We can create a single policy and add multiple objects to that policy else u can create multiple policy and attach single object to those policy.

If you create multiple policy u can find a brand violation details.

Compliance type : must have mustnothave , must only have

U just create a cluster and install the controller, then create configuration policy, then it will work and and report the vilotaions to use it with part governance policy framework.

You just embed entire configuration policy into the policy wrapper.

Then frame work will use this policy to distrubte the configuration policy to a manged cluster and result back

<https://github.com/open-cluster-management/governance-policy-framework>

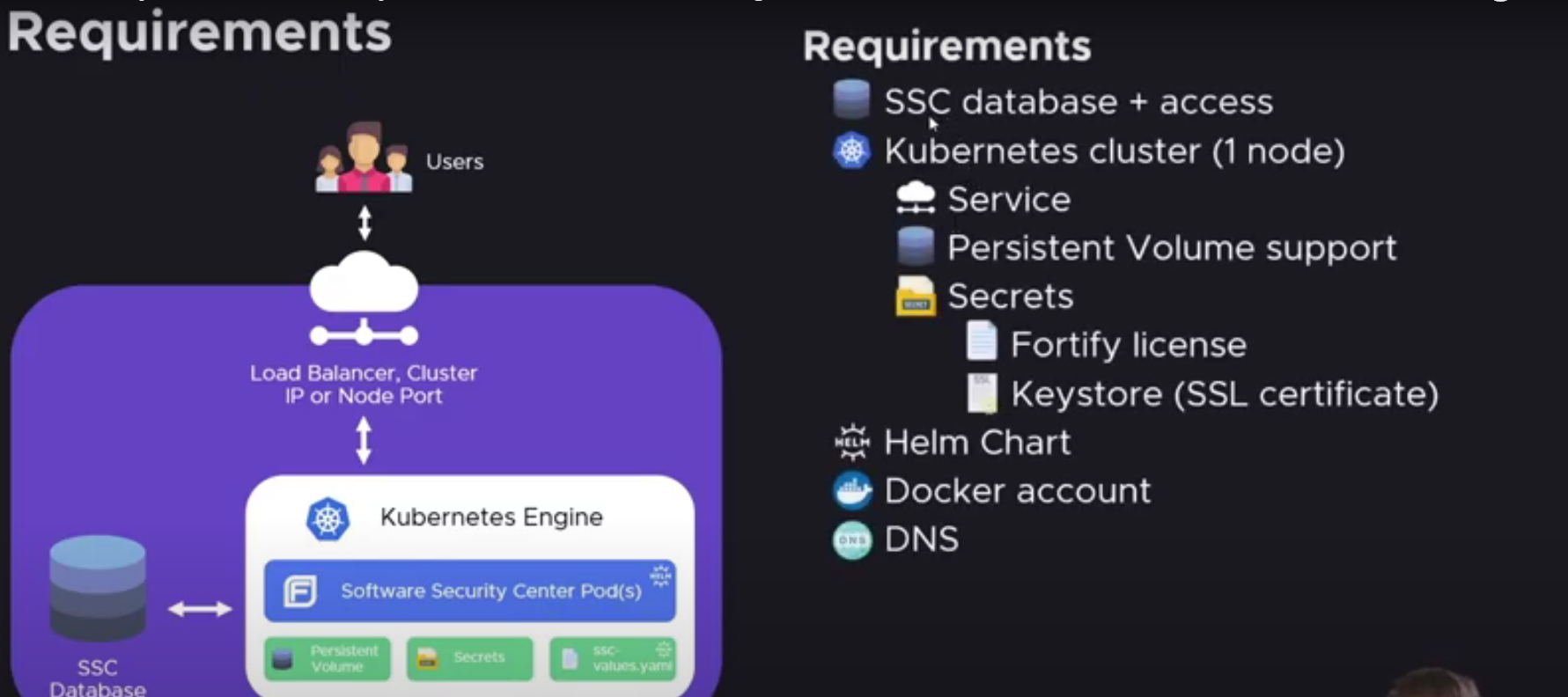
<https://github.com/open-cluster-management/governance-policy-framework/tree/main/doc/configuration-policy>

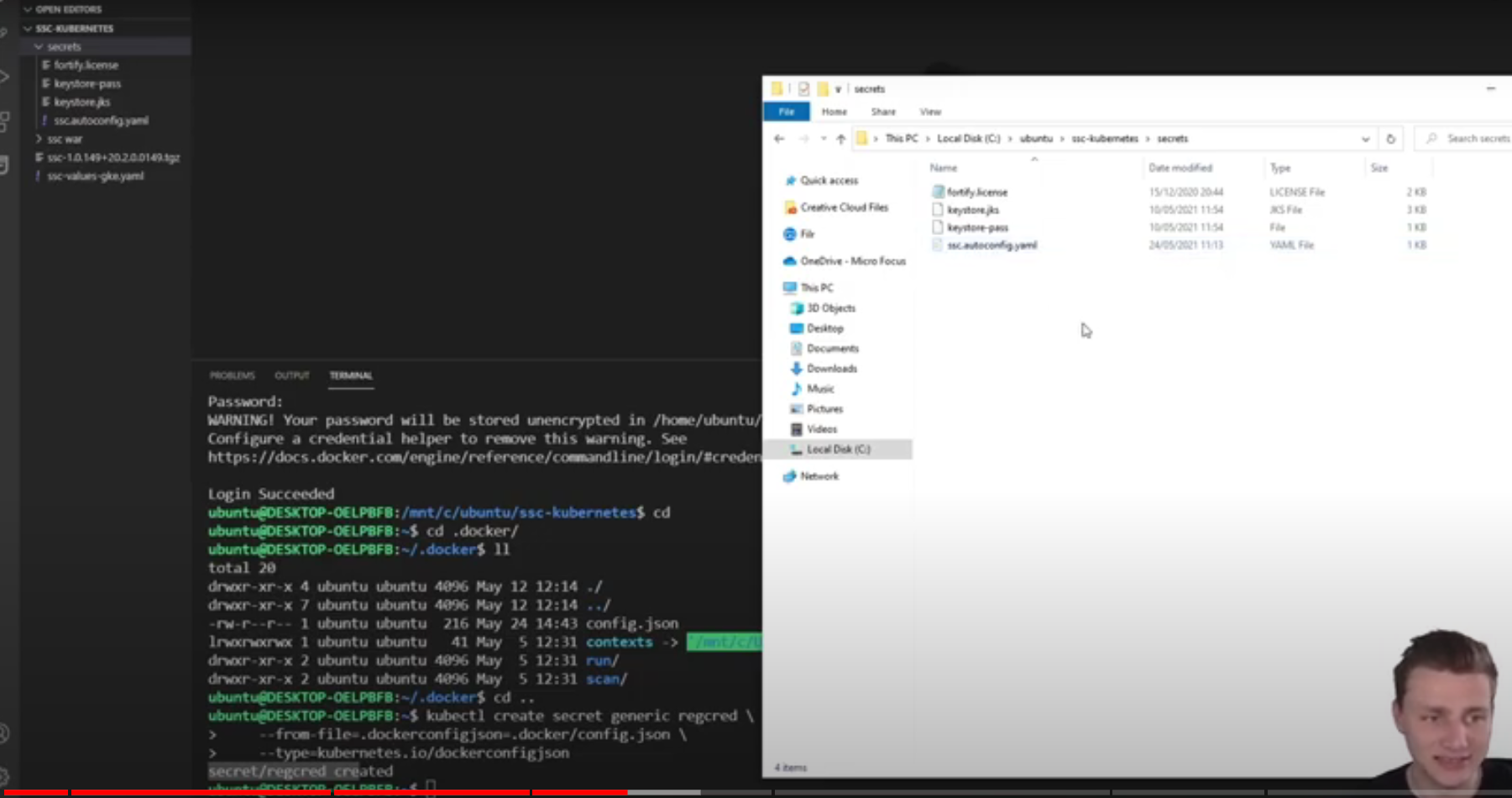
<https://www.youtube.com/watch?v=1NZxXwclRQY>

<https://github.com/open-cluster-management/policy-collection/tree/main/deploy>

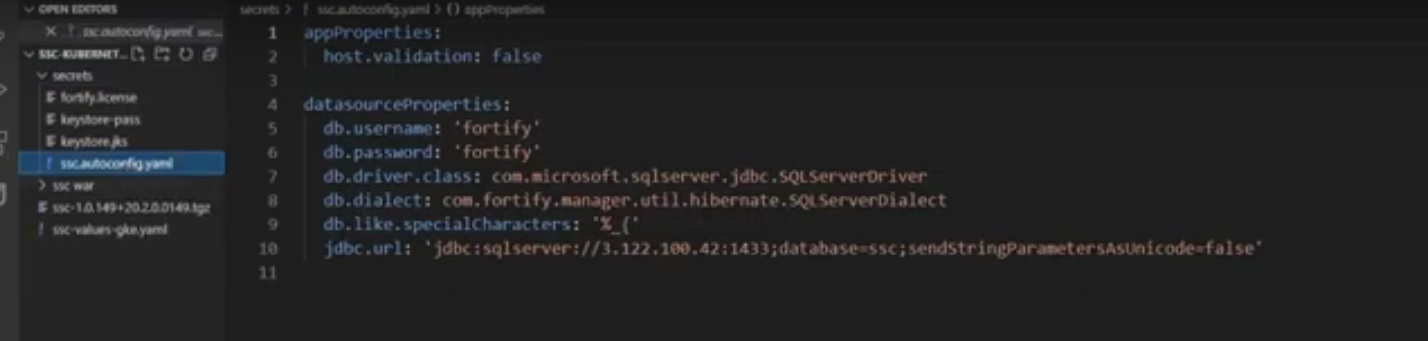
<https://github.com/open-cluster-management/governance-policy-framework>

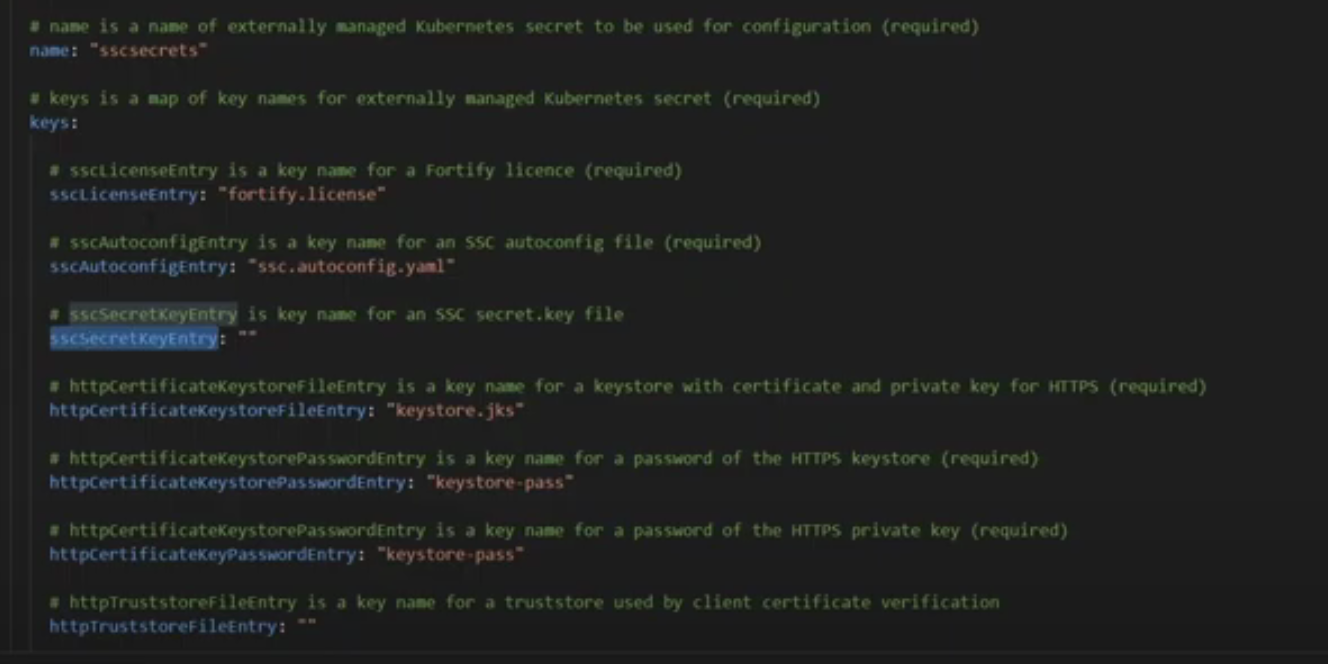
<https://docs.openshift.com/container-platform/4.5/security/pod-vulnerability-scan.html>





Ssc.autoconfig.yaml





<https://github.com/quay/container-security-operator>

**Configuration Policy Controller**

With the Configuration Policy Controller, you can create ConfigurationPolicies to check if the specified objects are present in the cluster. The controller records compliancy details in the status of each ConfigurationPolicy, and as Kubernetes Events. If the policy is set to enforce the configuration, then the controller will attempt to create, update, or delete objects on the cluster as necessary to match the specified state. The controller can be run as a stand-alone program or as an integrated part of governing risk with the Open Cluster Management project.

**ConfigurationPolicy** : spec includes the following fields:

Severity : Optional: low, medium, or high.’

remediationAction: Required: inform or enforce. Determines what actions the controller will take if the actual state of the object-templates does not match what is desired.

namespaceSelector: Optional: an object with include and exclude lists, specifying where the controller will look for the actual state of the object-templates, if the object is namespaced and not already specified in the object.

object-templates: Required: A list of Kubernetes objects that will be checked on the cluster.

Additionally, each item in the **object-templates** includes these fields:

complianceType: Required: musthave, mustnothave or mustonlyhave. Determines how to decide if the cluster is compliant with the policy.

objectDefinition: Required: A Kubernetes object which must (or must not) match an object on the cluster in order to comply with this policy.

### Templating

Configuration Policies supports inclusion of [Golang text templates](https://golang.org/pkg/text/template/) in ObjectDefinitions. These templates are resolved at runtime on the target cluster using configuration local to that cluster giving the user the ability to define policies customized to the target cluster. Following custom template functions are available to allow referencing kube-resources on the target cluster.

1. fromSecret - returns the value of the specified data key in the Secret resource
2. fromConfigMap - returns the values of the specified data key in the ConfigMap resource.
3. fromClusterClaim - returns the value of Spec.Value field in the ClusterClaim resource.
4. lookup - a generic lookup function to retreive any kube resource.

placement binding crd :

[*https://open-cluster-management.io/getting-started/core/cluster-manager*](https://open-cluster-management.io/getting-started/core/cluster-manager)

[*https://open-cluster-management.io/getting-started/integration/policy-framework*](https://open-cluster-management.io/getting-started/integration/policy-framework)

*kubectl get policy -n default policy-pod -o yaml*

*kubectl get policy*

[*https://access.redhat.com/documentation/en-us/openshift\_cluster\_manager/2021-02/html/managing\_clusters/assembly-what-is-ocm*](https://access.redhat.com/documentation/en-us/openshift_cluster_manager/2021-02/html/managing_clusters/assembly-what-is-ocm)

[*https://github.com/quay/container-security-operator/tree/master/deploy*](https://github.com/quay/container-security-operator/tree/master/deploy)

[*https://github.com/quay/container-security-operator/tree/master/deploy*](https://github.com/quay/container-security-operator/tree/master/deploy)

[*https://github.com/open-cluster-management/governance-policy-framework*](https://github.com/open-cluster-management/governance-policy-framework)

[*https://github.com/open-cluster-management/policy-collection/blob/main/stable/CM-Configuration-Management/policy-pod.yaml*](https://github.com/open-cluster-management/policy-collection/blob/main/stable/CM-Configuration-Management/policy-pod.yaml)

[*https://github.com/open-cluster-management/config-policy-controller/tree/main/deploy*](https://github.com/open-cluster-management/config-policy-controller/tree/main/deploy)

[*https://github.com/quay/container-security-operator/blob/master/deploy/imagemanifestvuln.crd.yaml*](https://github.com/quay/container-security-operator/blob/master/deploy/imagemanifestvuln.crd.yaml)

[*https://github.com/open-cluster-management/policy-collection/tree/main/stable*](https://github.com/open-cluster-management/policy-collection/tree/main/stable)

[*https://open-cluster-management.io/getting-started/integration/policy-controllers/*](https://open-cluster-management.io/getting-started/integration/policy-controllers/)

[*https://github.com/open-cluster-management/governance-policy-propagator/blob/main/deploy/rbac/role.yaml*](https://github.com/open-cluster-management/governance-policy-propagator/blob/main/deploy/rbac/role.yaml)

[*https://github.com/openshift/file-integrity-operator*](https://github.com/openshift/file-integrity-operator)