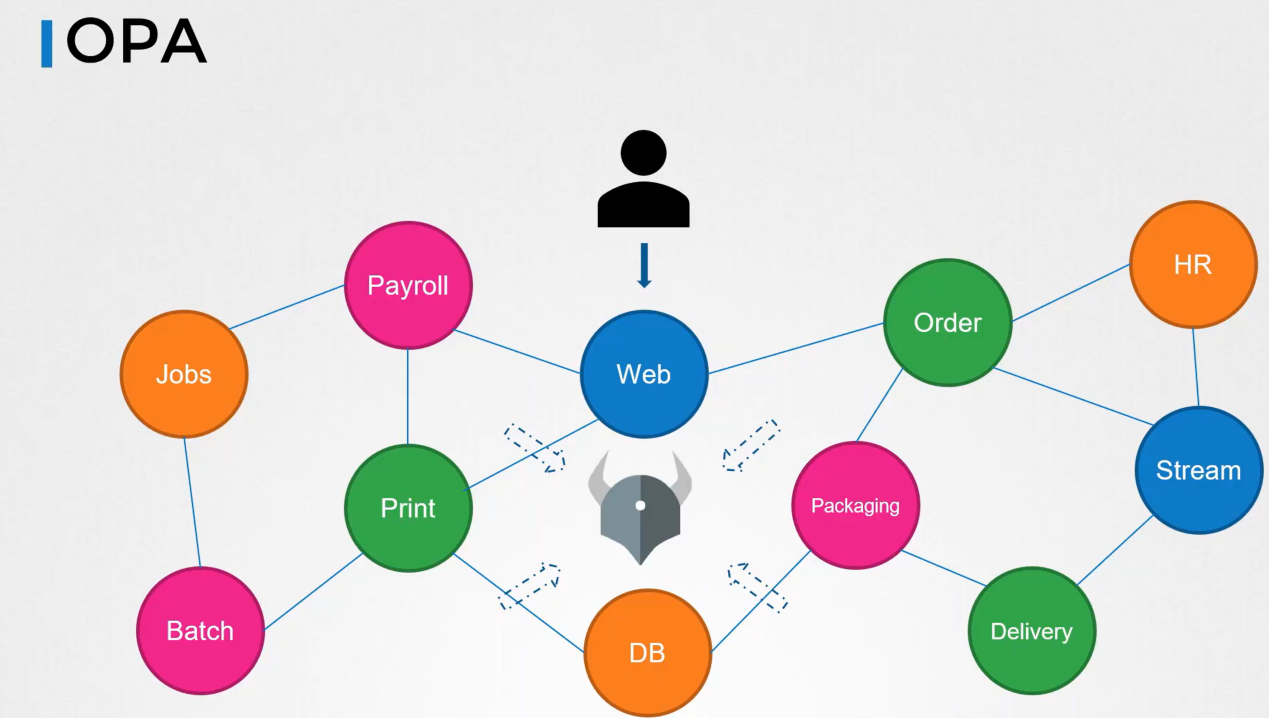
# Open Policy Agent



All services reach out to OPA for authorizing that request. OPA verifies the request and returns an allowed or denied message which the service then process the request.

### Install OPA



### Load Policy

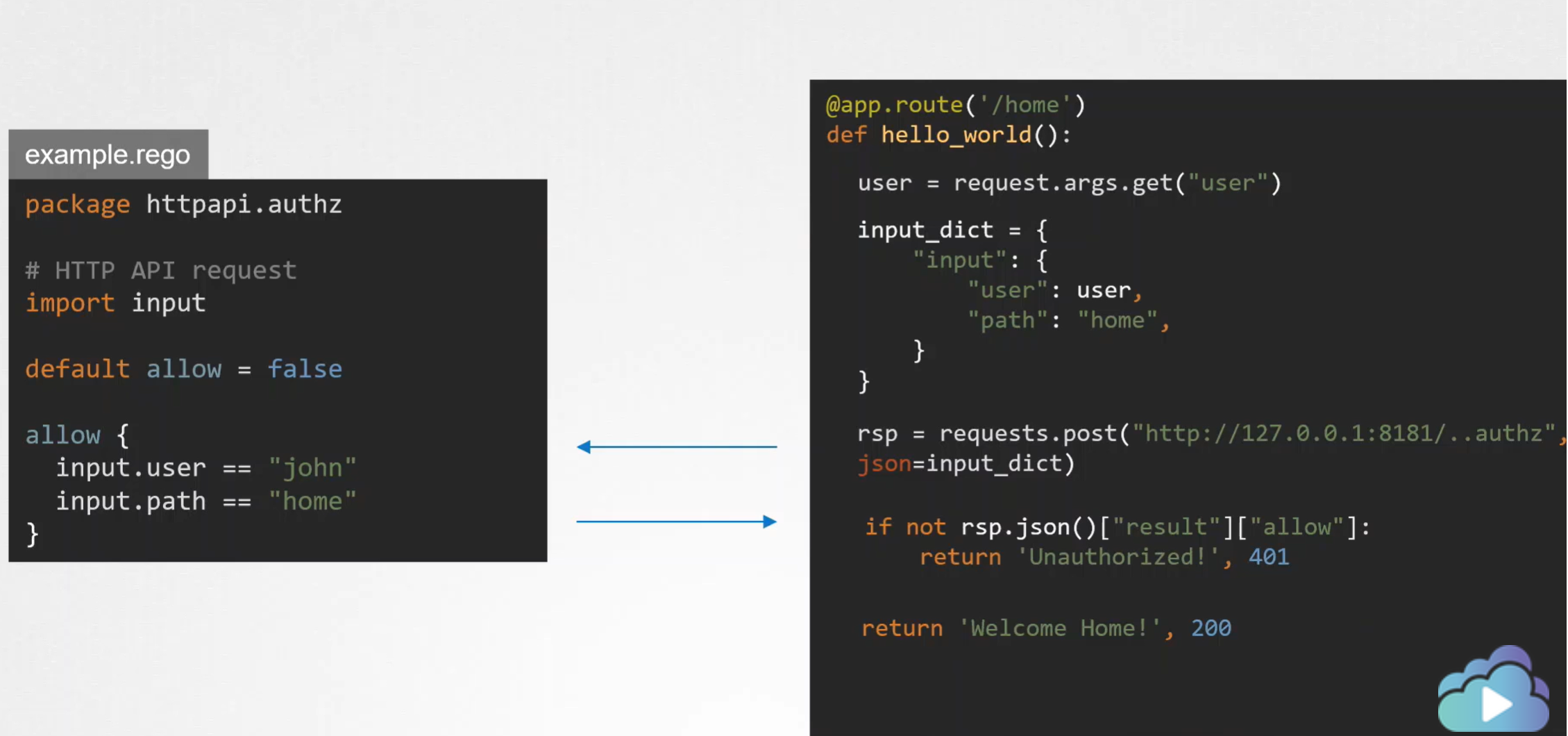


Using rego language

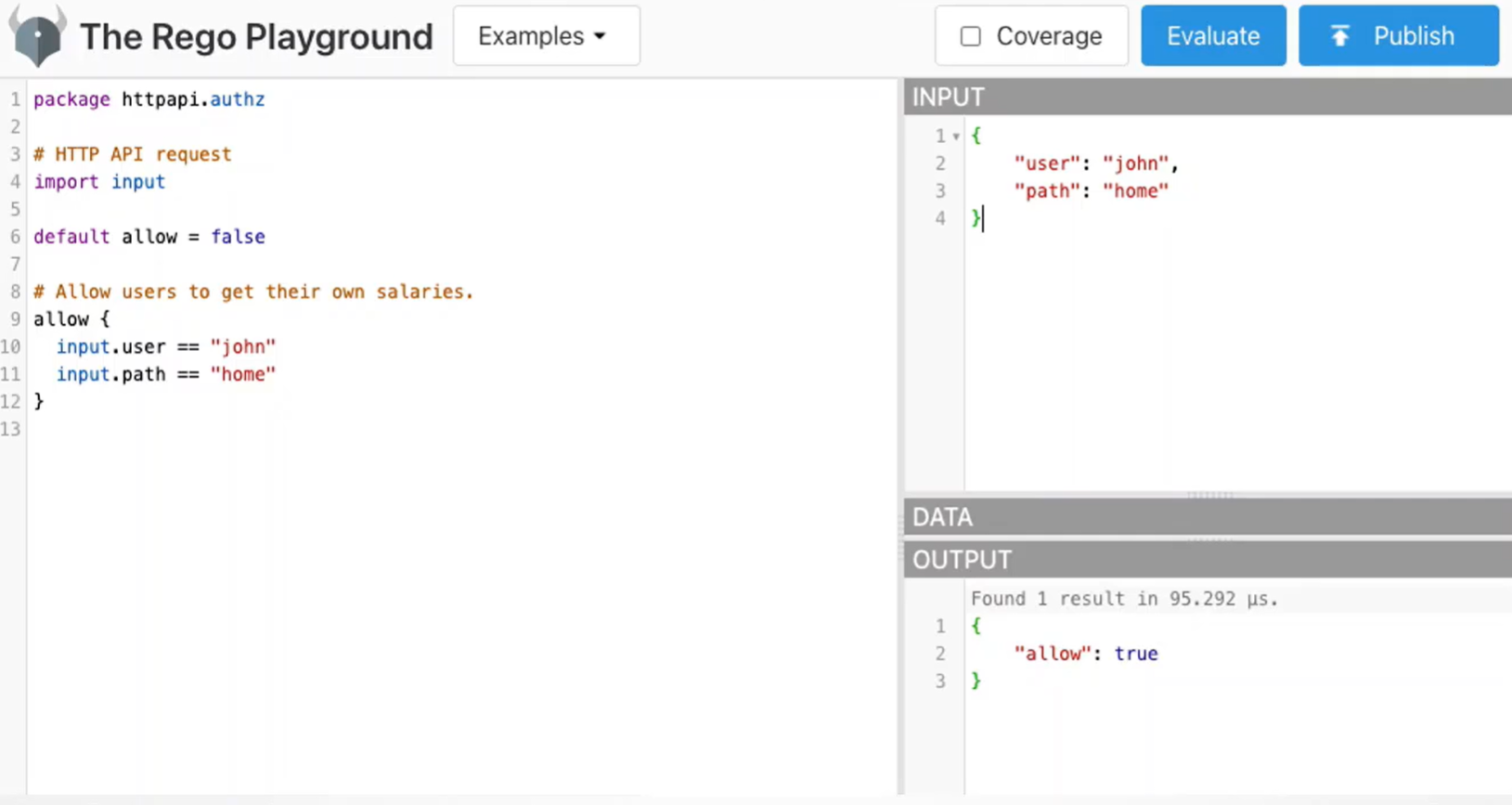
Application without OPA



Application with OPA



### Rego Playground



### Policy Testing



Below are some references:

How Netflix Is Solving Authorization Across Their Cloud [I] – Manish Mehta & Torin Sandall, Netflix

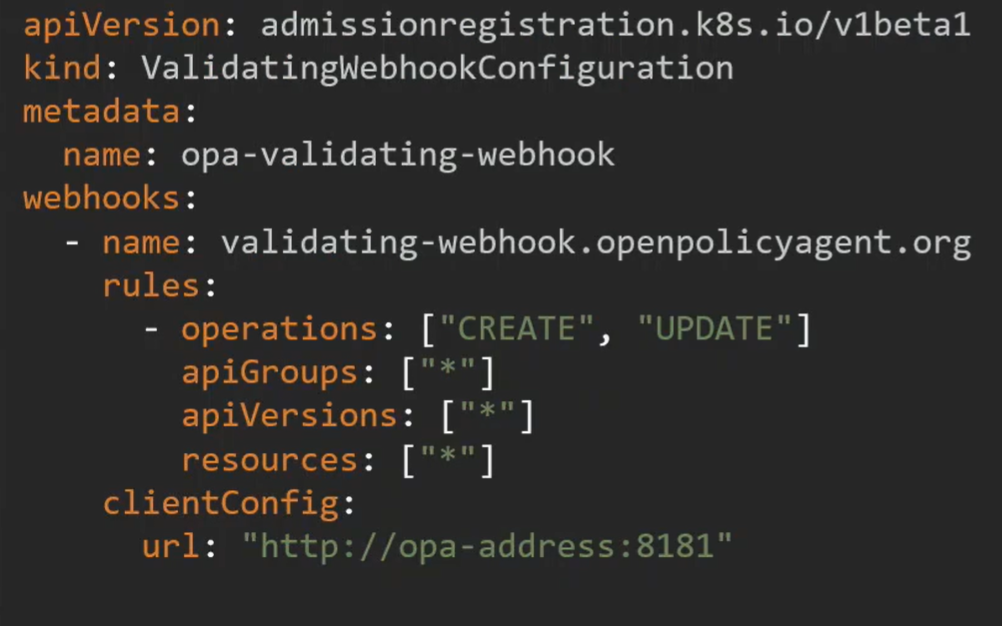
https://www.youtube.com/watch?v=R6tUNpRpdnY

OPA Deep Dive

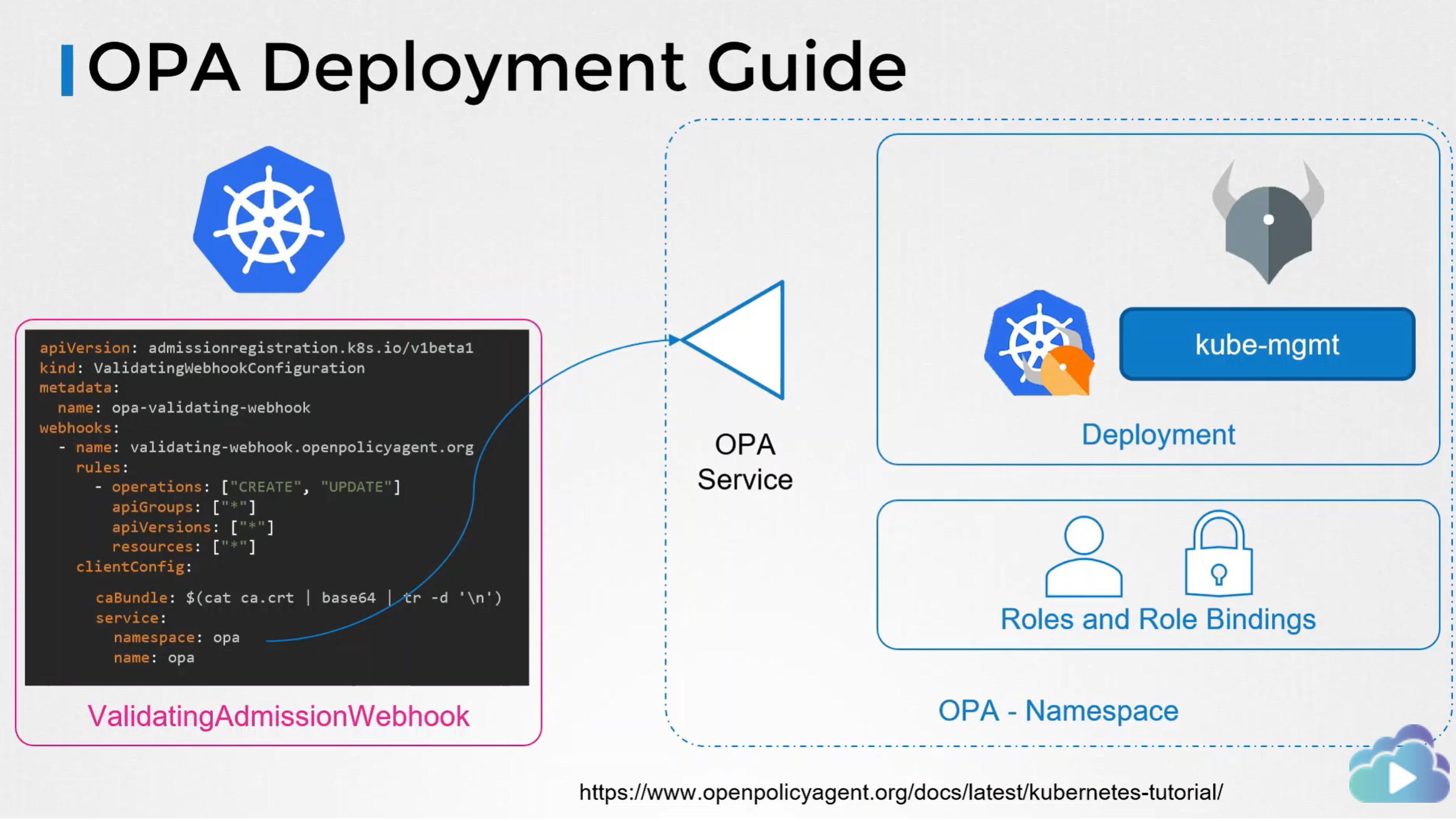
<https://www.youtube.com/watch?v=4mBJSIhs2xQ>

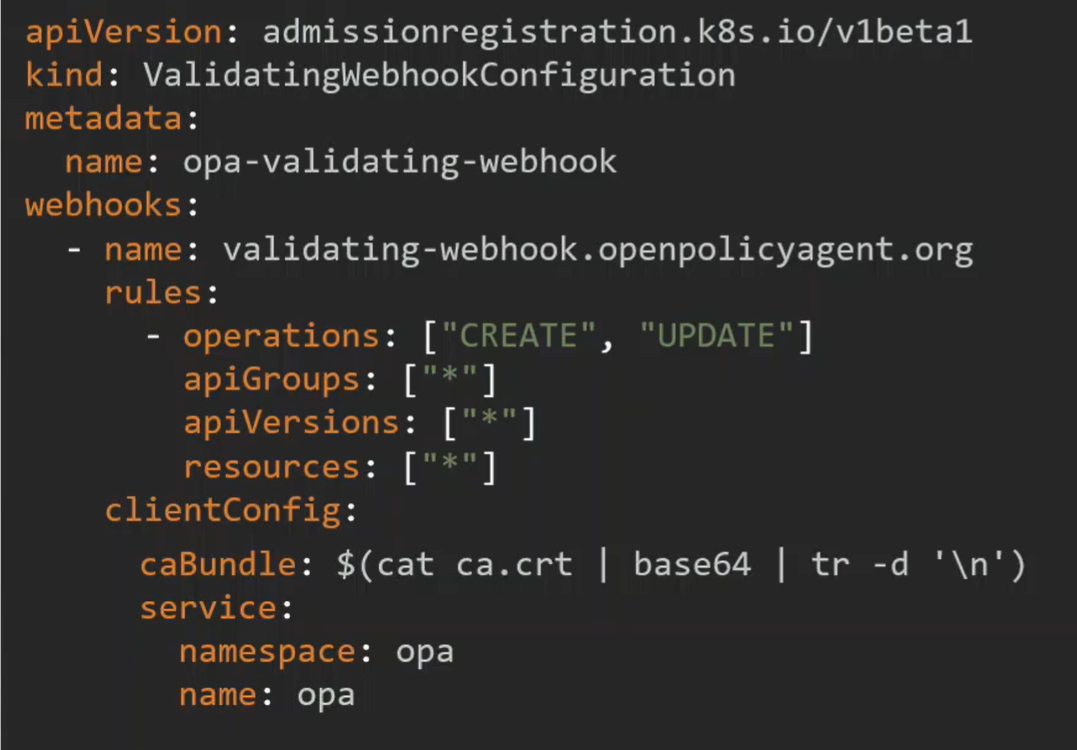
### Connecting mutating & validating admission to OPA

OPA deploy out of the k8s cluster



OPA deploy in the k8s cluster

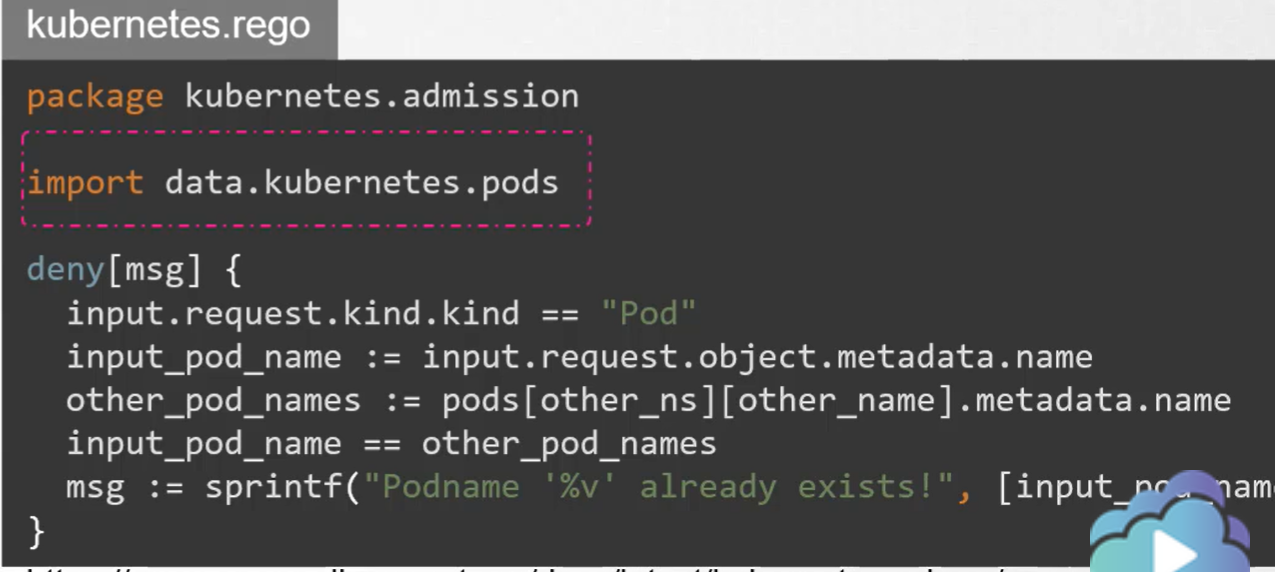




OPA Policy



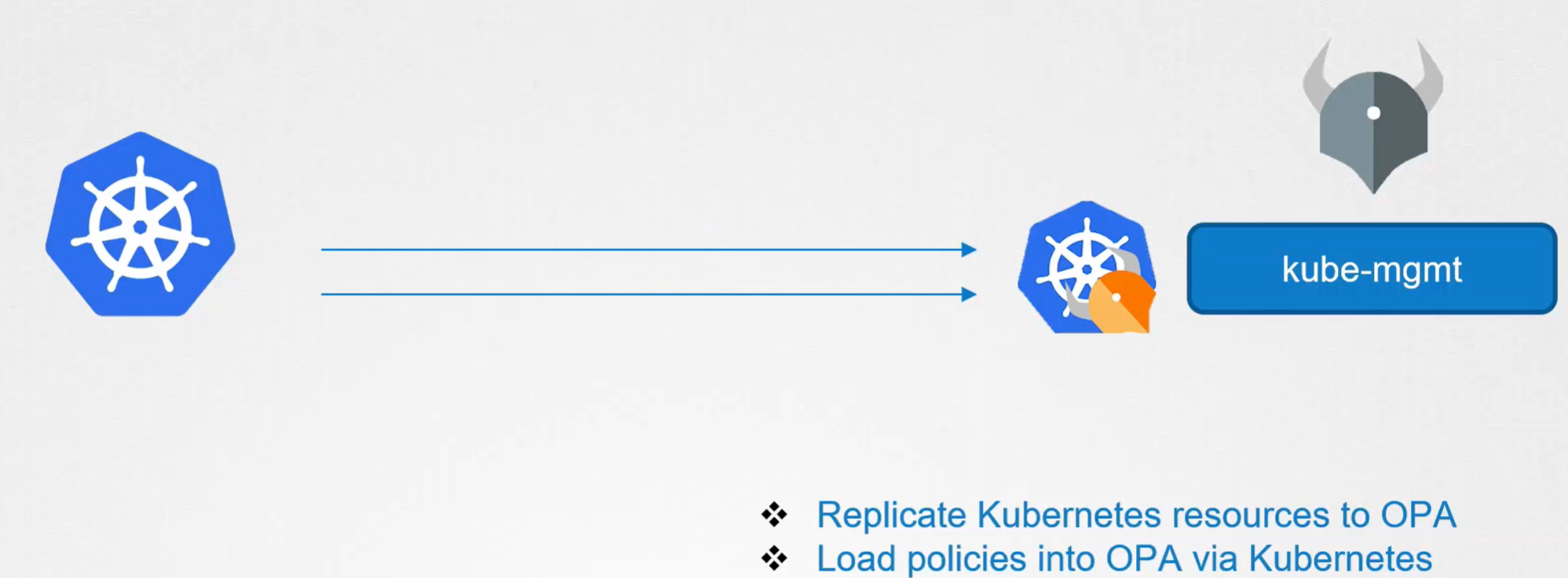
If some needed information not sent to OPA, like all the pods name in the cluster. OPA can get them anyway by using some modules.



But how does OPA know about the resource in k8s?

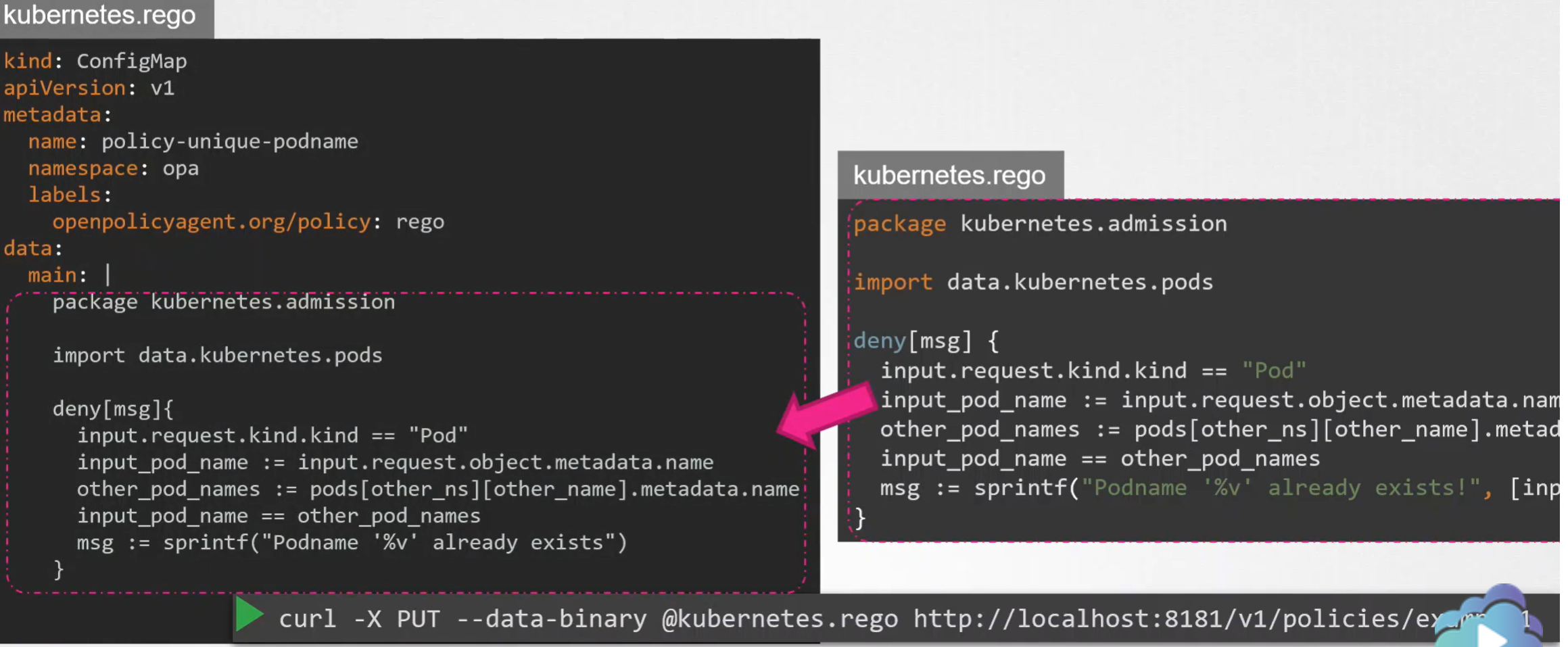
We can make that happen using the kube-mgmt service

Kube-mgmt is a service deployed as a side-car container alone with OPA and is used to replicate the resource definitions from kubernetes, so they can be cached at OPA.



This information can be imported and used to refer to objects in kubernetes while deploying policies.

Load policies for OPA in K8S using configmaps instead of curl -X PUT .......



Kube-mgmt service will automatically identify these policies configmaps and load them into OPA as policies.