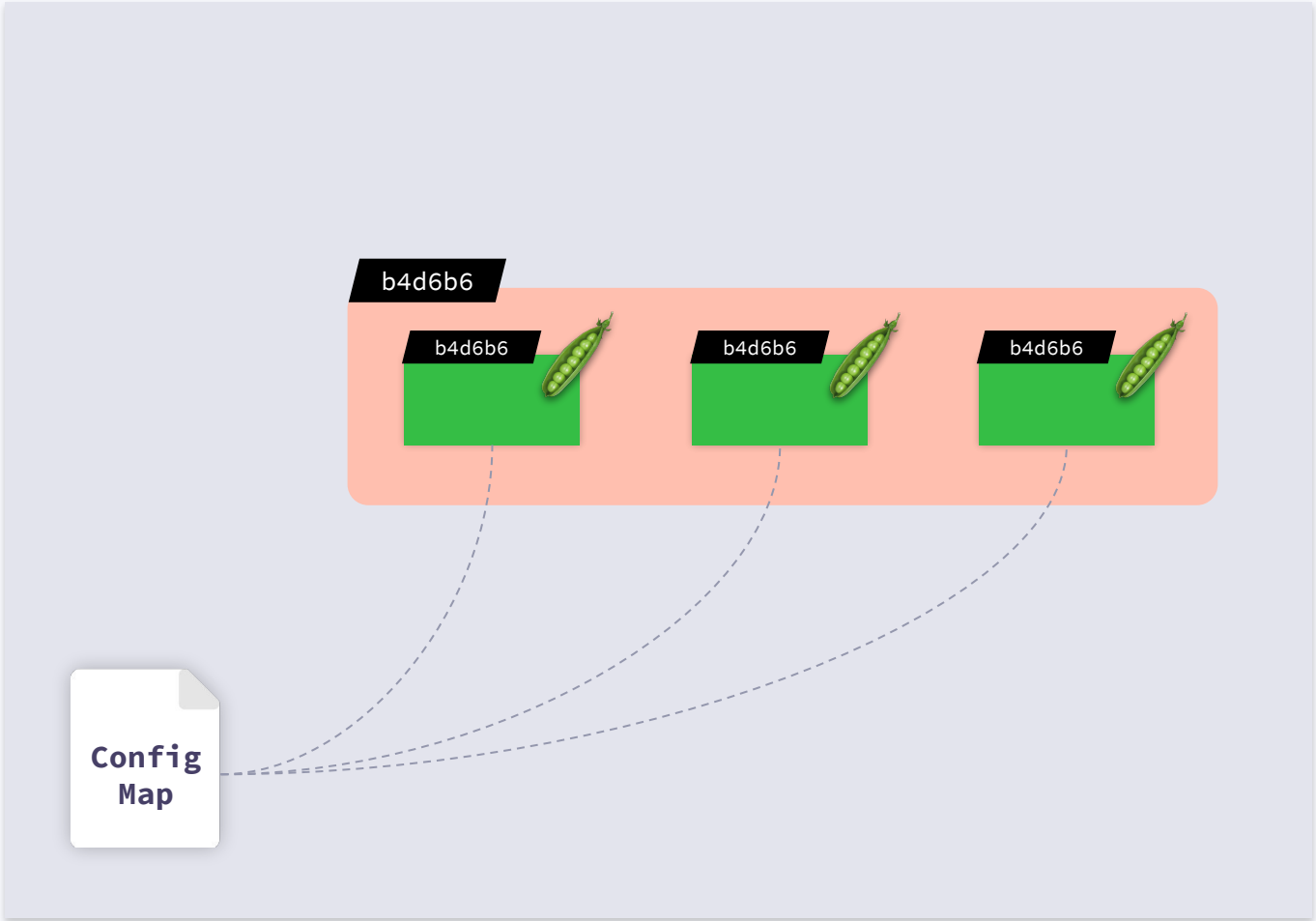

Config
Map

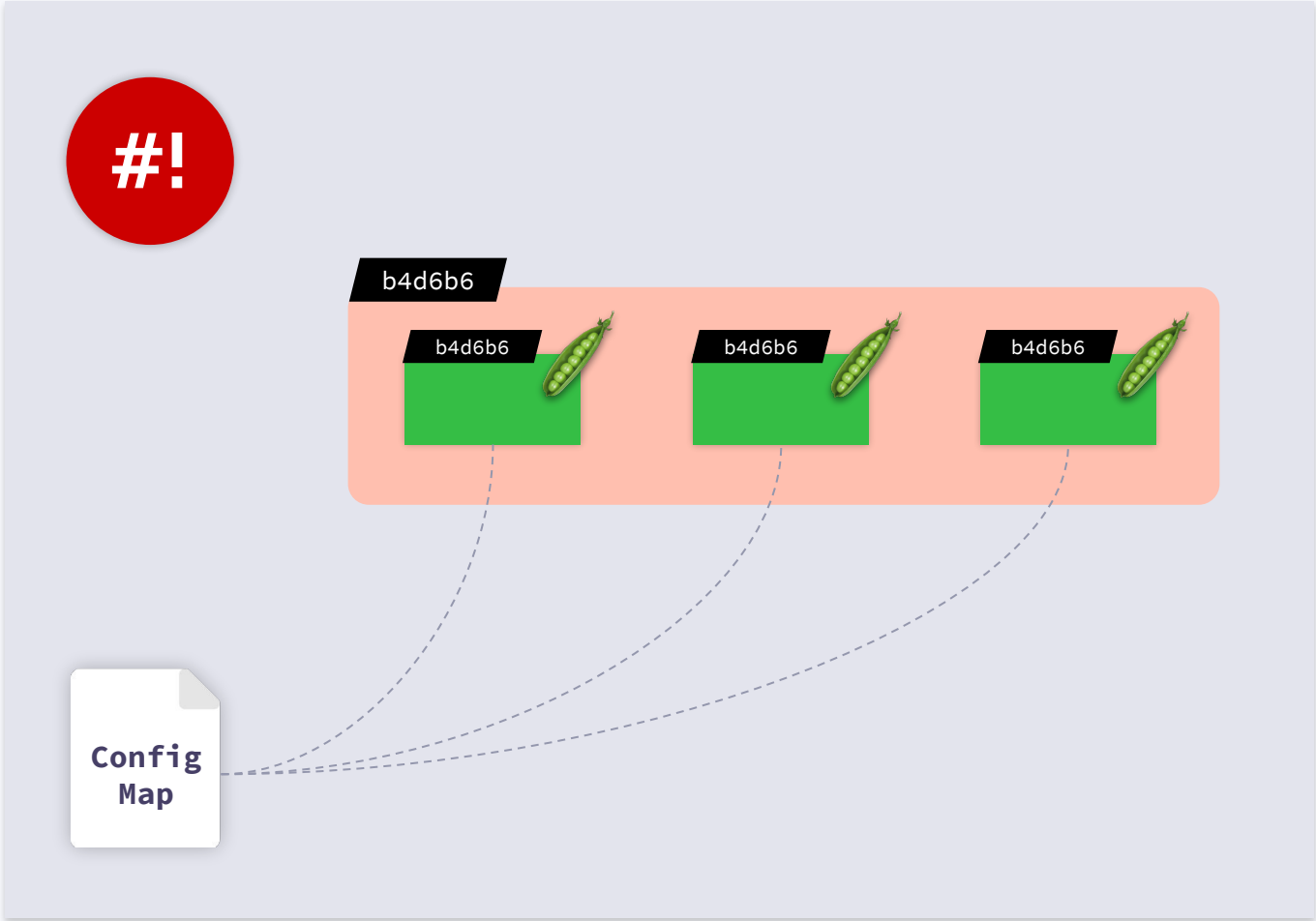

Config Map
b4d6b6


kubernetes
b4d6b6
b4d6b6
b4d6b6
b4d6b6
Config
Map


kubernetes
b4d6b6
b4d6b6
b4d6b6
b4d6b6
Config
Map


kubernetes
b4d6b6
b4d6b6
b4d6b6
b4d6b6
Config
Map

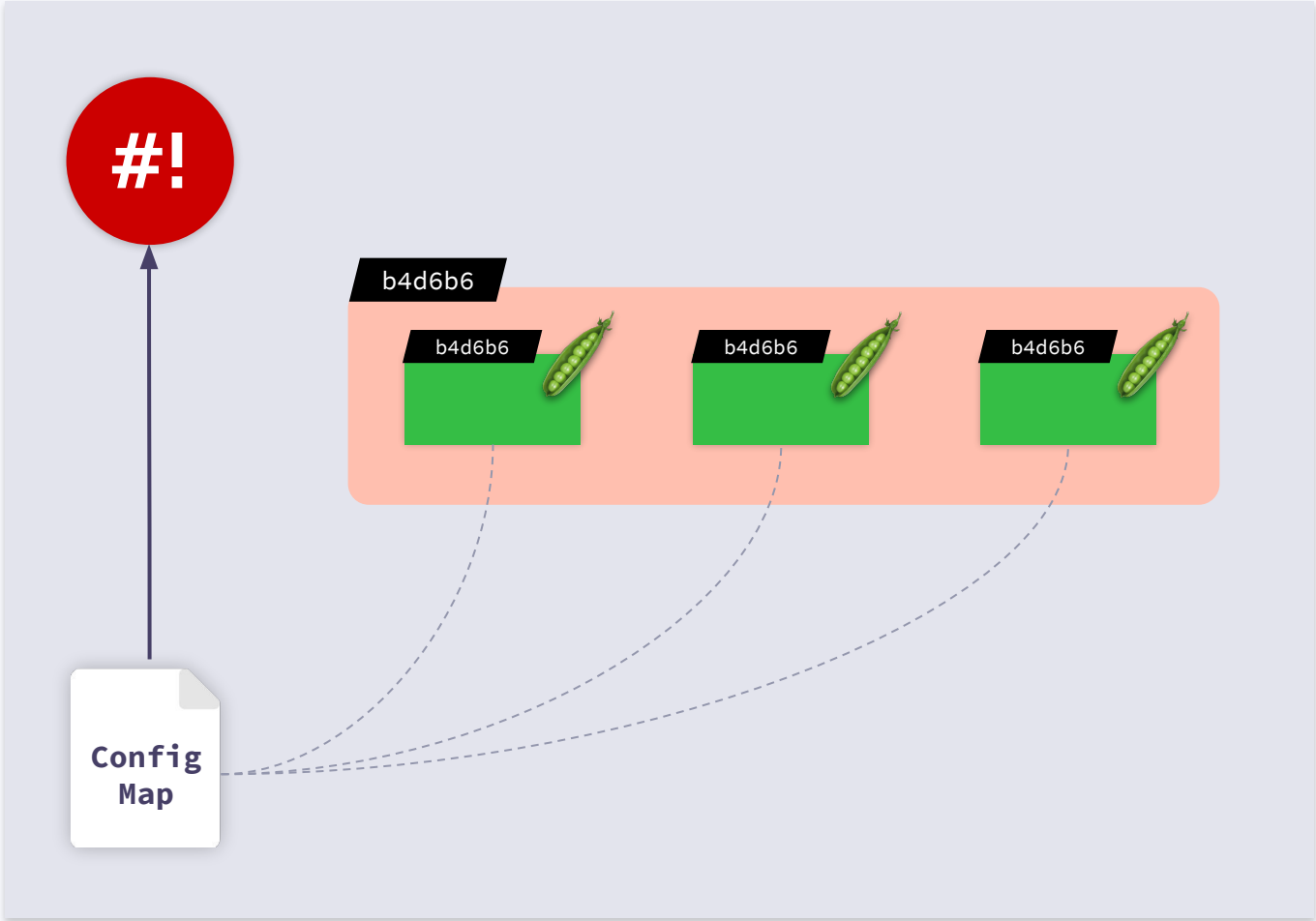

kubernetes
b4d6b6
b4d6b6
b4d6b6
b4d6b6
Config
Map

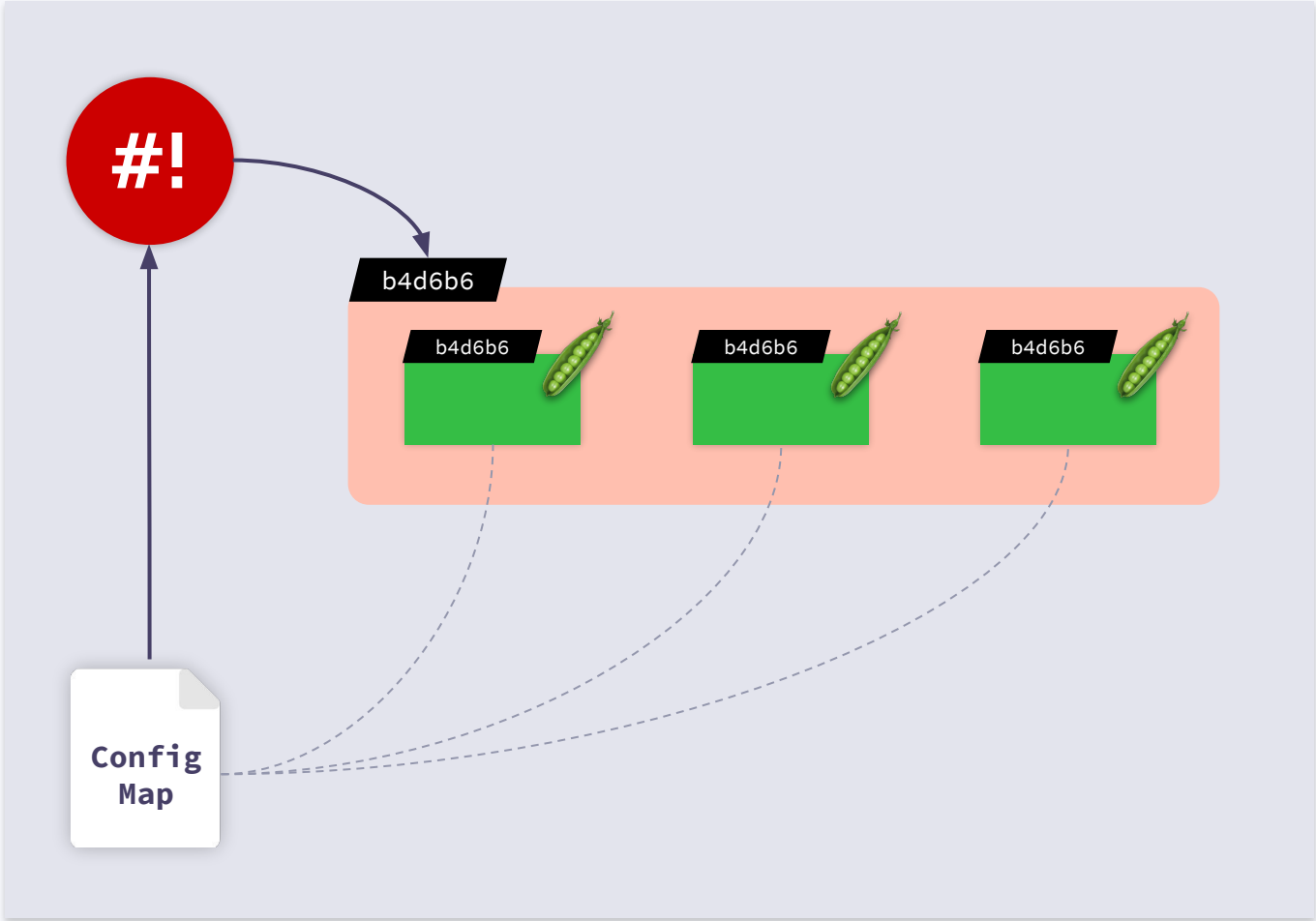


kubernetes

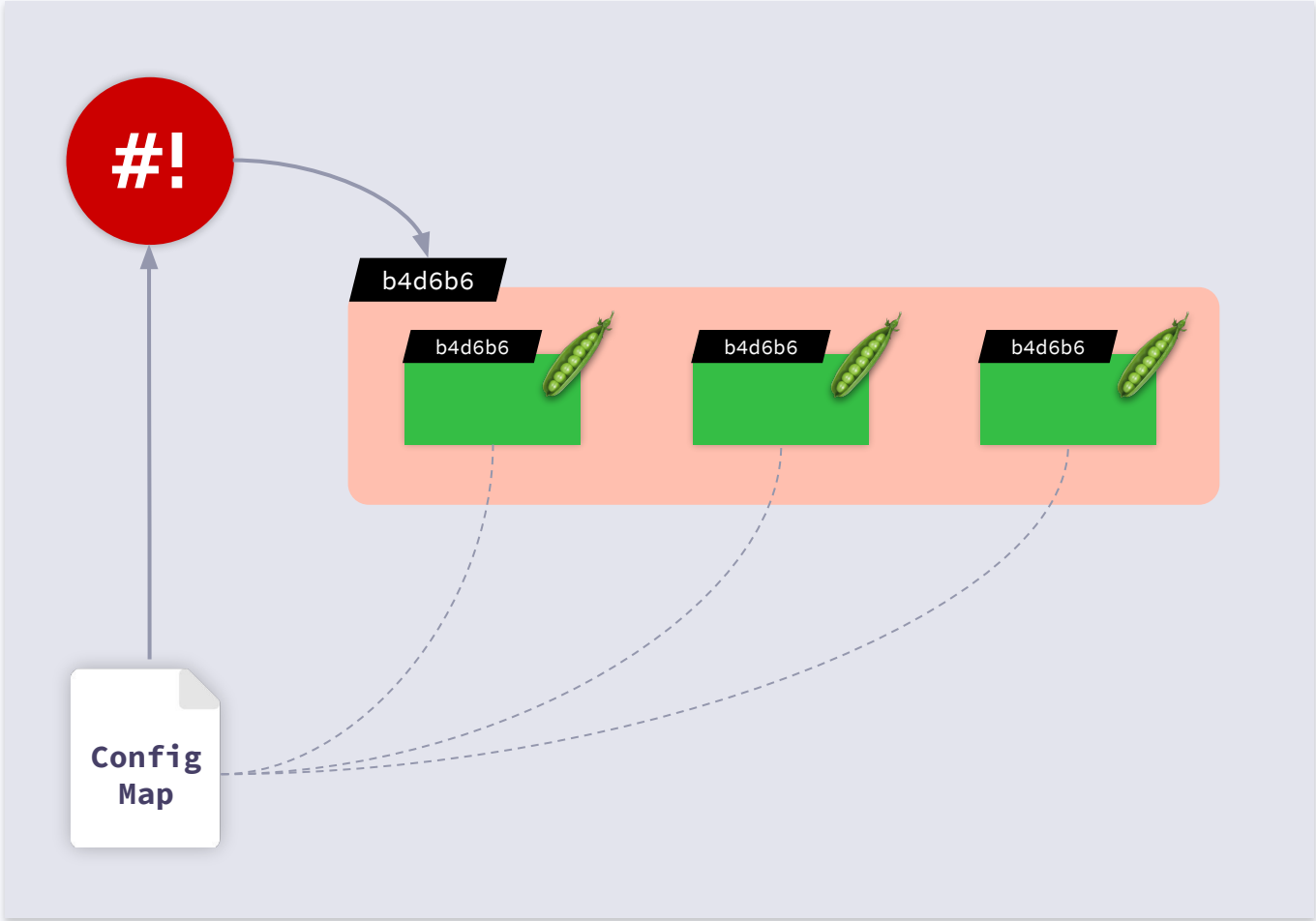
Go? Bash! Meet the Shell-operator

FLANT


kubernetes
#!
b4d6b6
b4d6b6
b4d6b6
b4d6b6
Config
Map

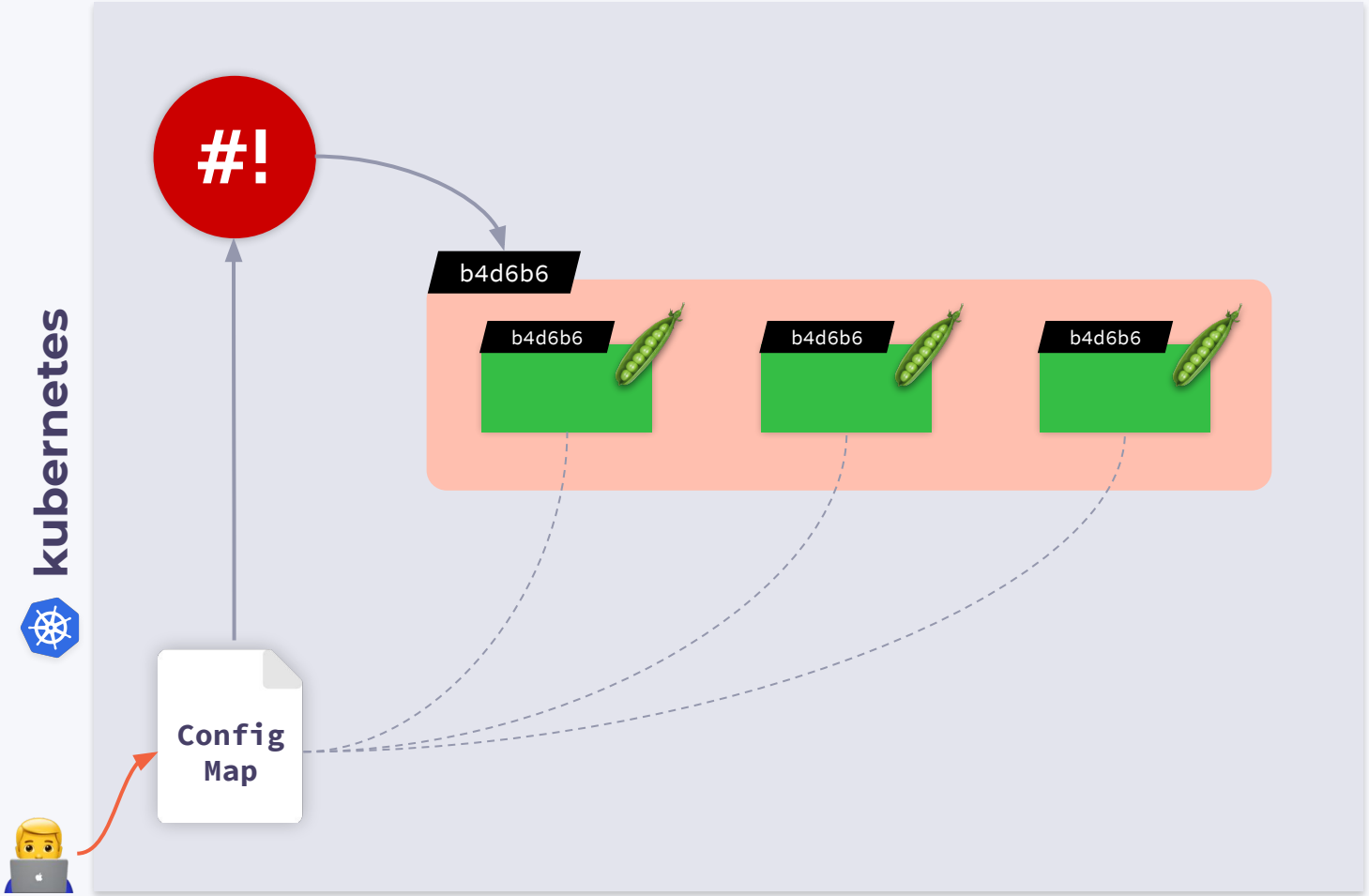


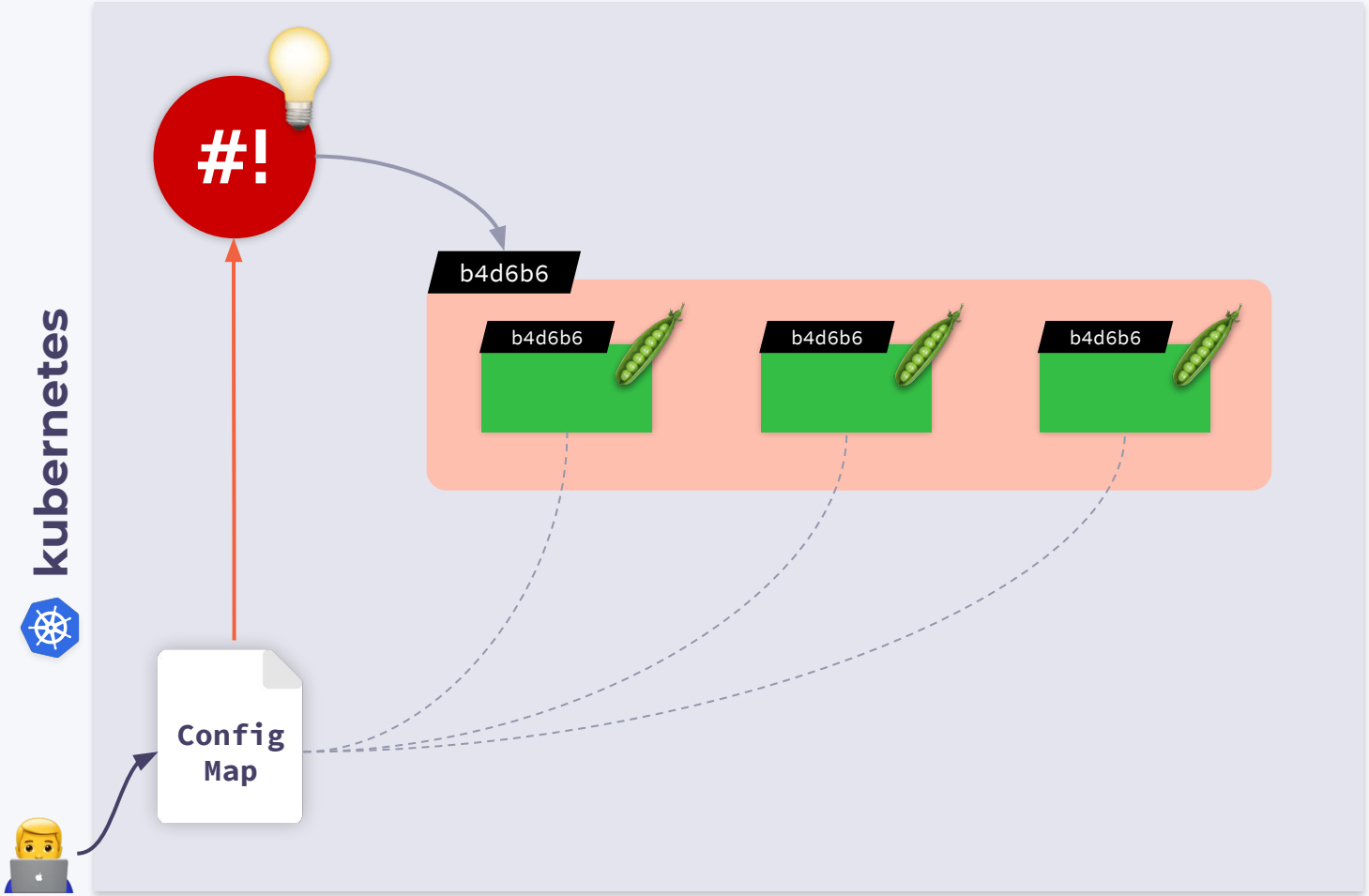
kubernetes

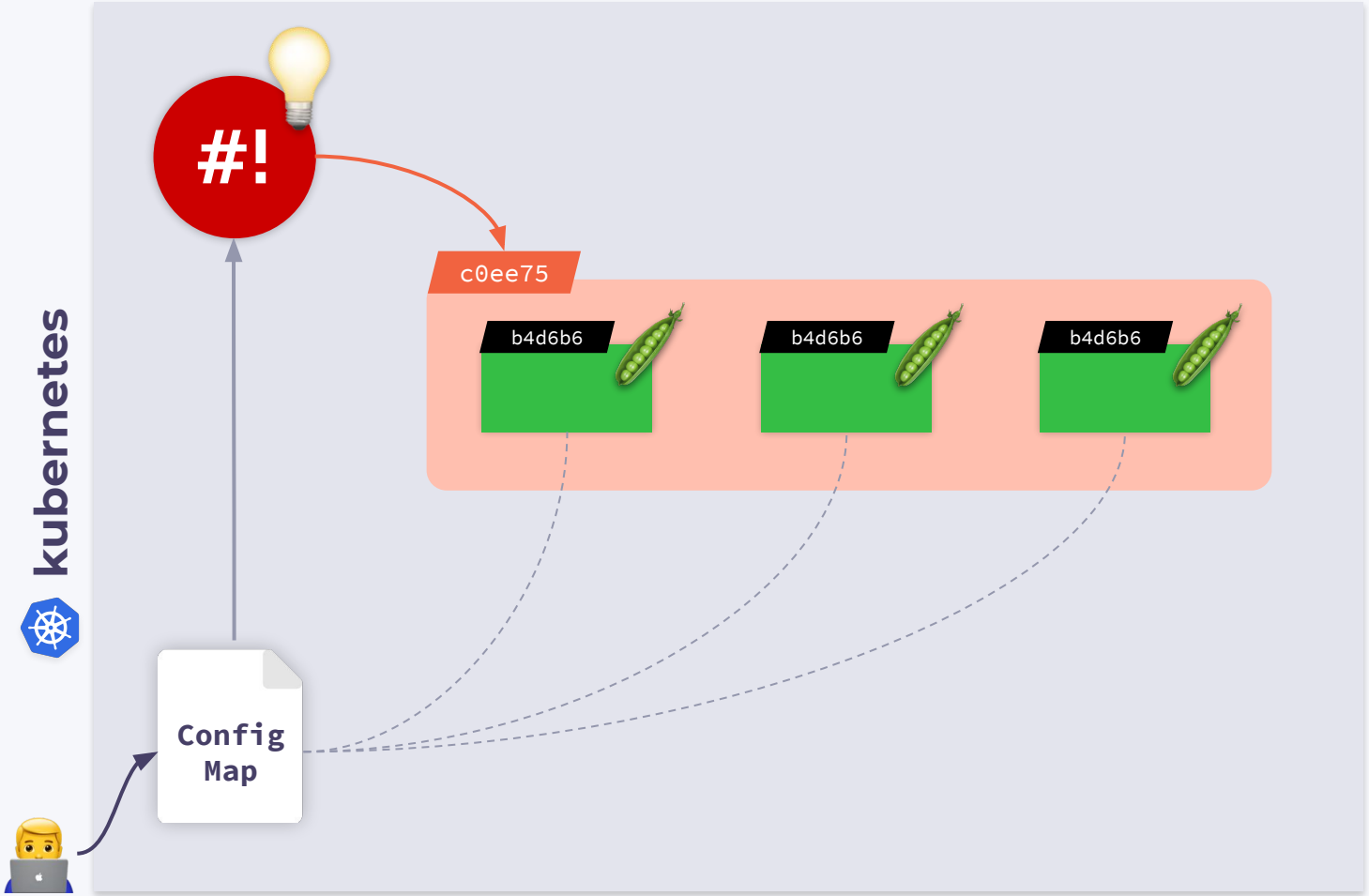


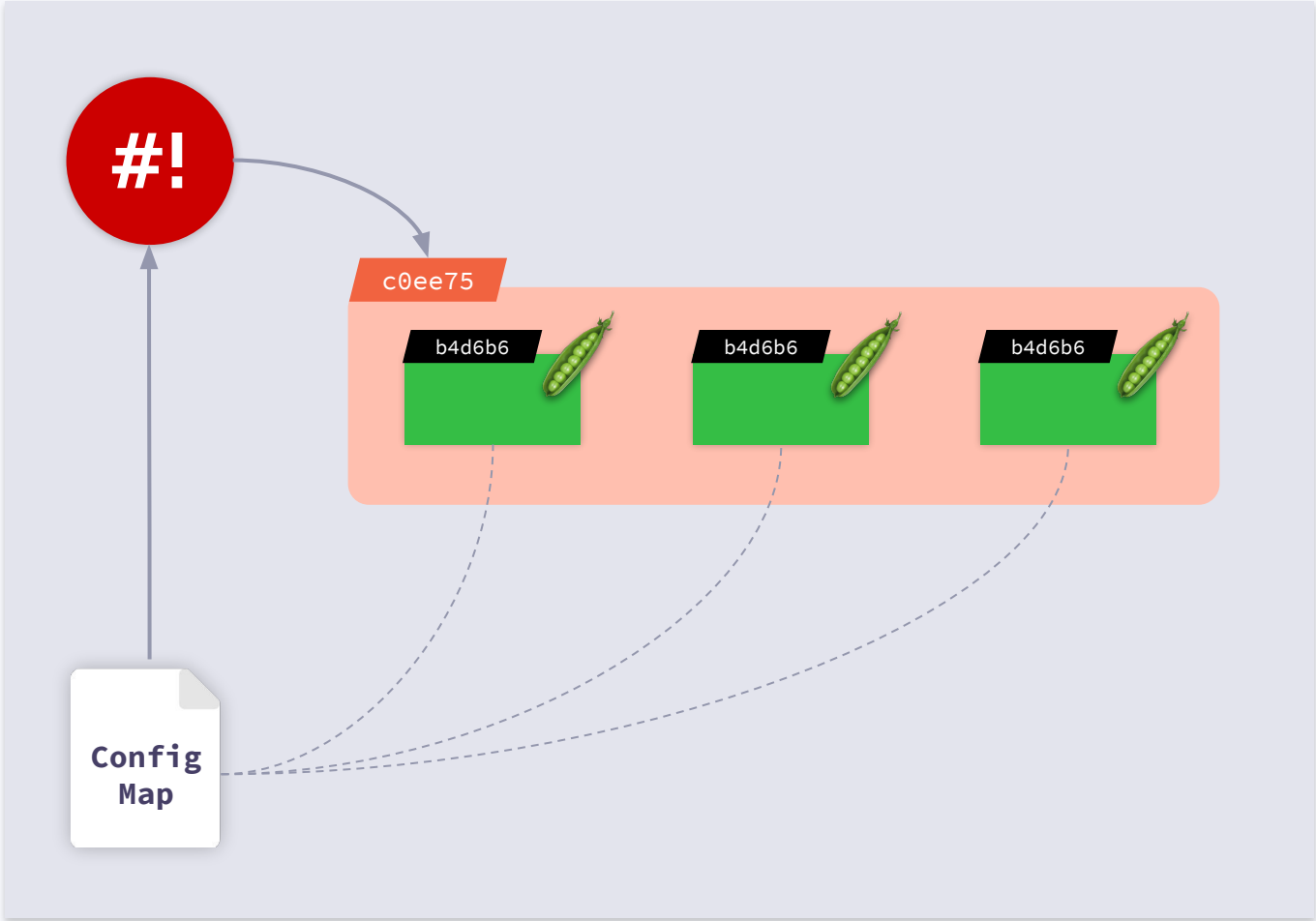
kubernetes

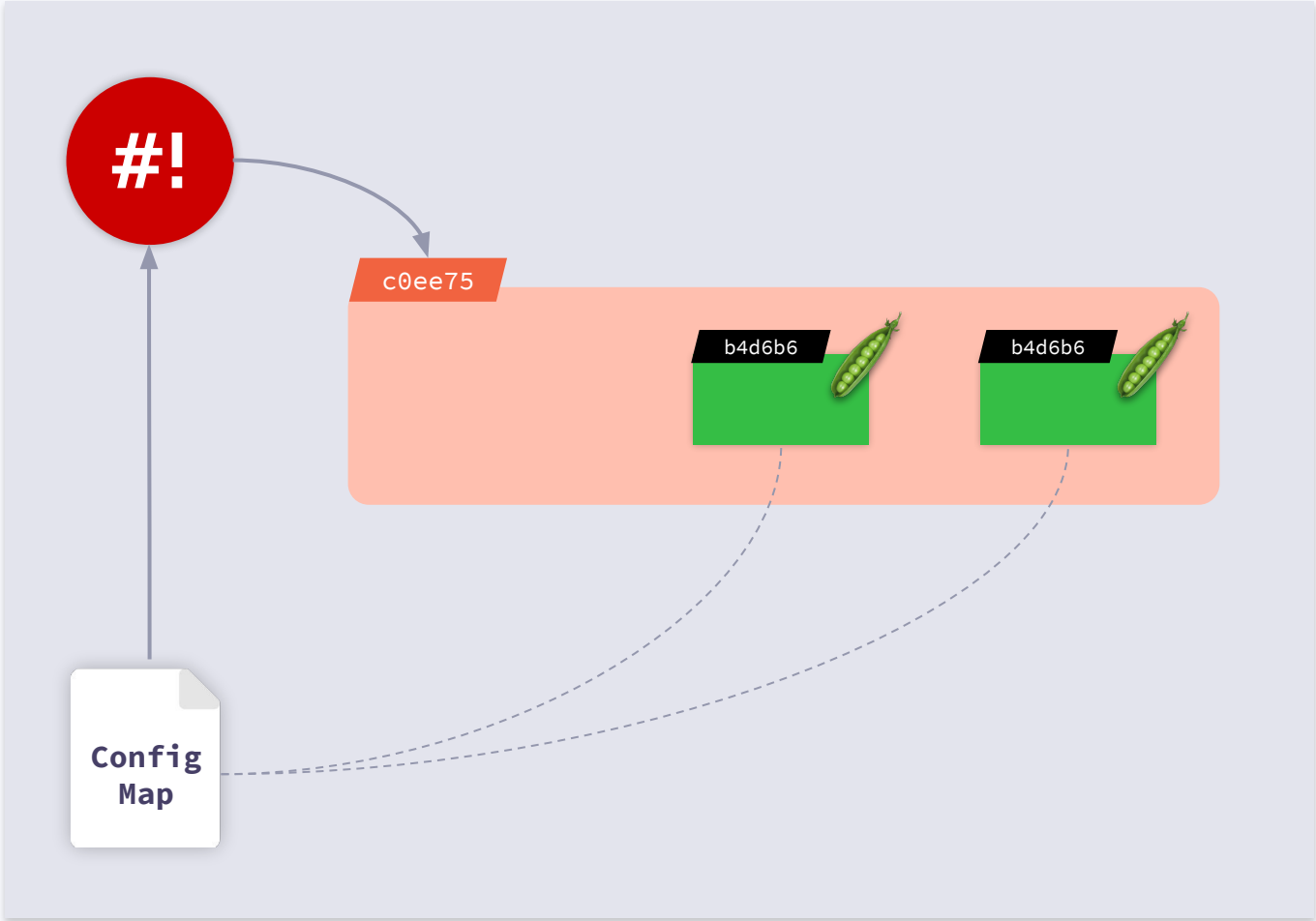
FLANT

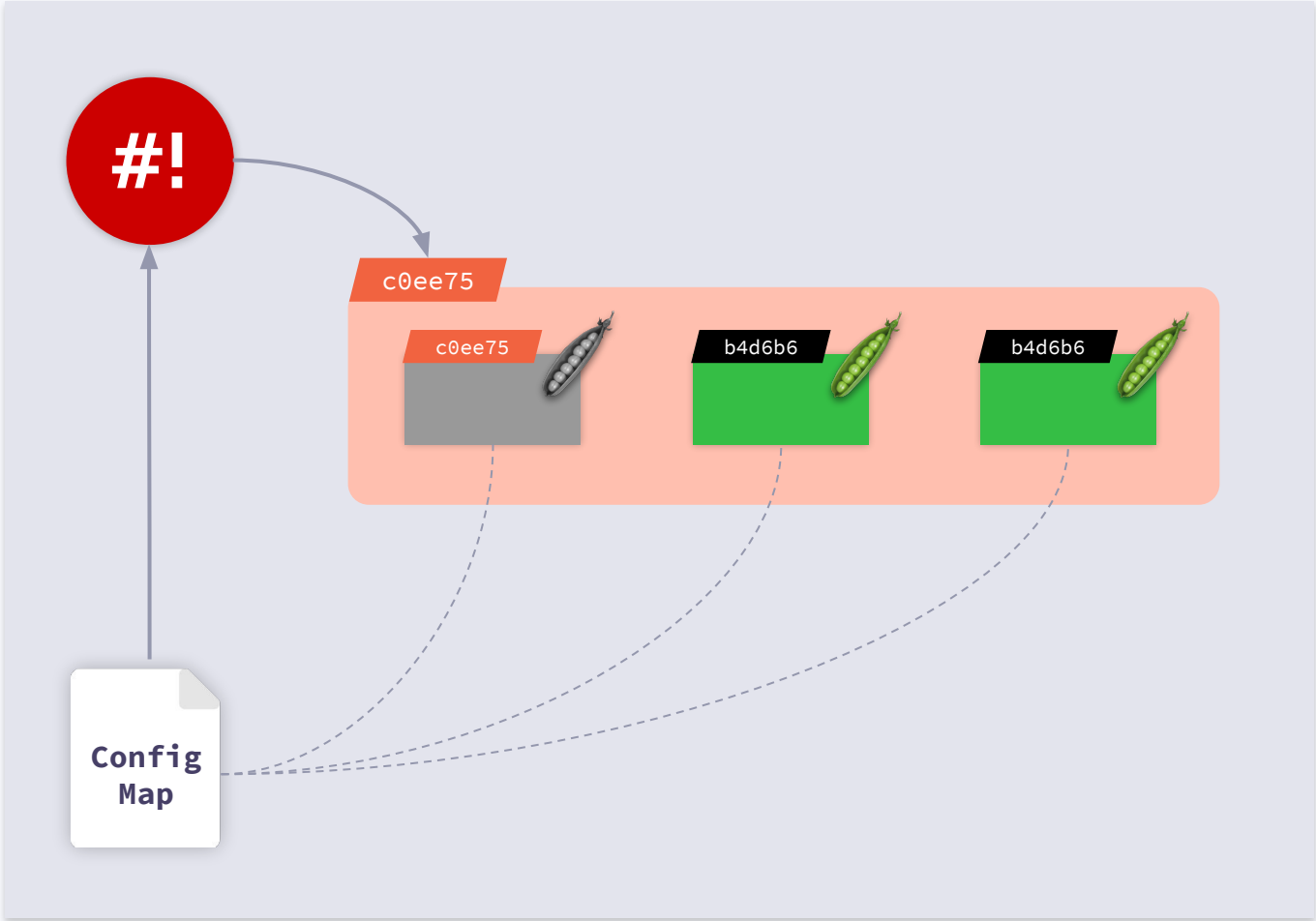

kubernetes
#!
b4d6b6
b4d6b6
b4d6b6
b4d6b6
Config
Map

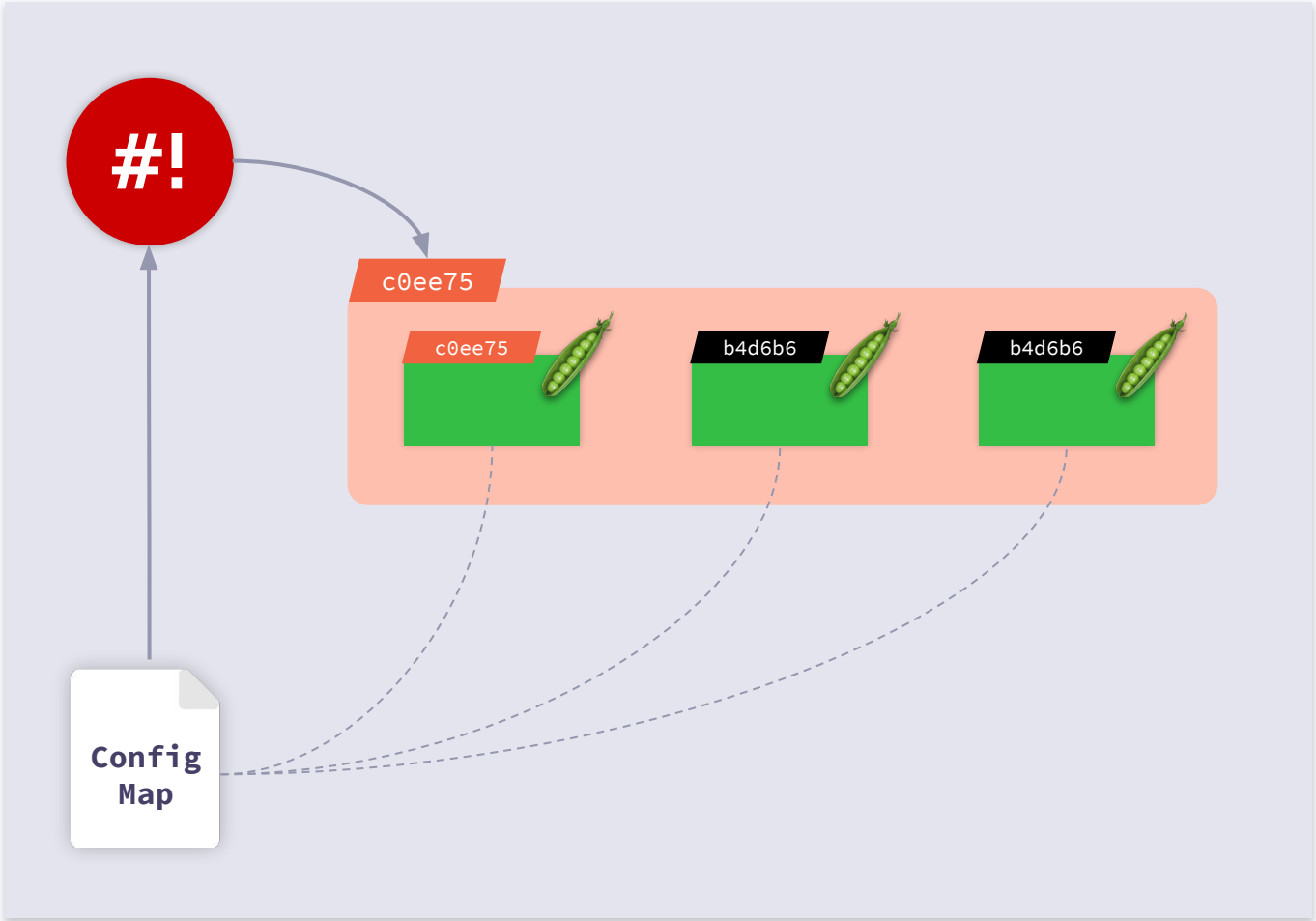

kubernetes
#!
b4d6b6
b4d6b6
b4d6b6
b4d6b6
Config
Map

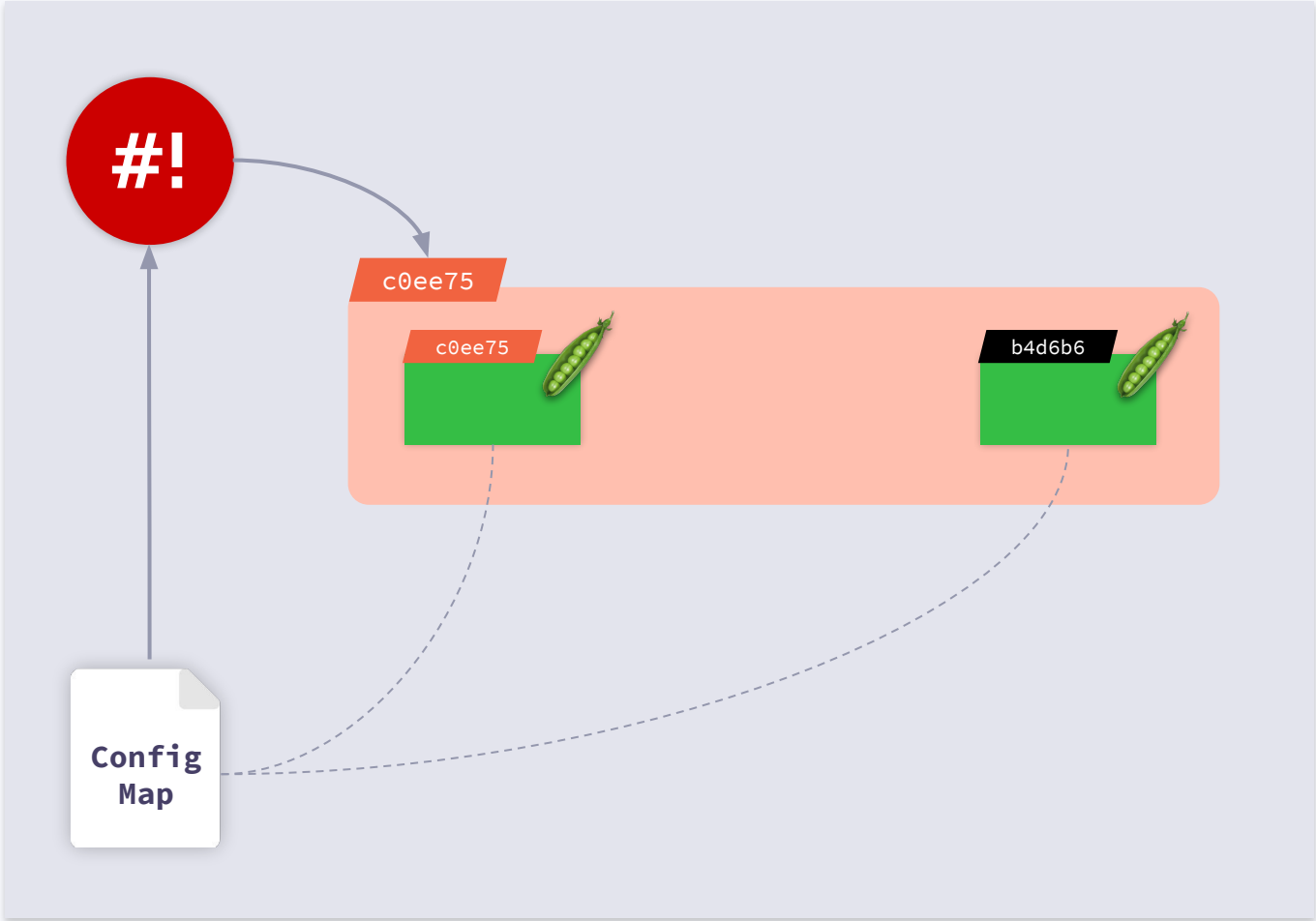

kubernetes
#!
c0ee75
b4d6b6
b4d6b6
b4d6b6
Config Map

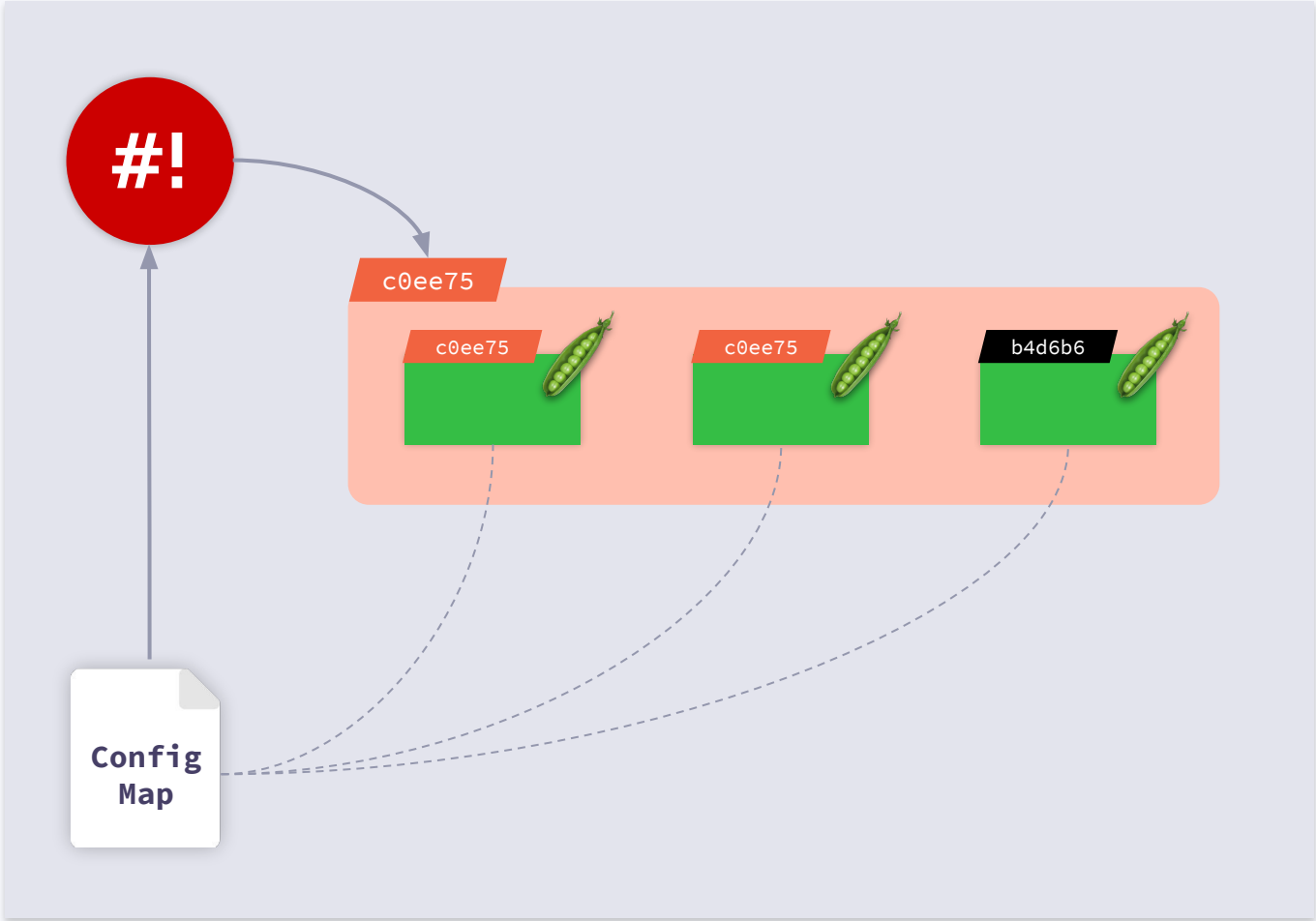

kubernetes
#!
c0ee75
b4d6b6
b4d6b6
b4d6b6
Config
Map

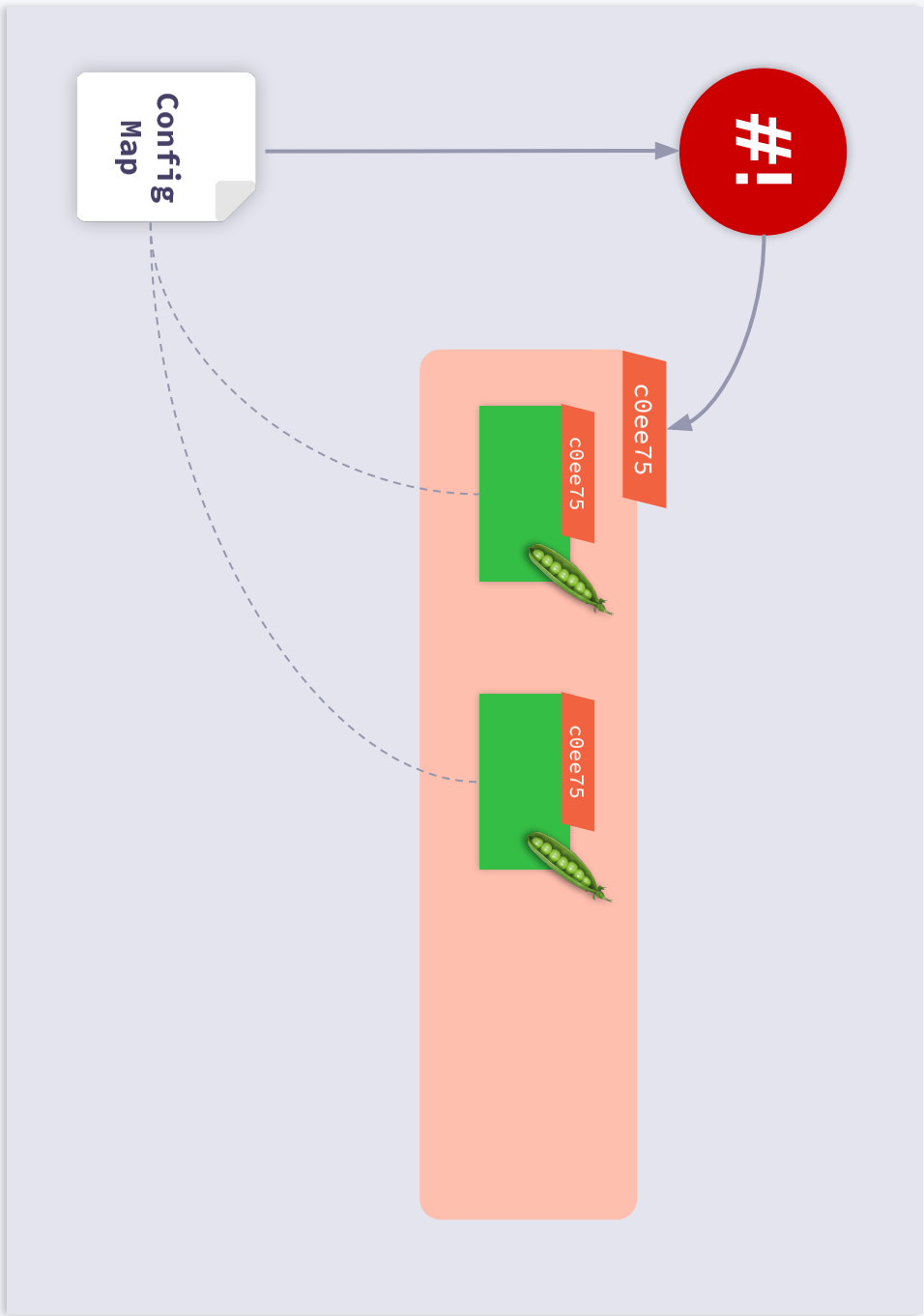

kubernetes
#!
c0ee75
b4d6b6
b4d6b6
Config
Map

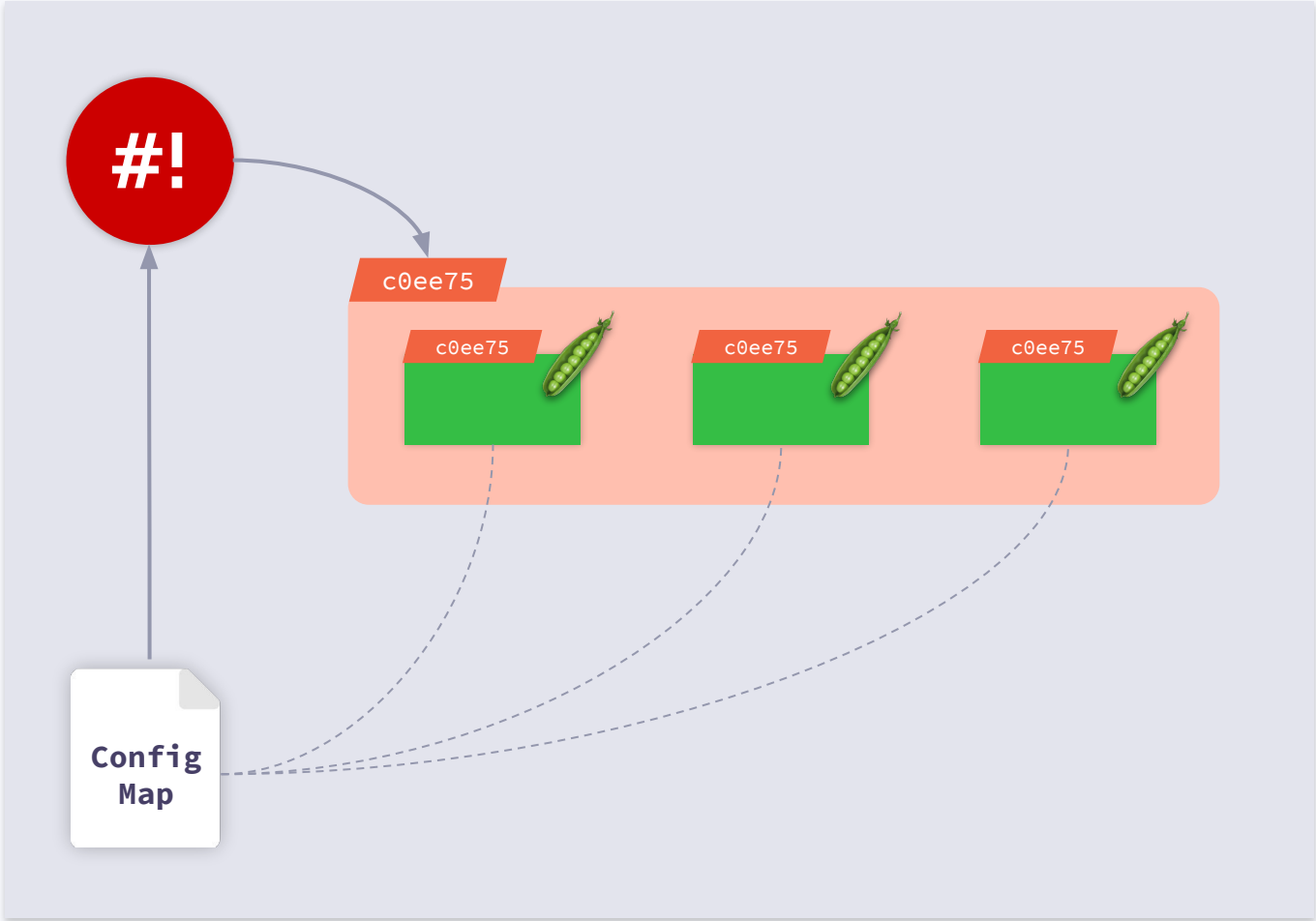

kubernetes
#!
c0ee75
c0ee75
b4d6b6
b4d6b6
Config
Map

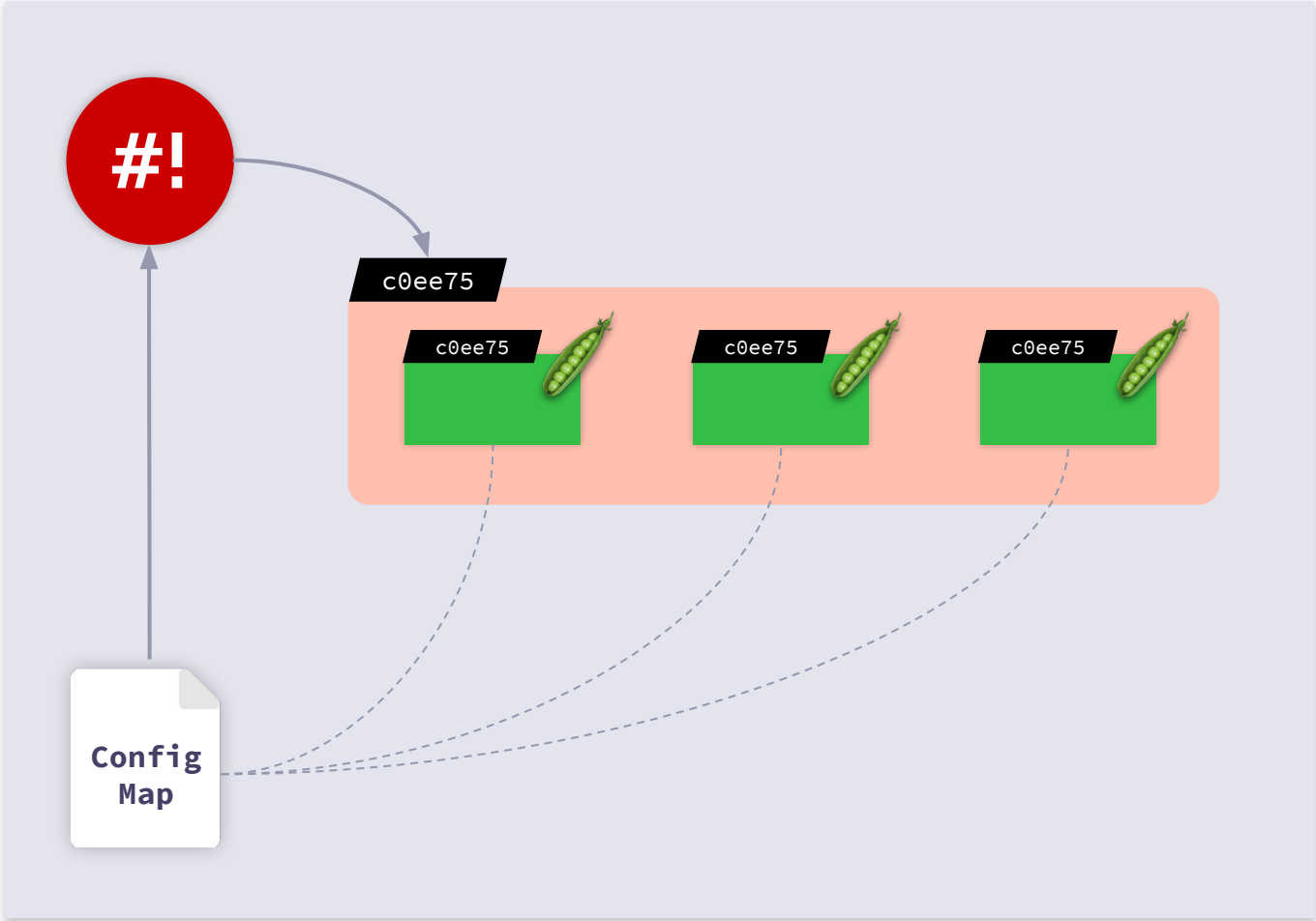

kubernetes
#!
c0ee75
c0ee75
b4d6b6
b4d6b6
Config Map


kubernetes
#!
c0ee75
c0ee75
b4d6b6
Config
Map


kubernetes
#!
c0ee75
c0ee75
c0ee75
b4d6b6
Config Map


kubernetes
Config Map
#!
c0ee75
c0ee75
c0ee75

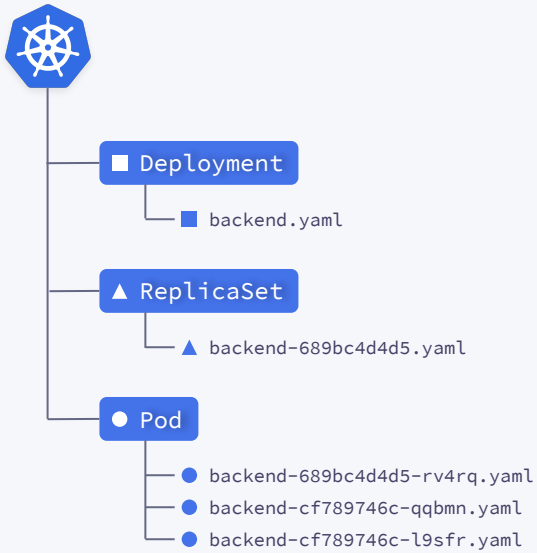

kubernetes
#!
c0ee75
c0ee75
c0ee75
c0ee75
Config
Map

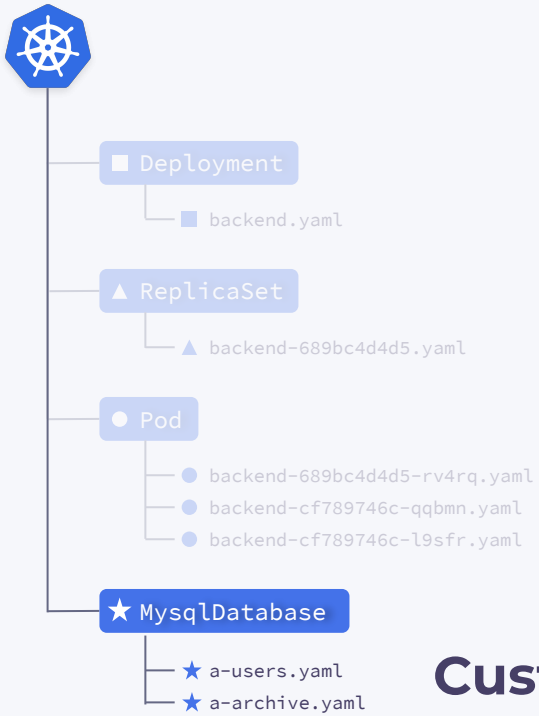


kubernetes

Go? Bash! Meet the Shell-operator

FLANT


Deployment
backend.yaml
ReplicaSet
backend-689bc4d4d5.yaml
Pod
backend-689bc4d4d5-rv4rq.yaml
backend-cf789746c-qqbmn.yaml
backend-cf789746c-l9sfr.yaml



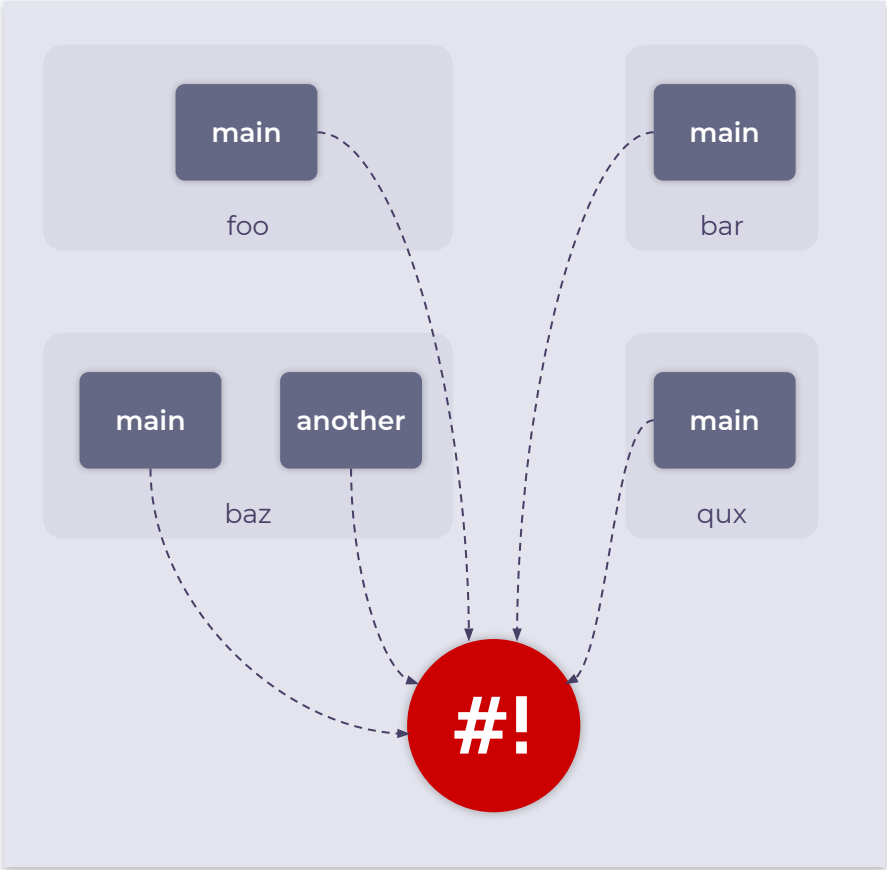
Custom Resource Definitions

```yaml
apiVersion: example.com/v1alpha1
kind: MysqlDatabase
metadata:
  name: foo
  namespace: bar
```

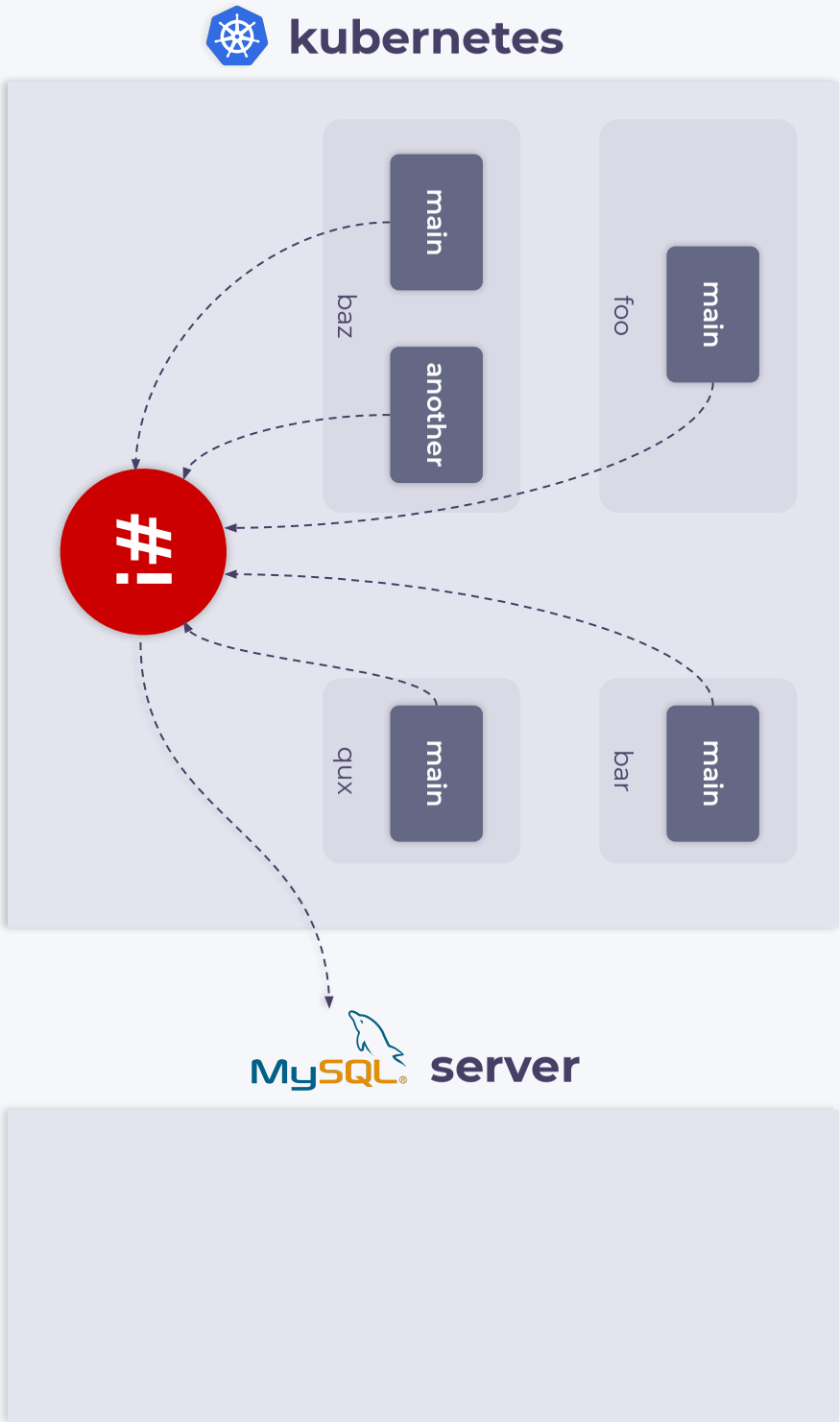

kubernetes

kubernetes

kubernetes



FLANT


kubernetes
main
another
baz
foo
main
#!
qux
main
bar
main
MySQL server

kubernetes
main
foo
main
bar
main
another
baz
main
qux
#!
MySQL
server
foo.main
bar.main
baz.main
baz.another
qux.main

```
configVersion: v1
kubernetes:
- name: nodes
  apiVersion: v1
  kind: Node
  jqFilter: |
    {
      name: .metadata.name,
      ip: (
       .status.addresses[] |
         select(.type == "InternalIP") |
         .address
      )
    }
  group: main
  keepFullObjectsInMemory: false
```


```
configVersion: v1
kubernetes:
- name: nodes
  apiVersion: v1
  kind: Node
  jqFilter: |
    {
      name: .metadata.name,
      ip: (
       .status.addresses[] |
         select(.type == "InternalIP") |
         .address
      )
    }
  group: main
  keepFullObjectsInMemory: false
```


```
configVersion: v1
kubernetes:
- name: nodes
  apiVersion: v1
  kind: Node
  jqFilter: |
    {
      name: .metadata.name,
      ip: (
       .status.addresses[] |
         select(.type == "InternalIP") |
         .address
      )
    }
  group: main
  keepFullObjectsInMemory: false
```


```
configVersion: v1
kubernetes:
- name: nodes
  apiVersion: v1
  kind: Node
  jqFilter: |
    {
      name: .metadata.name,
      ip: (
       .status.addresses[] |
         select(.type == "InternalIP") |
         .address
      )
    }
  group: main
  keepFullObjectsInMemory: false
  executeHookOnEvent: []
```

```
configVersion: v1
kubernetes:
- name: nodes
  apiVersion: v1
  kind: Node
  jqFilter: |
    {
      name: .metadata.name,
      ip: (
       .status.addresses[] |
         select(.type == "InternalIP") |
         .address
      )
    }
  group: main
  keepFullObjectsInMemory: false
  executeHookOnEvent: []
schedule:
- name: every_minute
  group: main
  crontab: "* * * * *"
```


```
function __main__() {
  for i in $(seq 0 "$(context::jq -r '(.snapshots.nodes | length) - 1')"); do
    node_name="$(context::jq -r '.snapshots.nodes['"$i"'].filterResult.name')"
    node_ip="$(context::jq -r '.snapshots.nodes['"$i"'].filterResult.ip')"

    packets_lost=0
    if ping -c 1 $node_ip -t 1 ; then
      packets_lost=1
    fi

    cat <<-END
      {
        "name": "node_packets_lost",
        "add": $packets_lost,
        "labels": {
          "node": $node_name,
        }
      }
      END  >> $METRICS_PATH
  done
}
```

```
function __main__() {
  for i in $(seq 0 "$(context::jq -r '(.snapshots.nodes | length) - 1')"); do
    node_name="$(context::jq -r '.snapshots.nodes['"$i"'].filterResult.name')"
    node_ip="$(context::jq -r '.snapshots.nodes['"$i"'].filterResult.ip')"

    packets_lost=0
    if ping -c 1 $node_ip -t 1 ; then
      packets_lost=1
    fi

    cat <<-END
      {
        "name": "node_packets_lost",
        "add": $packets_lost,
        "labels": {
          "node": $node_name,
        }
      }
      END  >> $METRICS_PATH
  done
}
```

```
function __main__() {
  for i in $(seq 0 "$(context::jq -r '(.snapshots.nodes | length) - 1')"); do
    node_name="$(context::jq -r '.snapshots.nodes['"$i"'].filterResult.name')"
    node_ip="$(context::jq -r '.snapshots.nodes['"$i"'].filterResult.ip')"

    packets_lost=0
    if ping -c 1 $node_ip -t 1 ; then
      packets_lost=1
    fi

    cat <<-END
      {
        "name": "node_packets_lost",
        "add": $packets_lost,
        "labels": {
          "node": $node_name,
        }
      }
      END  >> $METRICS_PATH
  done
}
```

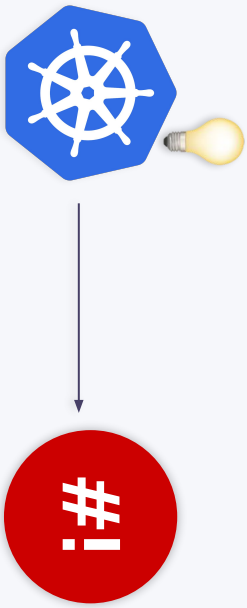
```
function __main__() {
  for i in $(seq 0 "$(context::jq -r '(.snapshots.nodes | length) - 1')"); do
    node_name="$(context::jq -r '.snapshots.nodes['"$i"'].filterResult.name')"
    node_ip="$(context::jq -r '.snapshots.nodes['"$i"'].filterResult.ip')"

    packets_lost=0
    if ping -c 1 $node_ip -t 1 ; then
      packets_lost=1
    fi

    cat <<-END
      {
        "name": "node_packets_lost",
        "add": $packets_lost,
        "labels": {
          "node": $node_name,
        }
      }
      END  >> $METRICS_PATH
  done
}
```

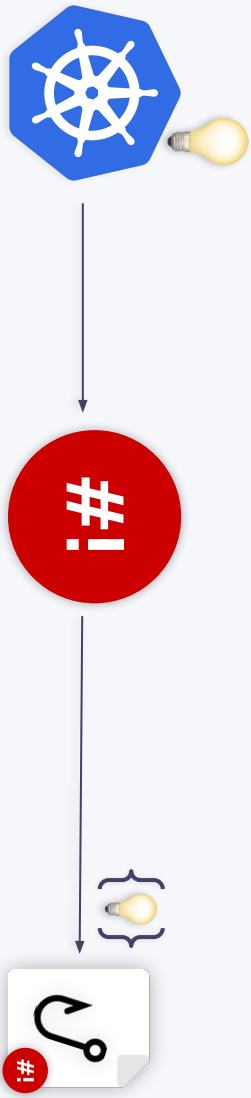
```
function __main__() {
  for i in $(seq 0 "$(context::jq -r '(.snapshots.nodes | length) - 1')"); do
    node_name="$(context::jq -r '.snapshots.nodes['"$i"'].filterResult.name')"
    node_ip="$(context::jq -r '.snapshots.nodes['"$i"'].filterResult.ip')"

    packets_lost=0
    if ping -c 1 $node_ip -t 1 ; then
      packets_lost=1
    fi

    cat <<-END
      {
        "name": "node_packets_lost",
        "add": $packets_lost,
        "labels": {
          "node": $node_name,
        }
      }
      END  >> $METRICS_PATH
  done
}
```

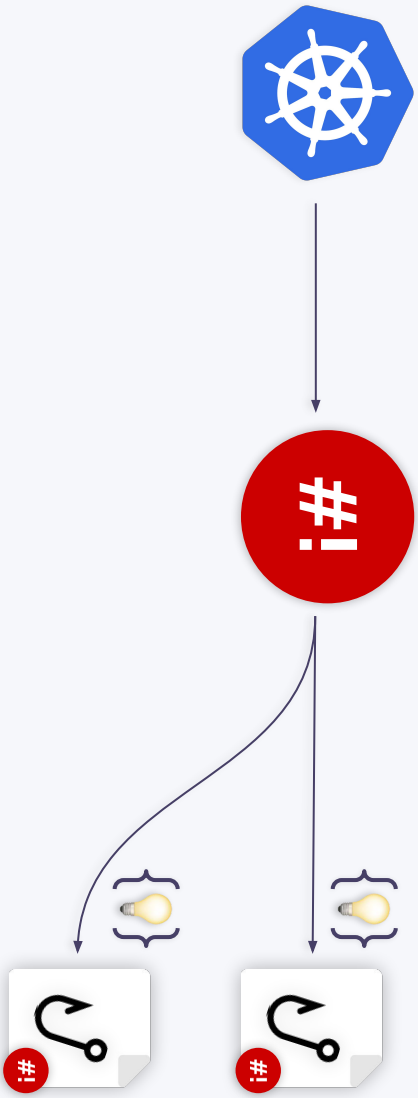
```
function __main__() {
  for i in $(seq 0 "$(context::jq -r '(.snapshots.nodes | length) - 1')"); do
    node_name="$(context::jq -r '.snapshots.nodes['"$i"'].filterResult.name')"
    node_ip="$(context::jq -r '.snapshots.nodes['"$i"'].filterResult.ip')"

    packets_lost=0
    if ping -c 1 $node_ip -t 1 ; then
      packets_lost=1
    fi

    cat <<-END
      {
        "name": "node_packets_lost",
        "add": $packets_lost,
        "labels": {
          "node": $node_name,
        }
      }
      END  >> $METRICS_PATH
  done
}
```

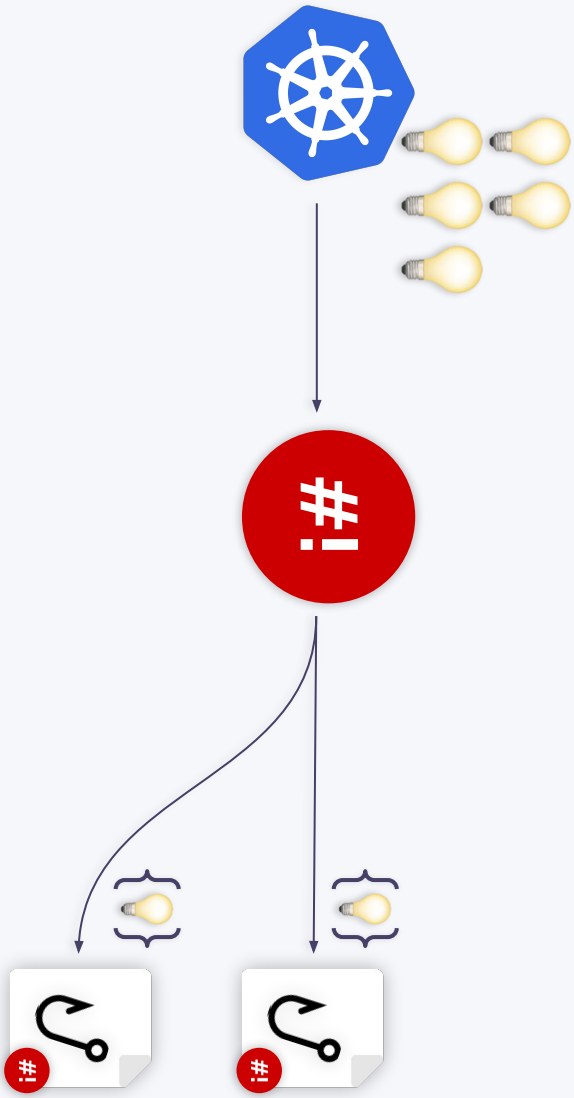
```
function __main__() {
  for i in $(seq 0 "$(context::jq -r '(.snapshots.nodes | length) - 1')"); do
    node_name="$(context::jq -r '.snapshots.nodes['"$i"'].filterResult.name')"
    node_ip="$(context::jq -r '.snapshots.nodes['"$i"'].filterResult.ip')"

    packets_lost=0
    if ping -c 1 $node_ip -t 1 ; then
      packets_lost=1
    fi

    cat <<-END
      {
        "name": "node_packets_lost",
        "add": $packets_lost,
        "labels": {
          "node": $node_name,
        }
      }
      END  >> $METRICS_PATH
  done
}
```

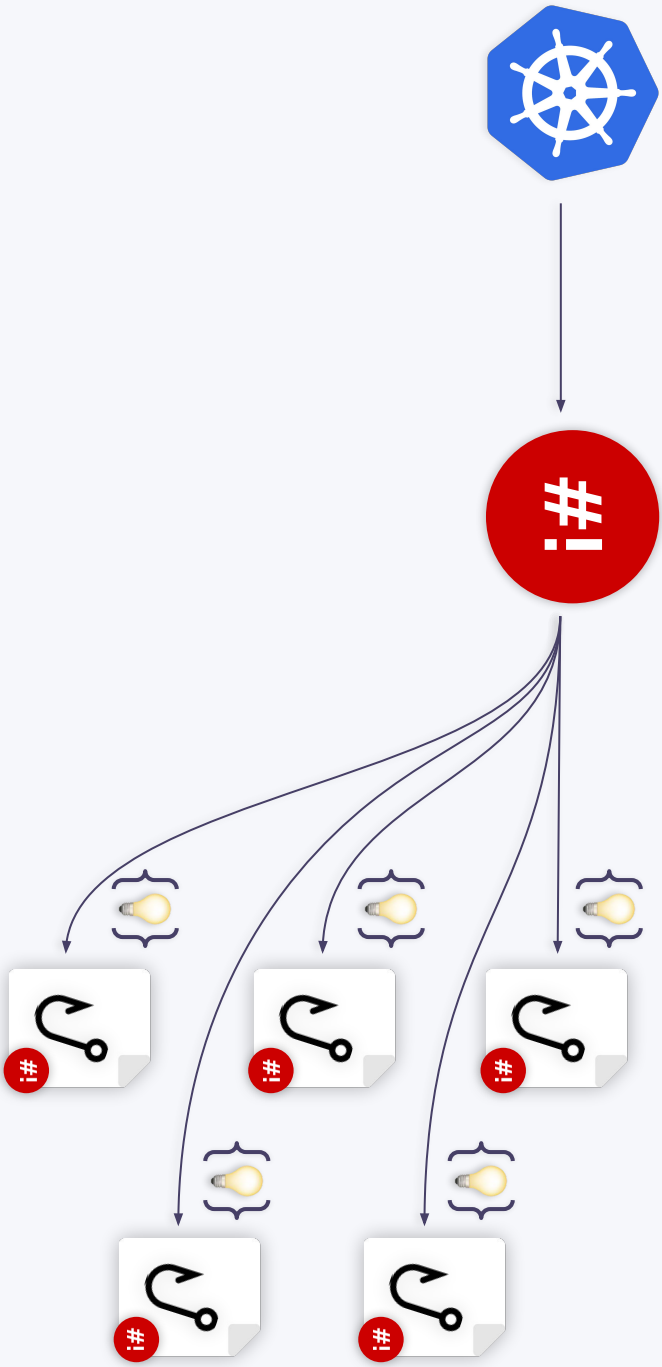
```
function __main__() {
  for i in $(seq 0 "$(context::jq -r '(.snapshots.nodes | length) - 1')"); do
    node_name="$(context::jq -r '.snapshots.nodes['"$i"'].filterResult.name')"
    node_ip="$(context::jq -r '.snapshots.nodes['"$i"'].filterResult.ip')"

    packets_lost=0
    if ping -c 1 $node_ip -t 1 ; then
      packets_lost=1
    fi

    cat <<-END
      {
        "name": "node_packets_lost",
        "add": $packets_lost,
        "labels": {
          "node": $node_name,
        }
      }
      END  >> $METRICS_PATH
  done
}
```

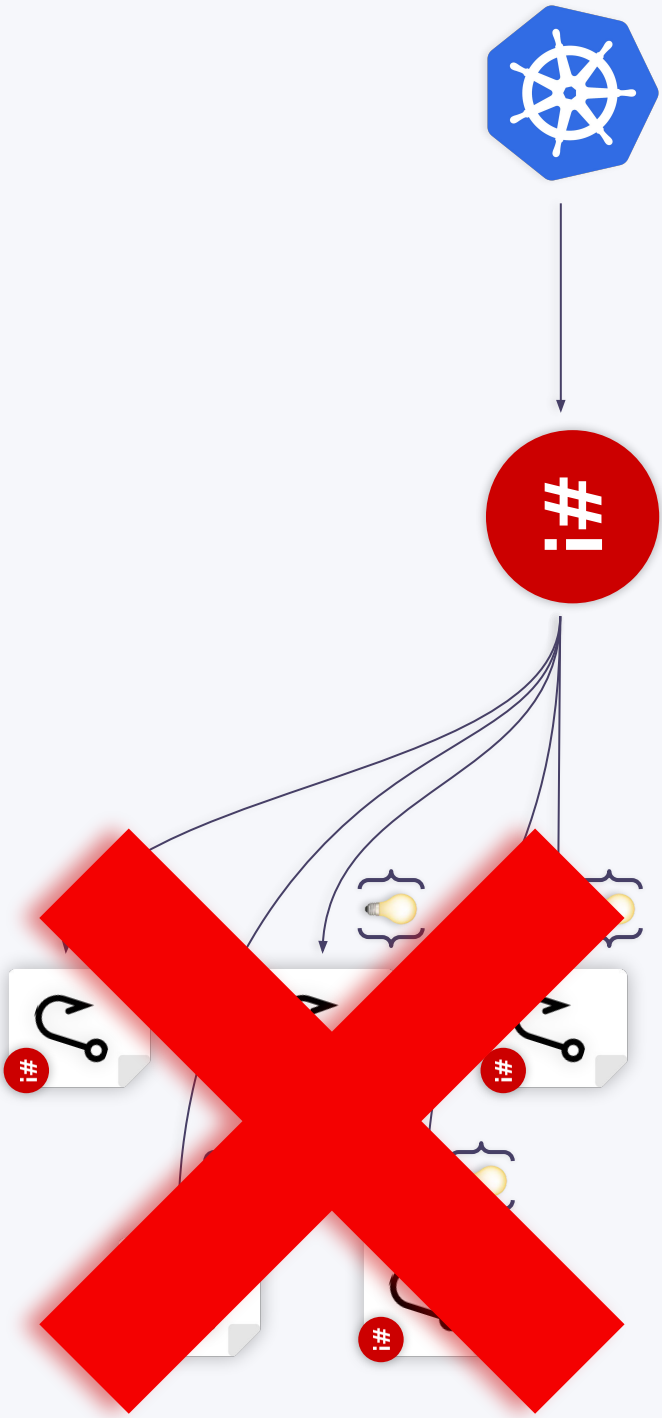


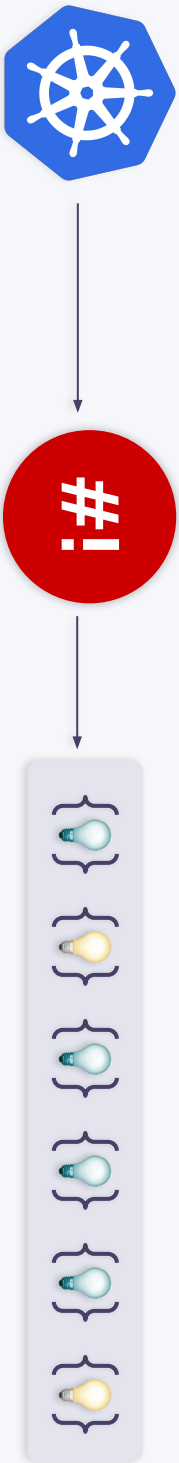


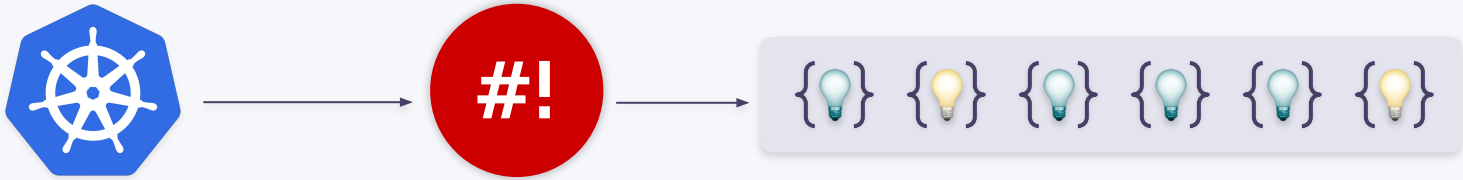


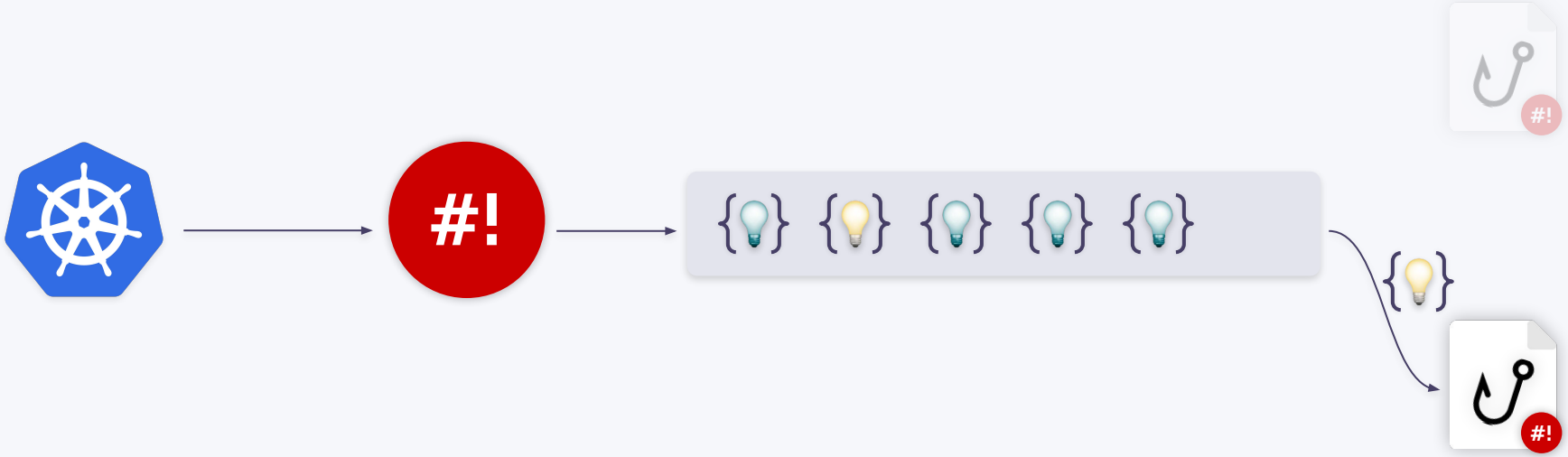


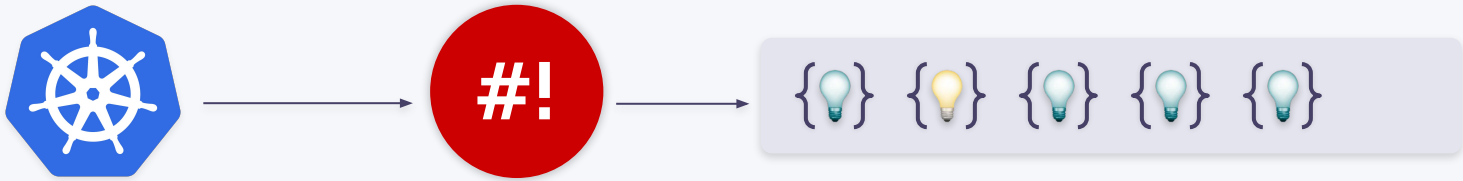


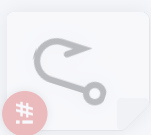


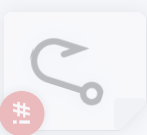


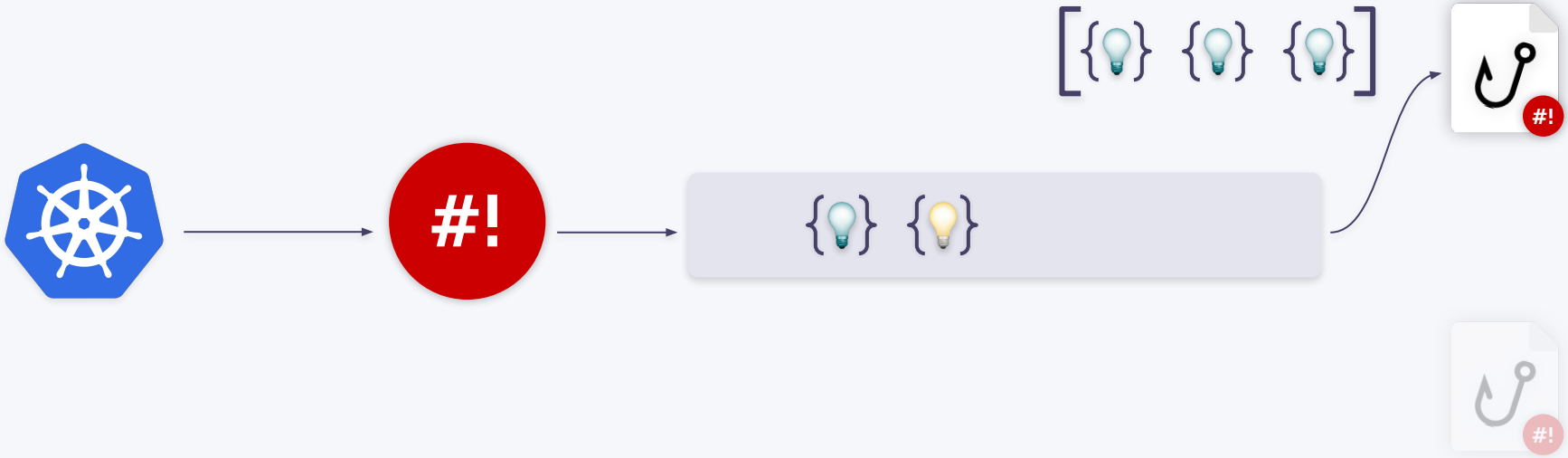


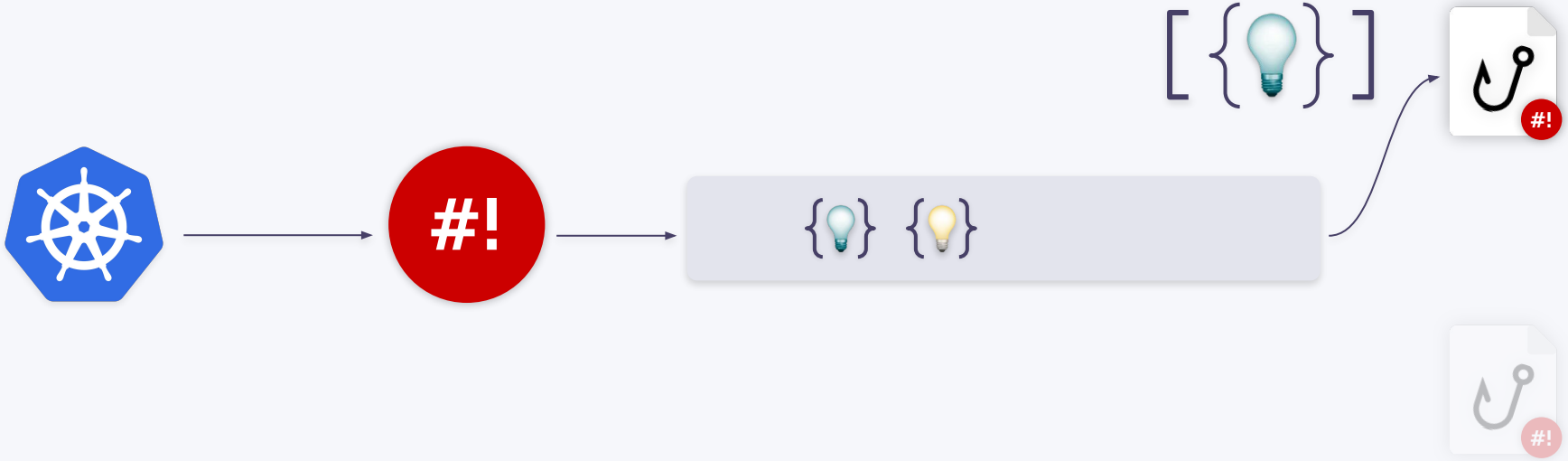










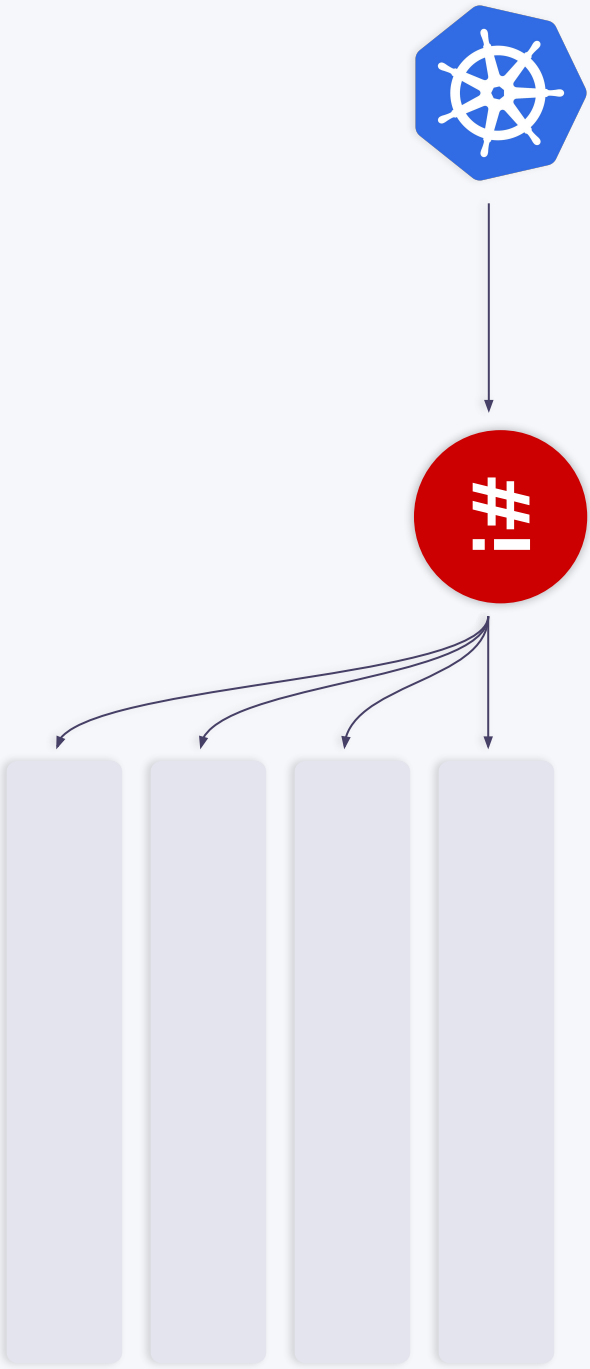


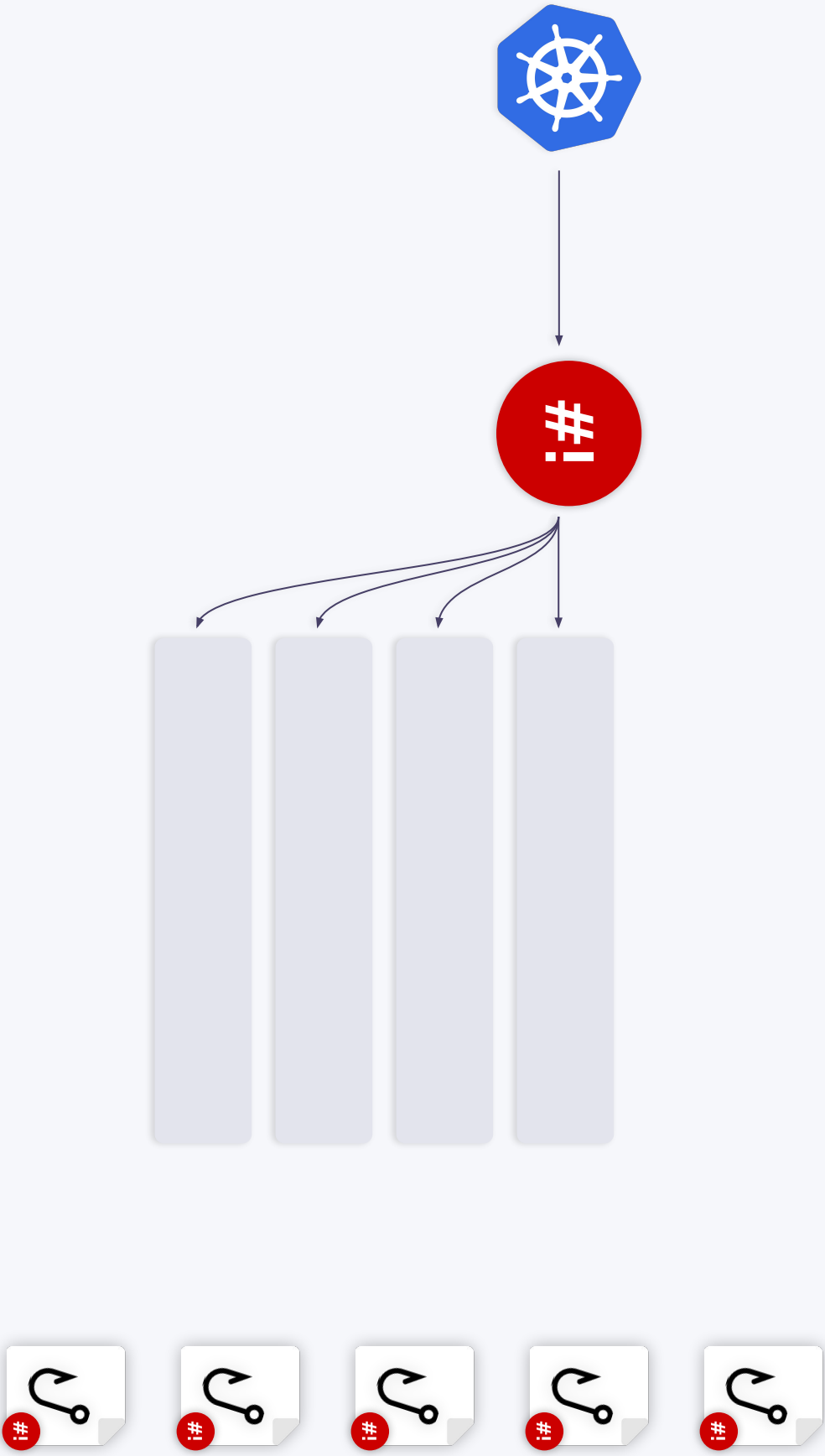
Binding configuration

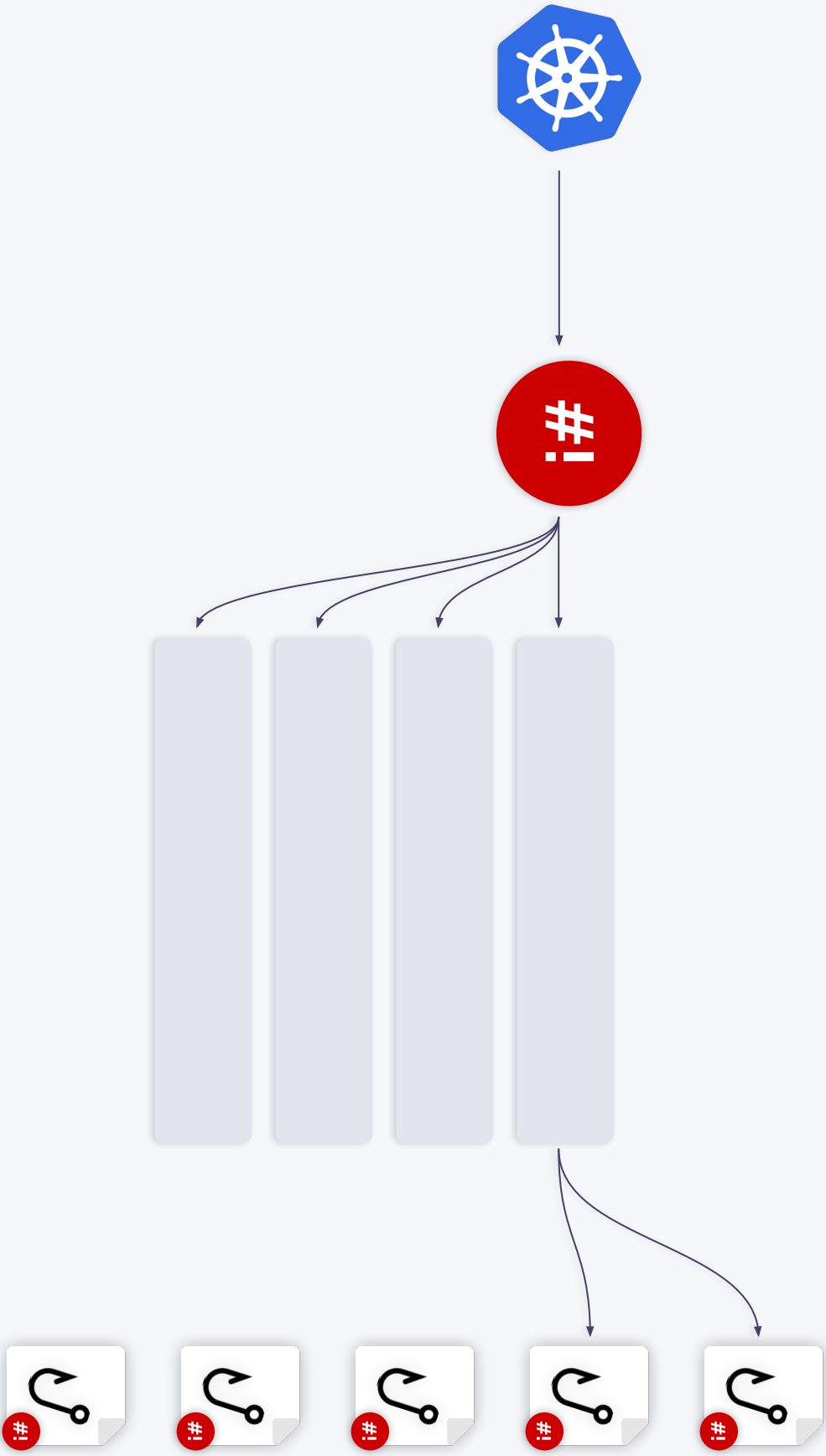
```
function __config__() {
  cat << EOF
    configVersion: v1
    kubernetes:
    - name: src_secret
      apiVersion: v1
      kind: Secret
      nameSelector:
        matchNames:
        - mysecret
      namespace:
        nameSelector:
          matchNames: ["default"]
      group: main
EOF
}
```

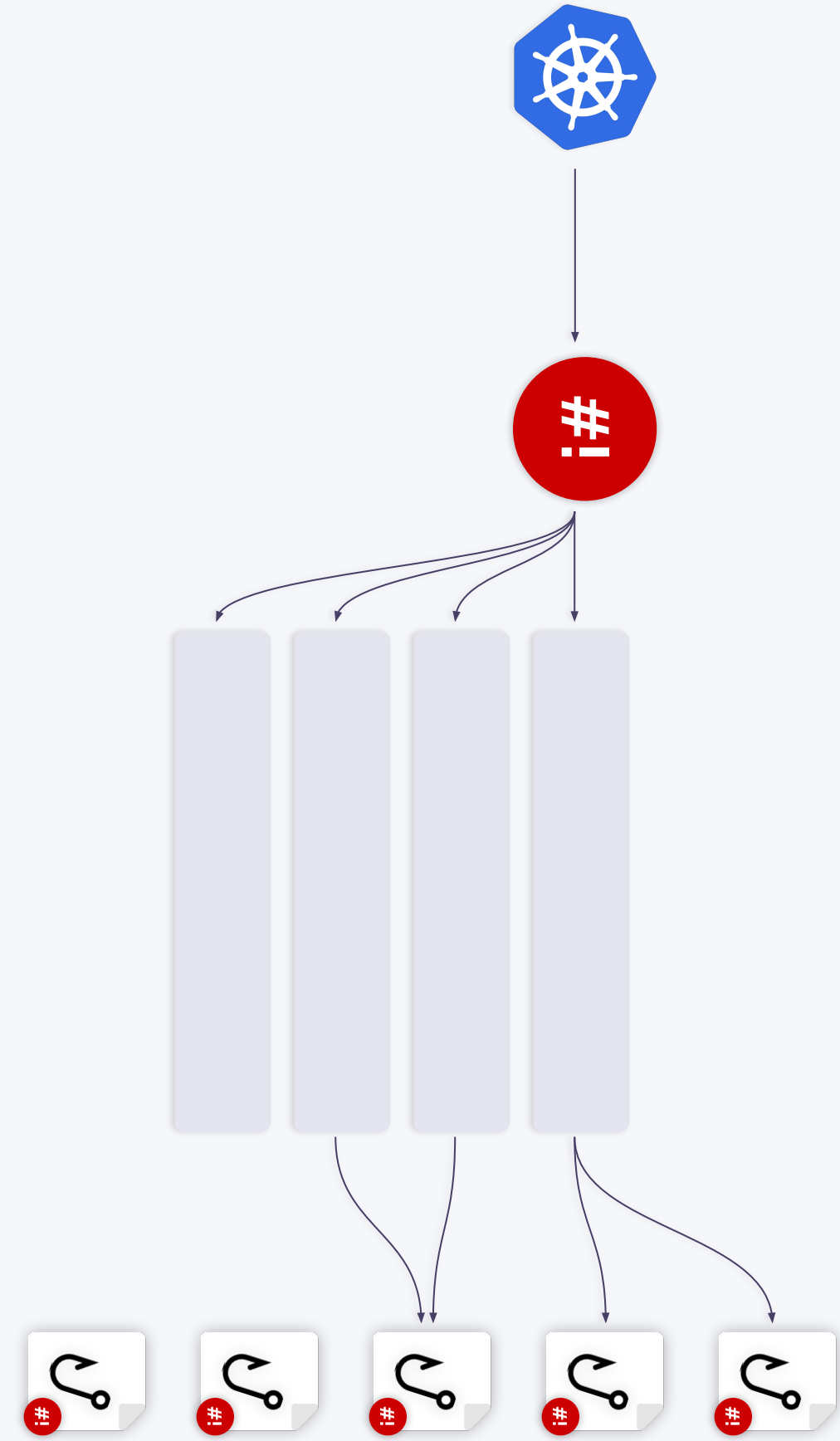
Binding configuration

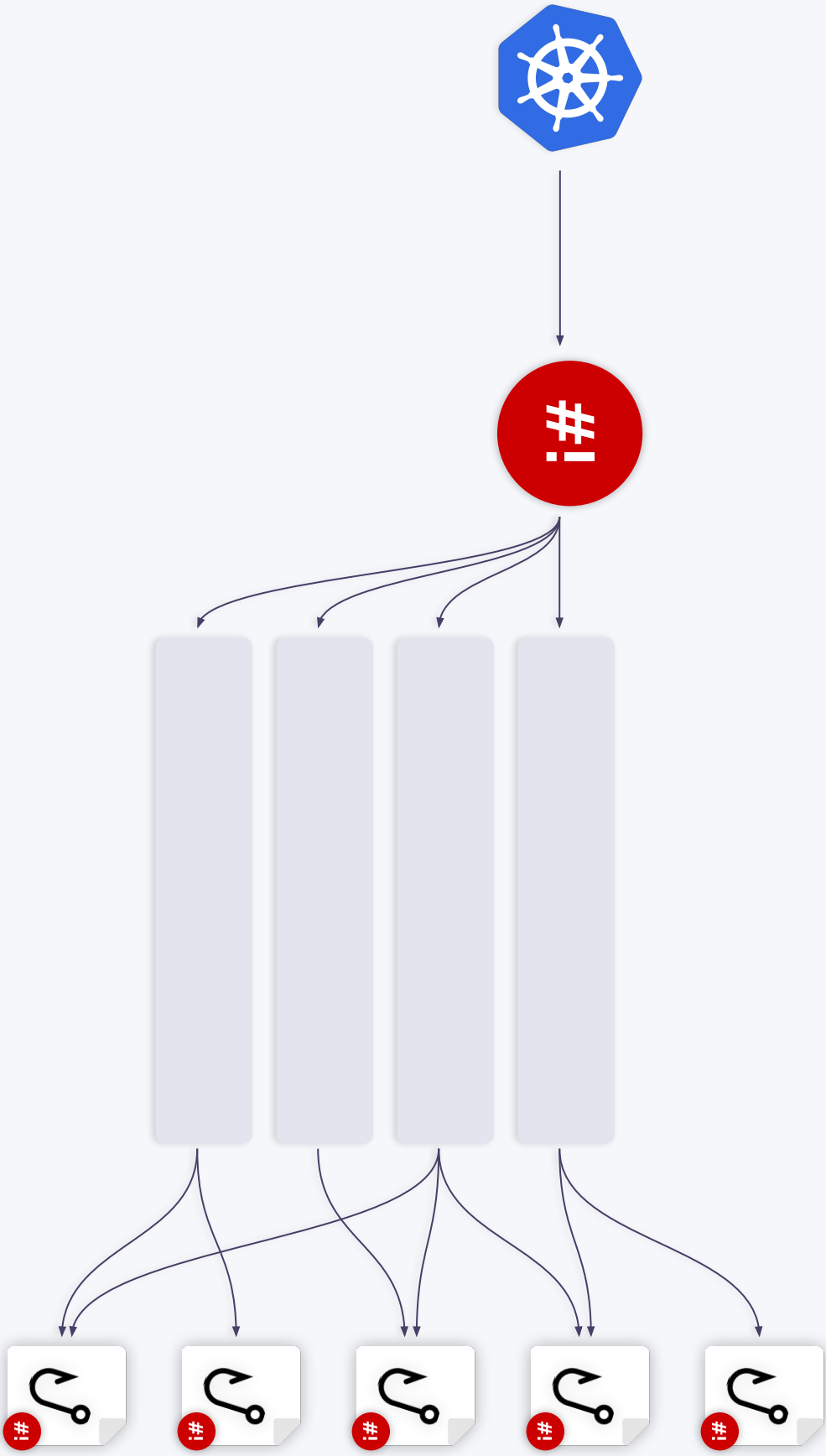
```
function __config__() {
  cat << EOF
    configVersion: v1
    kubernetes:
    - name: src_secret
      apiVersion: v1
      kind: Secret
      nameSelector:
        matchNames:
        - mysecret
      namespace:
        nameSelector:
          matchNames: ["default"]
      group: main
EOF
}
```











Binding configuration

```
function __config__() {
  cat << EOF
    configVersion: v1
    kubernetes:
    - name: src_secret
      apiVersion: v1
      kind: Secret
      nameSelector:
        matchNames:
        - mysecret
      namespace:
        nameSelector:
          matchNames: ["default"]
      group: main
EOF
}
```

Binding configuration

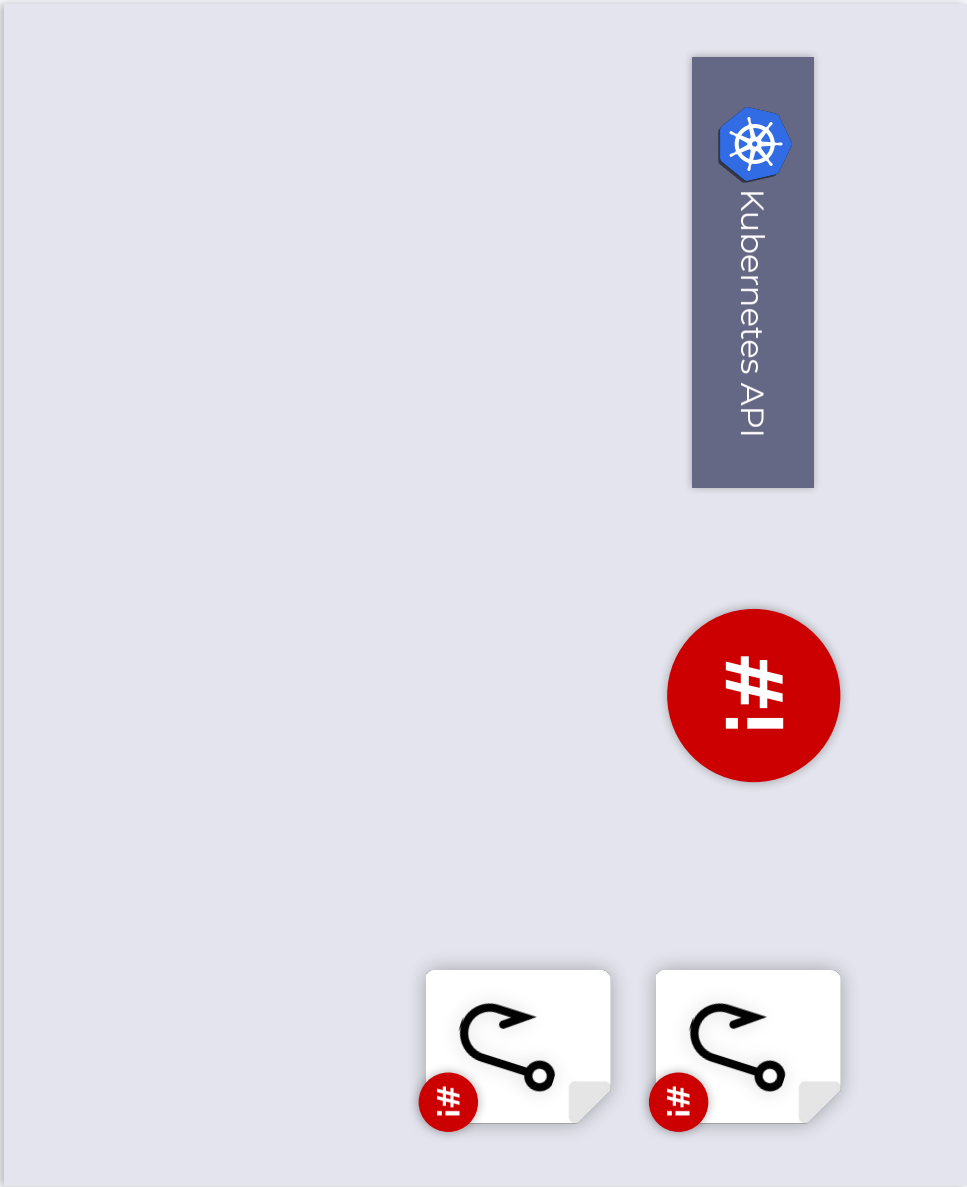
```
function __config__() {
  cat << EOF
    configVersion: v1
    kubernetes:
    - name: src_secret
      apiVersion: v1
      kind: Secret
      nameSelector:
        matchNames:
        - mysecret
      namespace:
        nameSelector:
          matchNames: ["default"]
      group: main
      queue: "some_name"
EOF
}
```

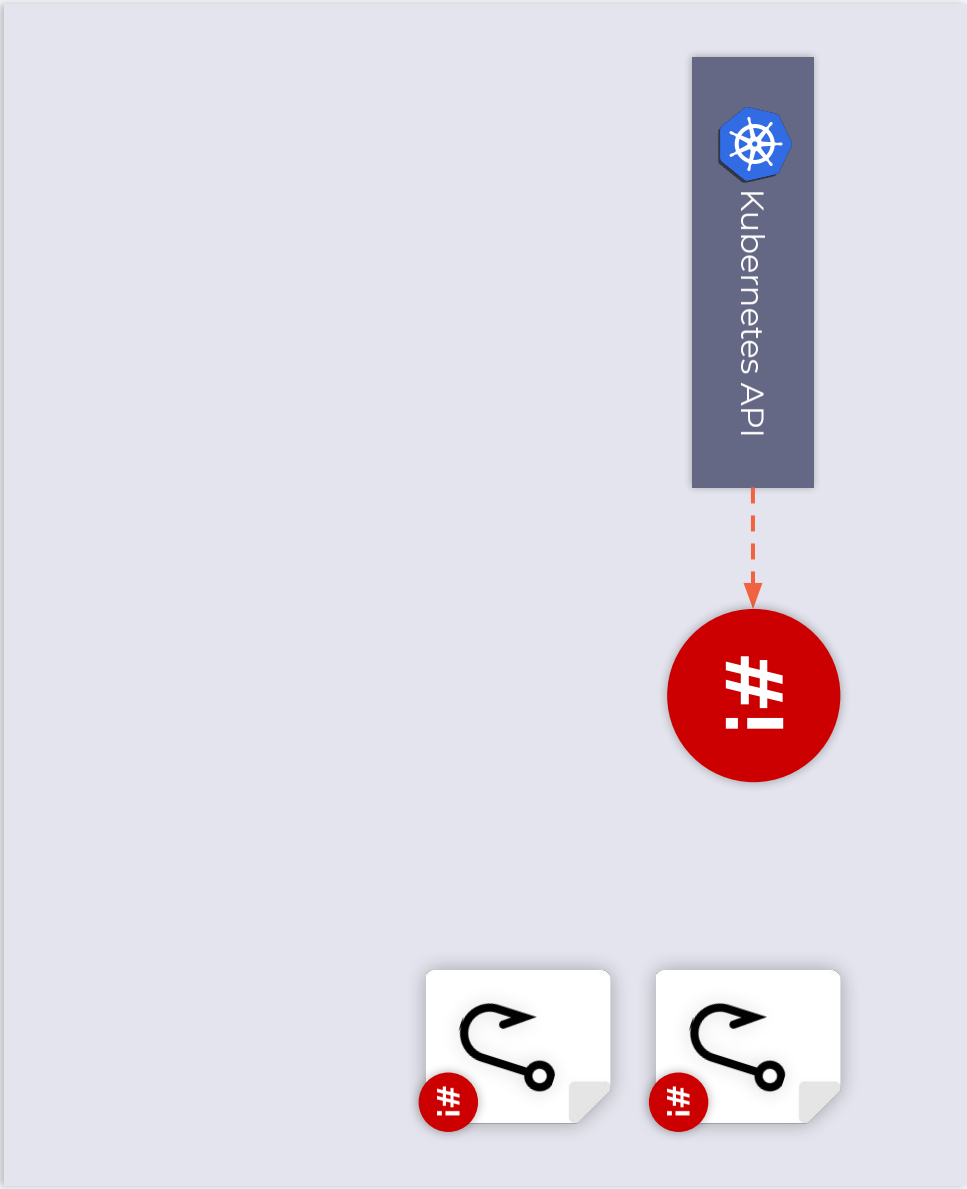
shell-operator

shell-operator

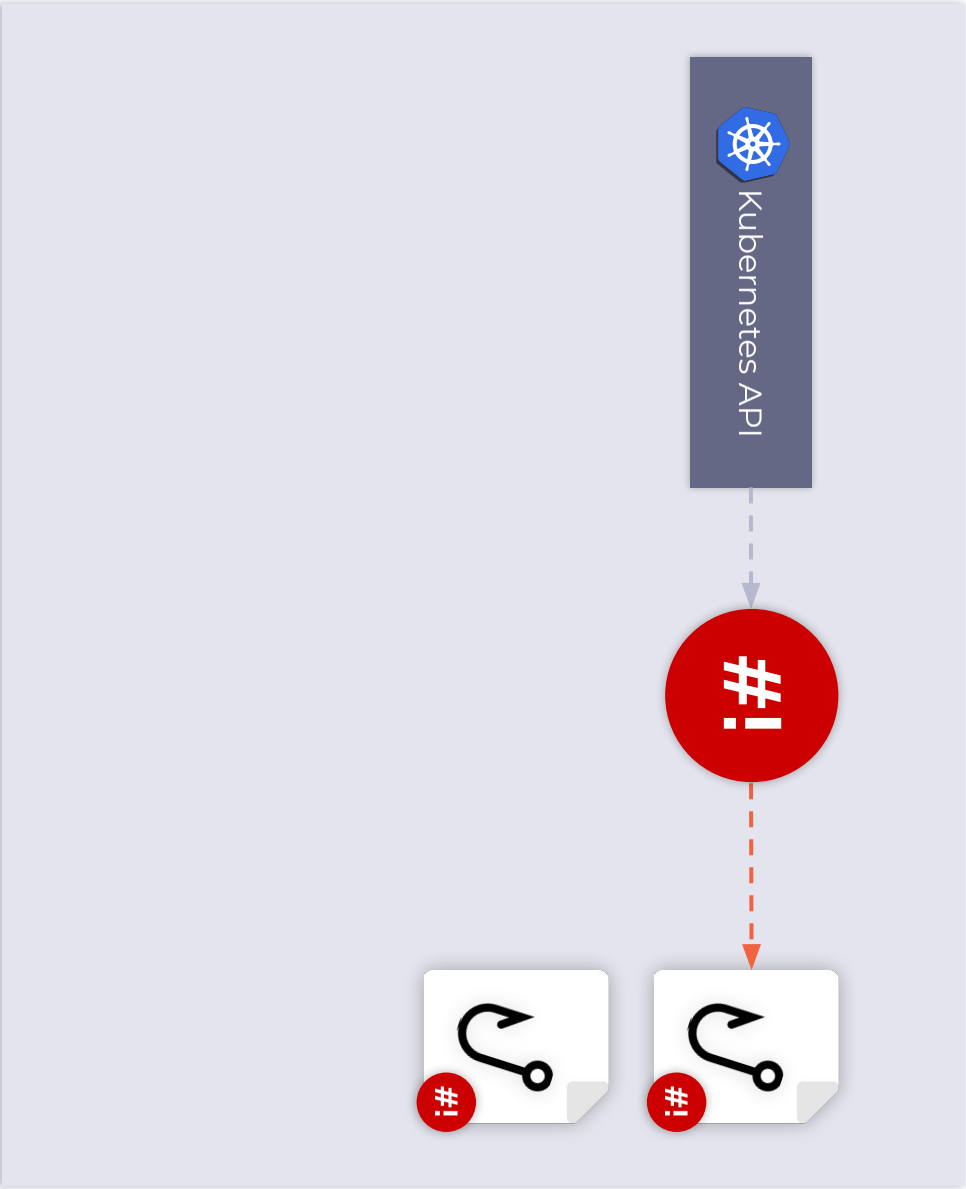
addon-operator

kubernetes

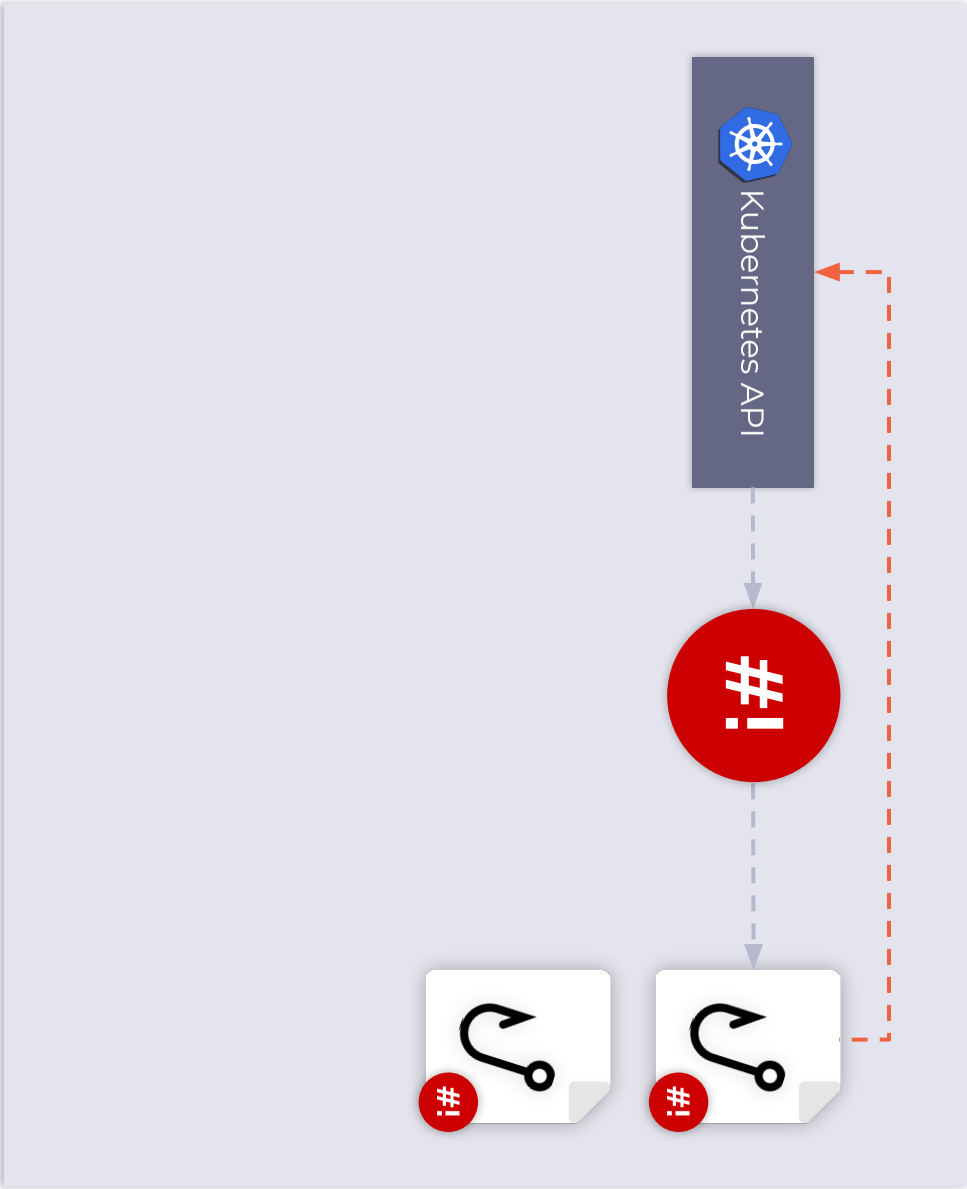


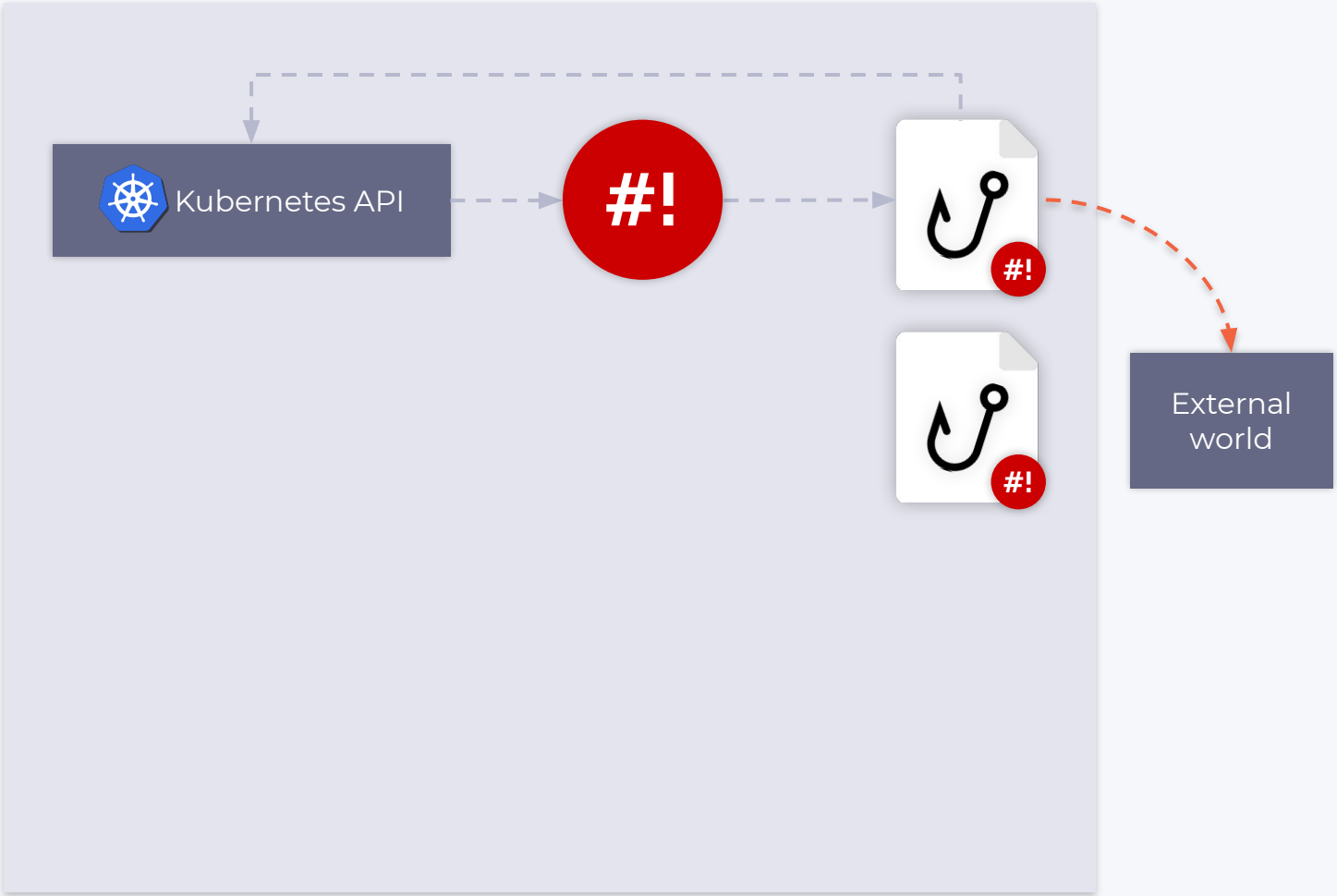

Kubernetes API
#!
#!
#!

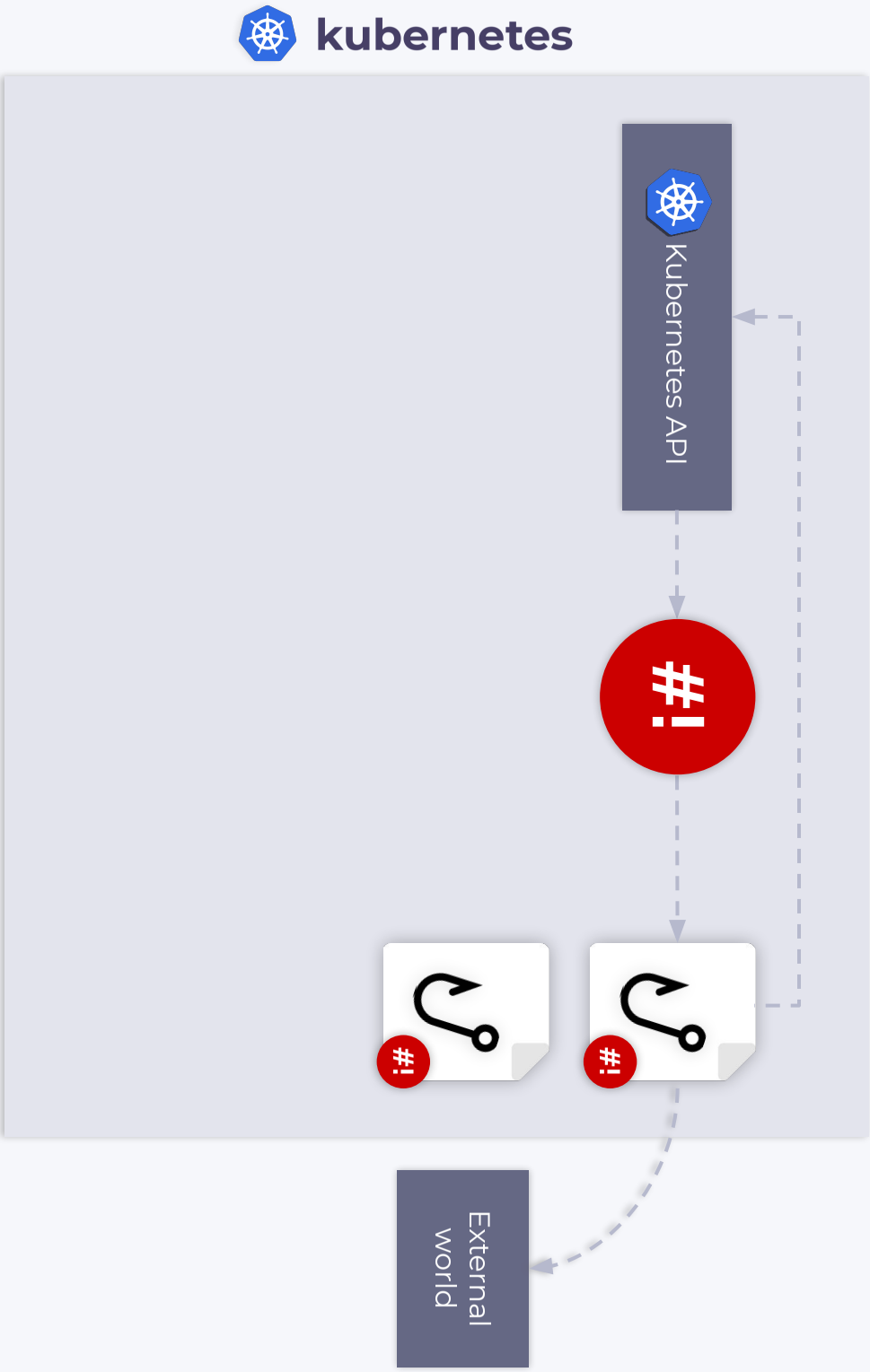
kubernetes

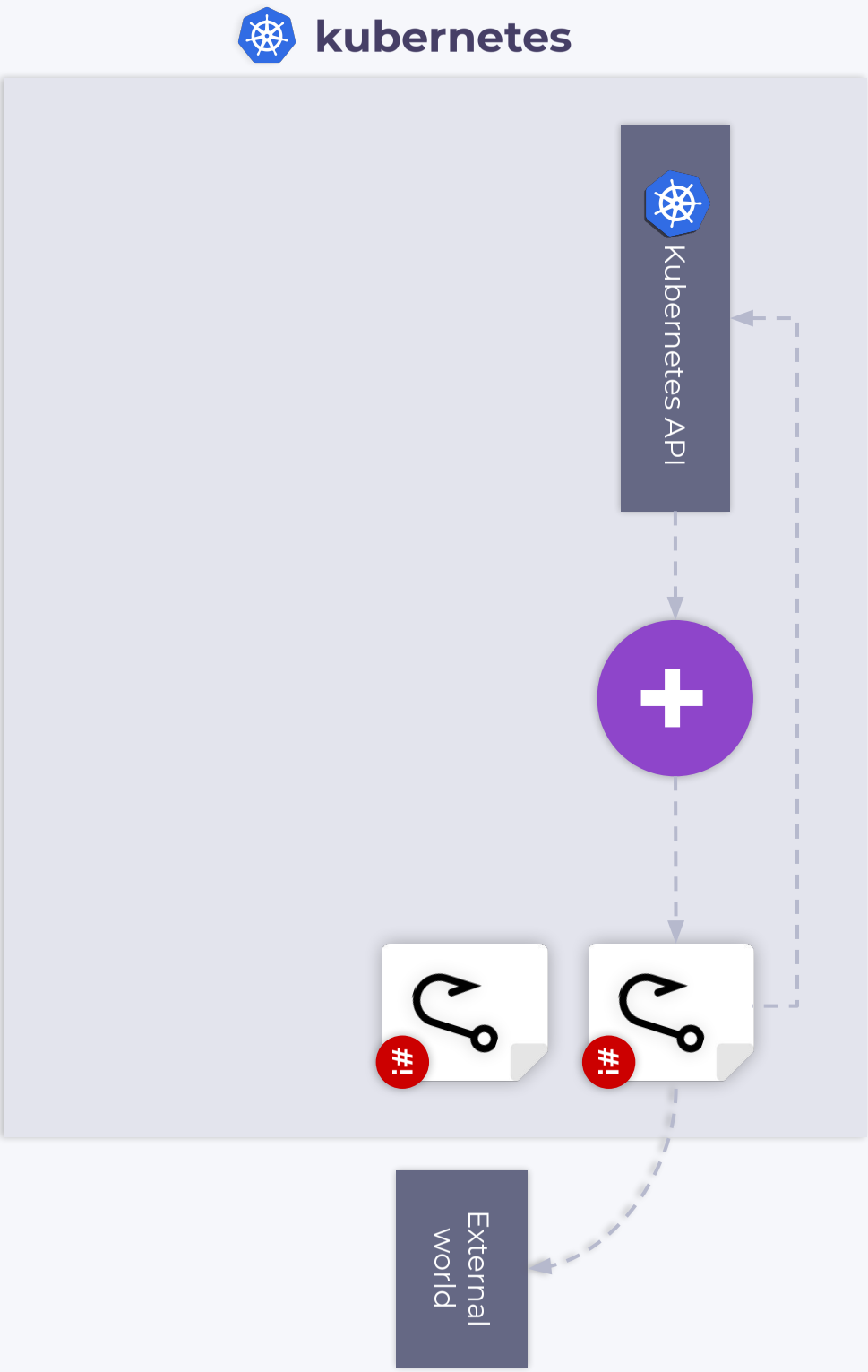


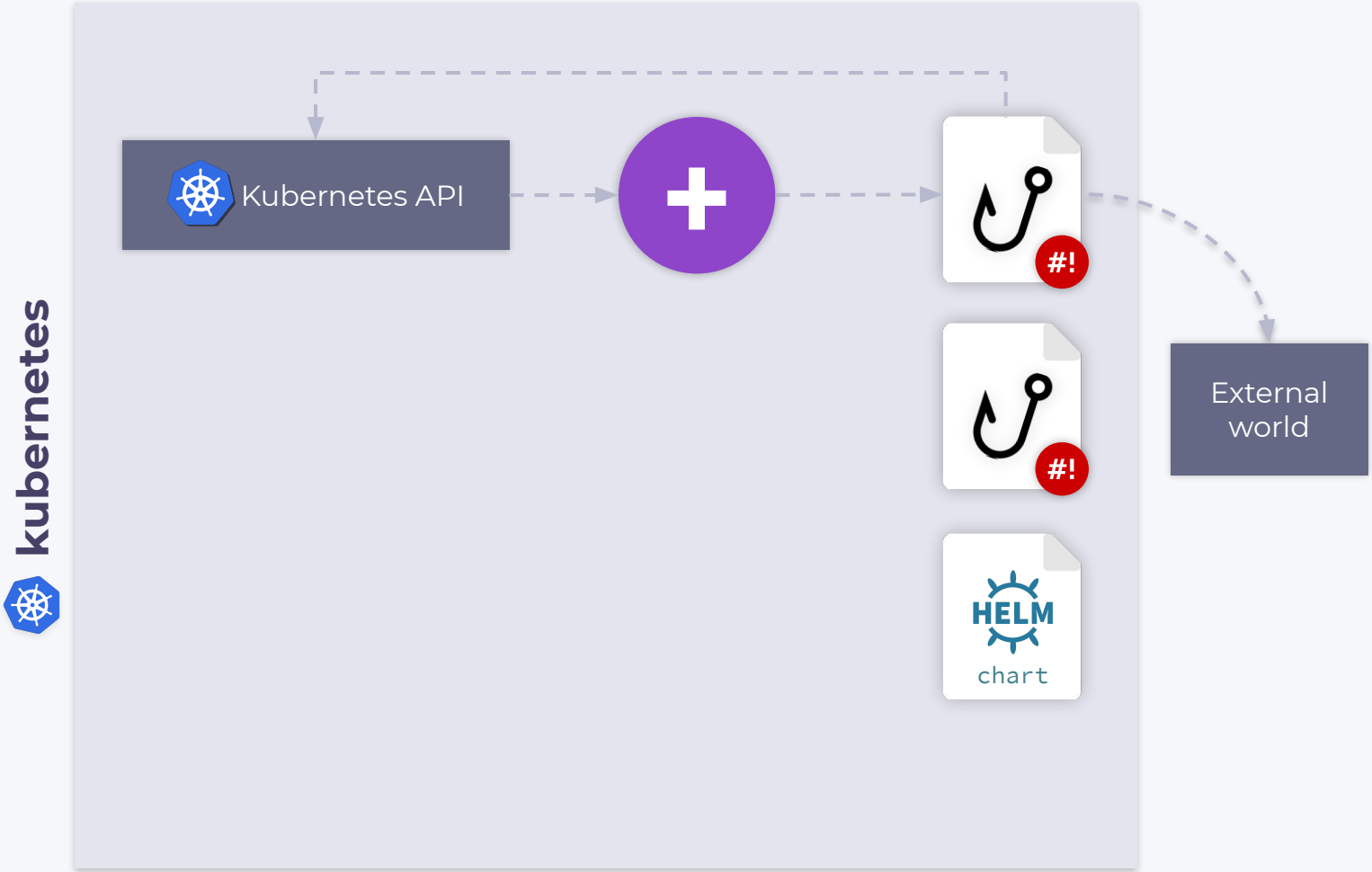
kubernetes

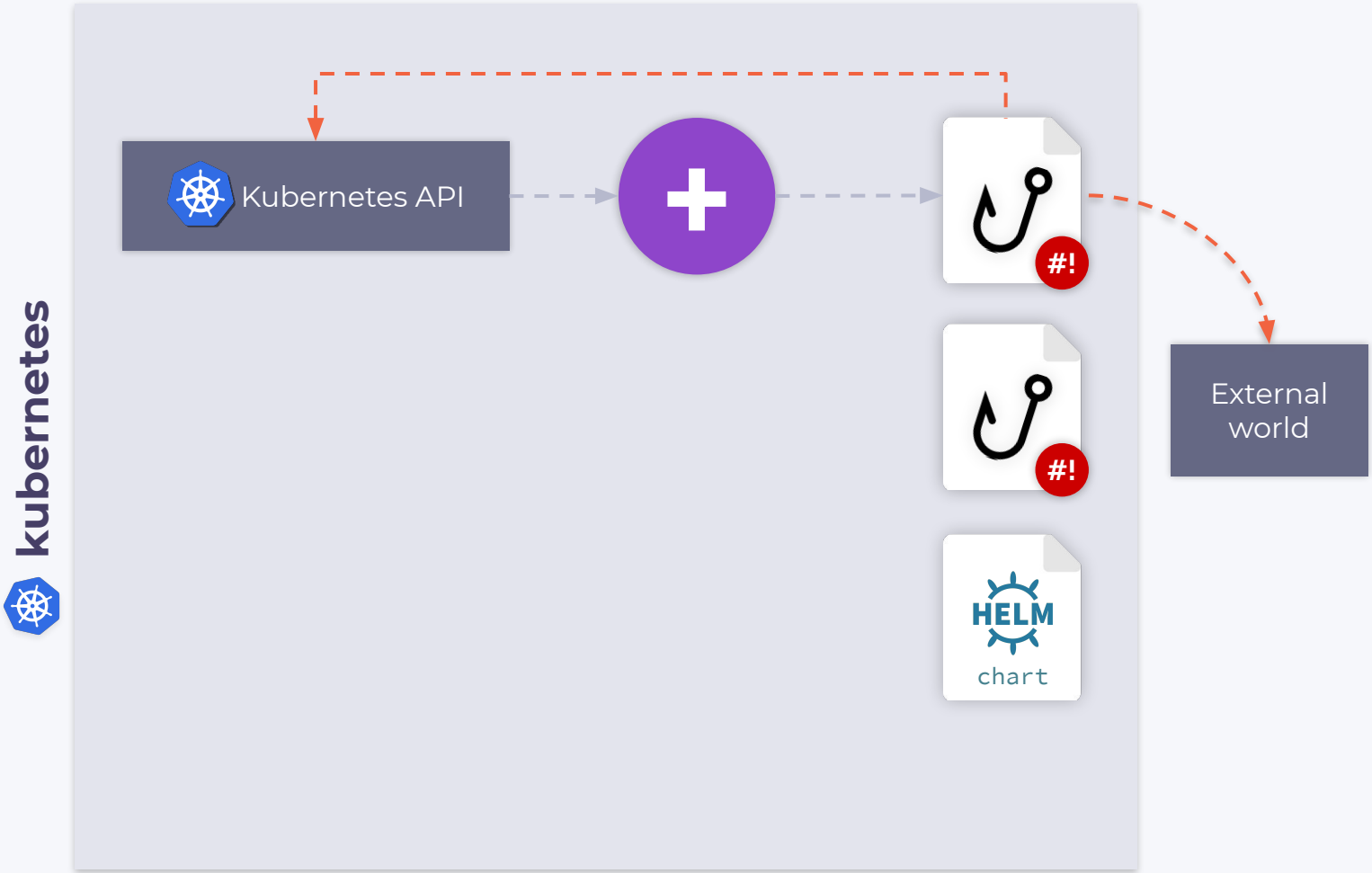


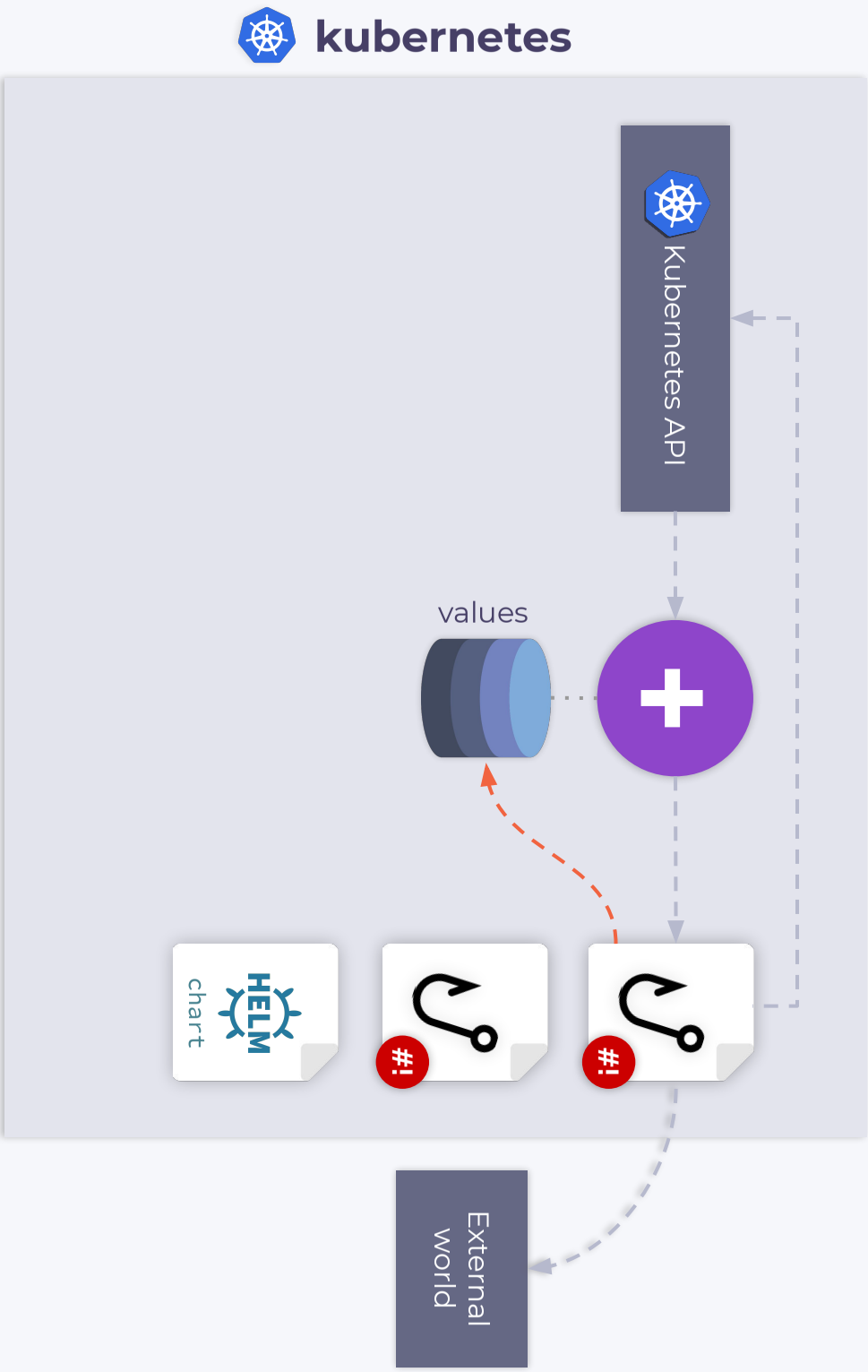

kubernetes
Kubernetes API
#!
#!
#!
External
world

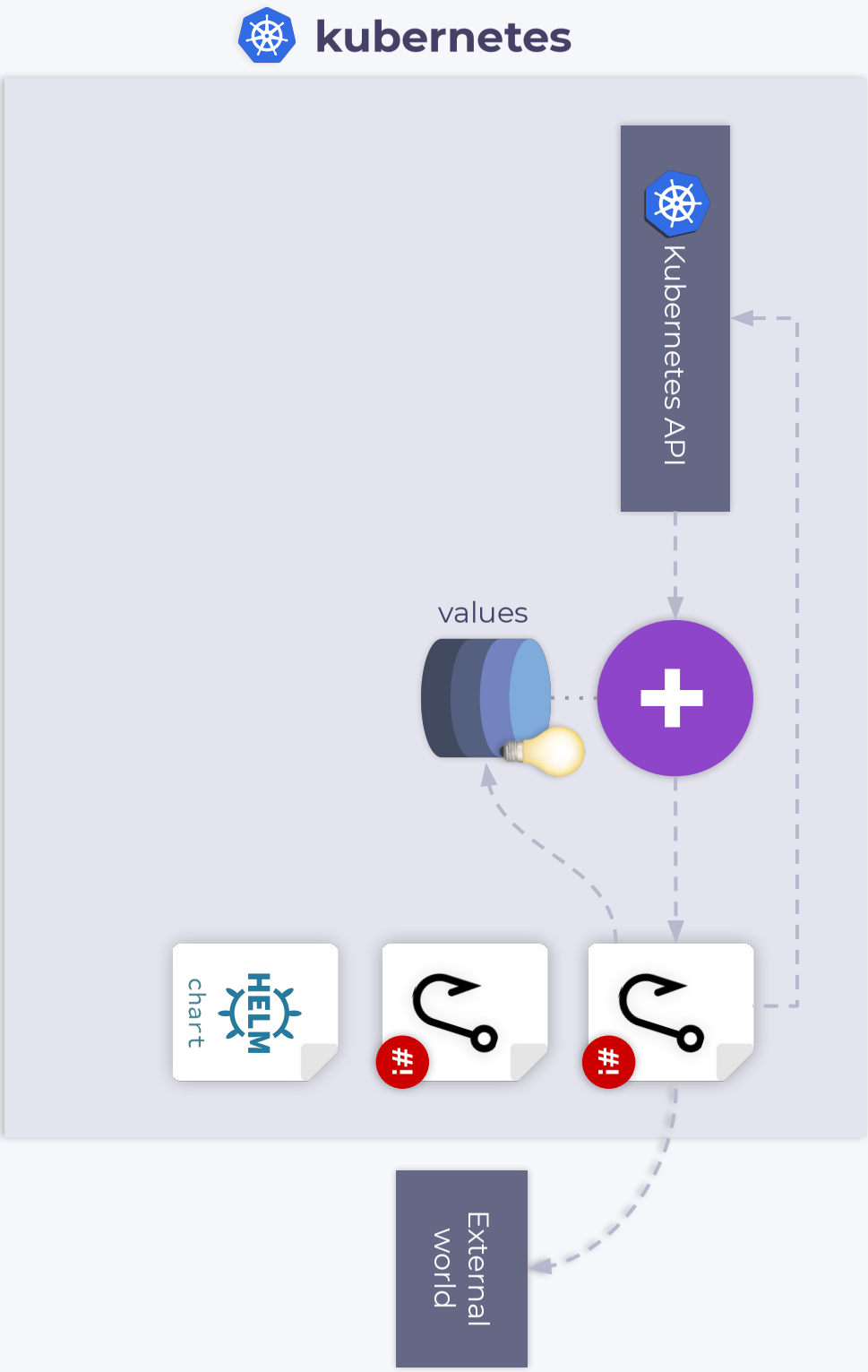

kubernetes
Kubernetes API
#!
External world

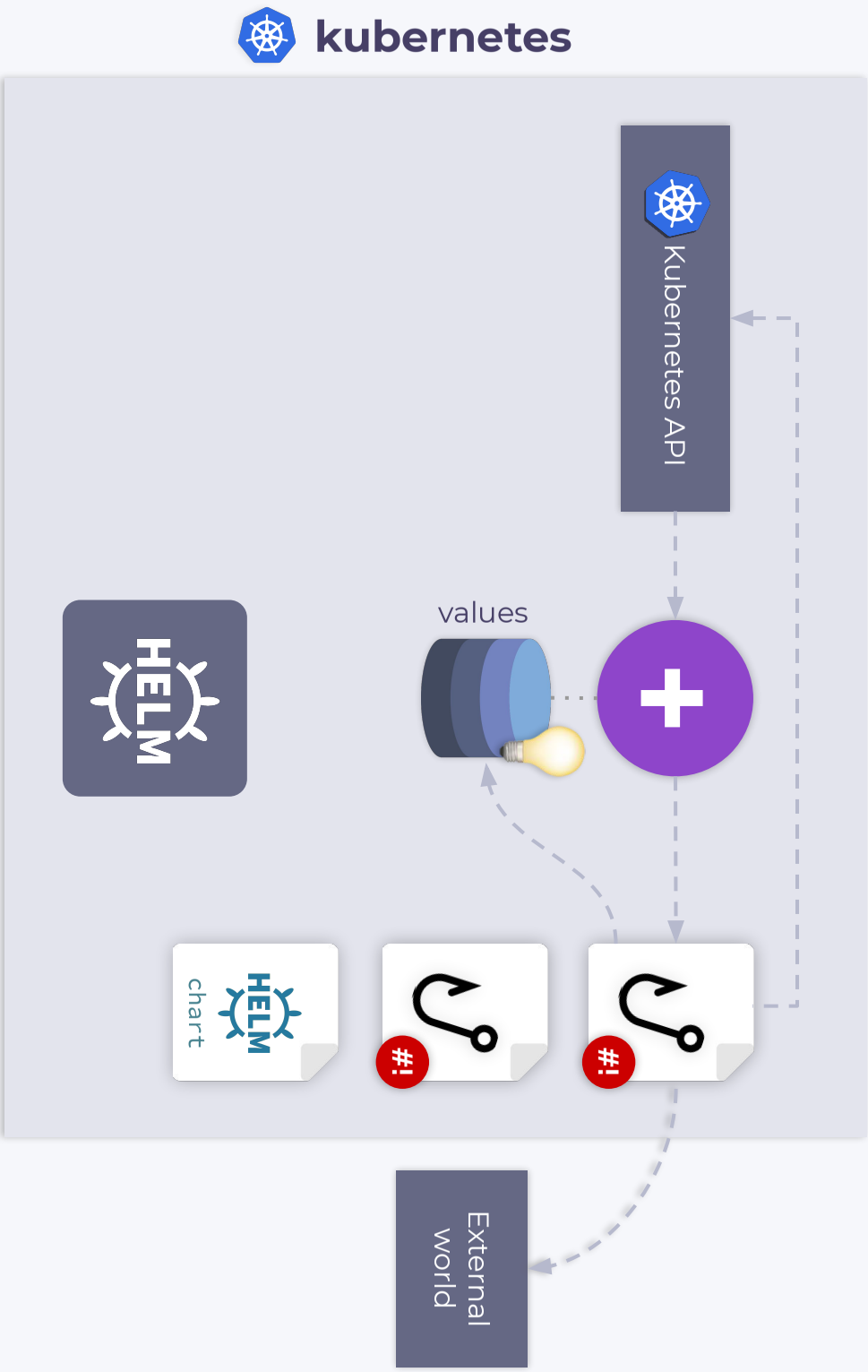

kubernetes
Kubernetes API
External world

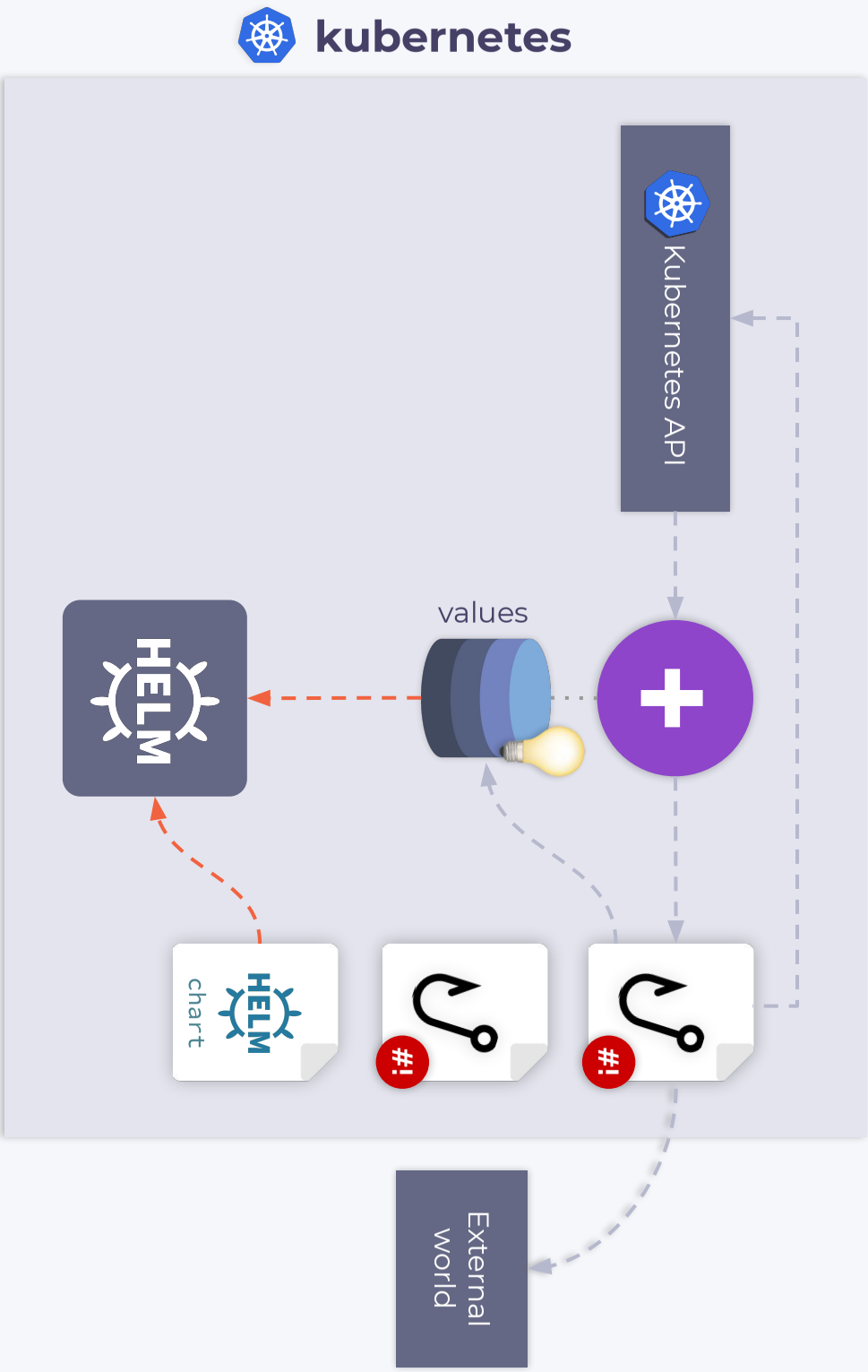

kubernetes
Kubernetes API
#!
#!
HELM
chart
External
world

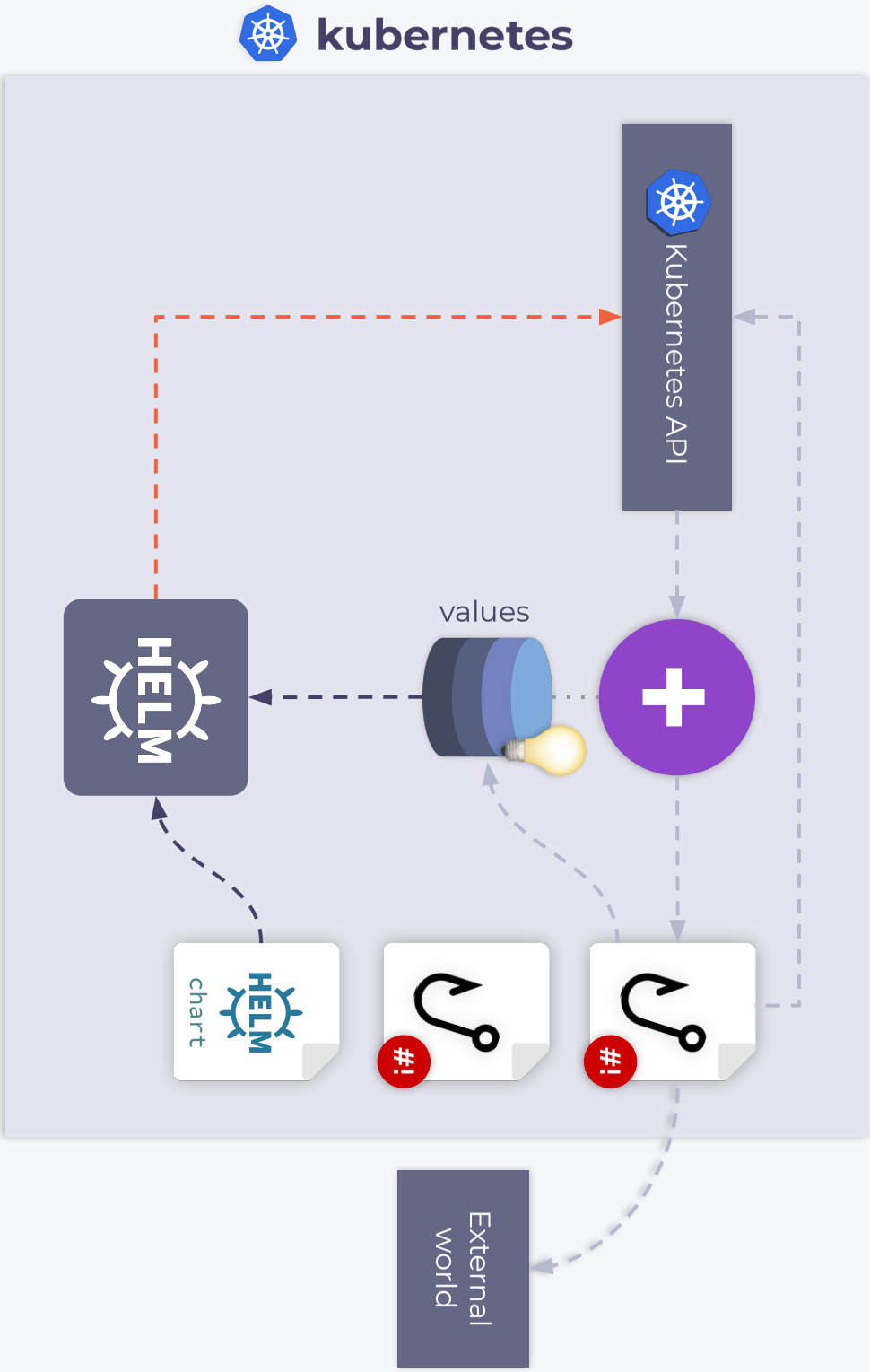

kubernetes
Kubernetes API
#!
#!
HELM
chart
External
world


kubernetes
Kubernetes API
values
HELM
chart
External
world


kubernetes
Kubernetes API
values
chart
External world


kubernetes
Kubernetes API
values
HELM
chart
External world


kubernetes
Kubernetes API
values
HELM
HELM
chart
External
world


kubernetes
Kubernetes API
values
HELM
HELM
chart
External
world

shell-operator hooks events

addon-operator hooks events helm

Thank you!

shell-operator

github.com/flant/shell-operator

addon-operator

github.com/flant/addon-operator

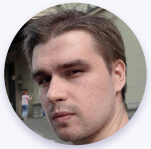
Like? Try!

And don't forget to star on GitHub

Dmitry Stolyarov

CTO & Co-founder

linkedin.com/in/distol twitter.com/dmistol



Andrey Klimentyev

Solutions Engineer

Flant

Running your production 24×7×365

flant.com medium.com/flant-com

youtube.com/c/flant_com