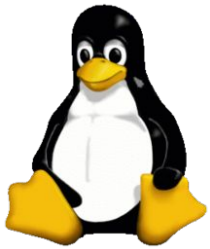
Getting to grips with Kubernetes RBAC

Liz Rice, Aqua Security

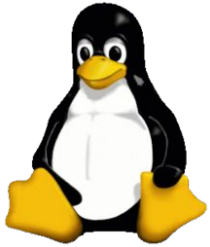
@lizrice | @aquasecteam


ning with Scissors
Liz Rice
echnology Ev
Aqu
O'REILLY
Compliments of aqua
Kubernetes Security
Operating Kubernetes Clusters and Applications Safely
Liz Rice & Michael Hausenblas
CLOUD NATIVE COMPUTING FOUNDATION
Technical Oversight Committee
Liz Rice
Aqua Security (TOC Chair)
OPENTRACING
kubernetes
CloudNativeCon
Prometheus
fluentd
CNI



```
$ ls -l
total 8
-rwxr-xr--  1 liz  staff  956  7 Mar 08:22 myapp
```

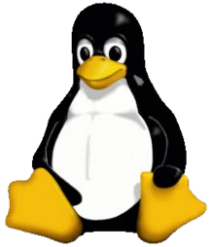
Files have owners



owner group

```
-rwxr-xr--  1 liz  staff  956  7 Mar 08:22 myapp
```

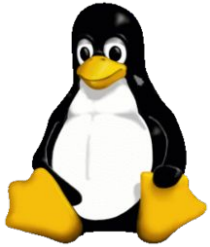
Files have owners and permissions

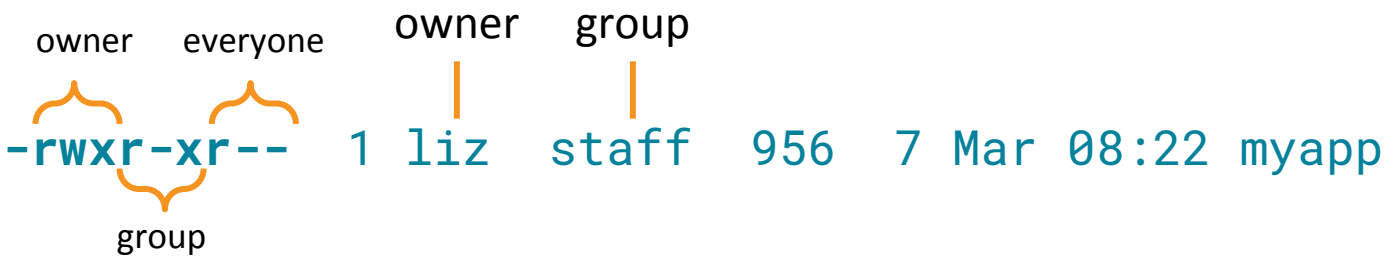


permissions owner group

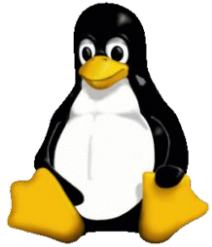
```
-rwxr-xr--  1 liz  staff  956  7 Mar 08:22 myapp
```

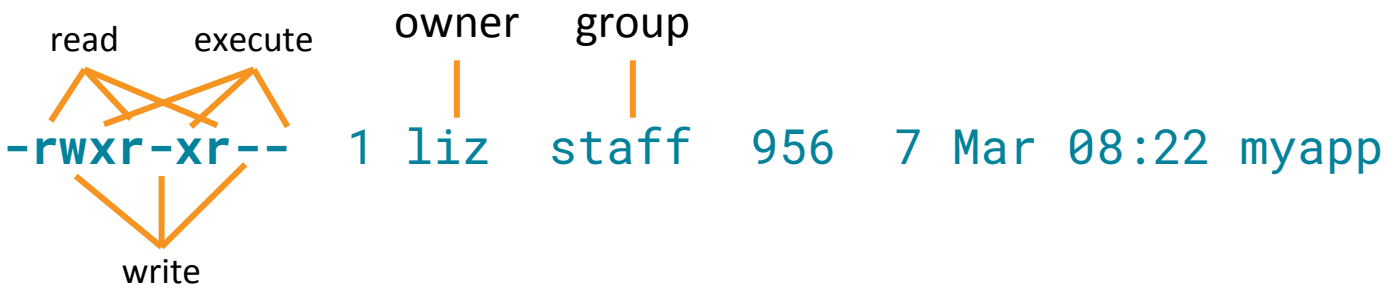
Files have owners and permissions



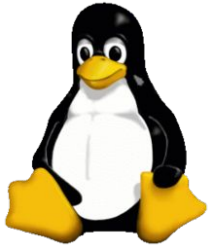


Files have owners and permissions





Files have owners and permissions



Read

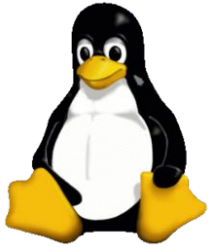
Anyone can read *myapp*

owner everyone owner group

```
-rwxr-xr--  1 liz  staff  956  7 Mar 08:22 myapp
```

group

Files have owners and permissions



Write

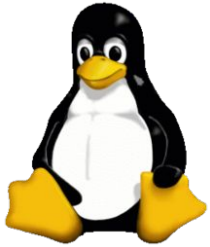
Only user *liz* can write *myapp*

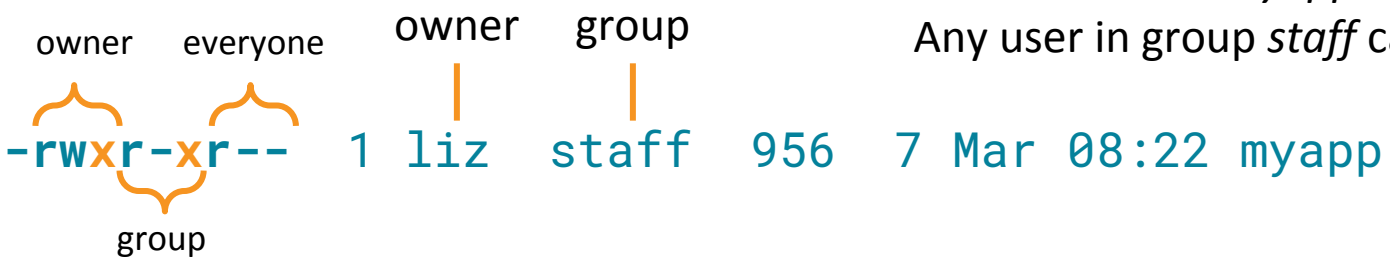
owner everyone owner group

```
-rwxr-xr--  1 liz  staff  956  7 Mar 08:22 myapp
```

group

Files have owners and permissions



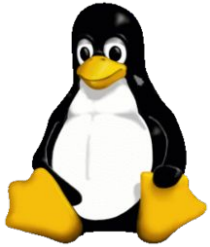


Execute

User *liz* can run *myapp*

Any user in group *staff* can run *myapp*

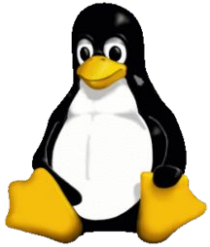
User is performing the action



owner group

```
-rwxr-xr--  1 liz  staff  956  7 Mar 08:22 myapp
```

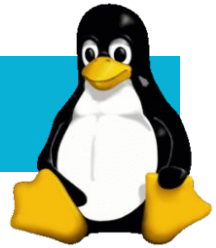
Verb might be permitted depending on
user making request, and their current group

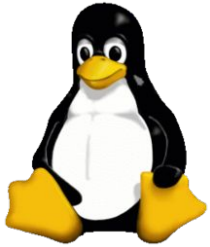


File

Permissions

Owner

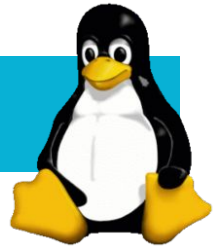
Linux		Kubernetes	
File		?	
File owner		?	
File permissions		?	



In Linux, everything is a file

In Kubernetes, everything is a resource

Linux

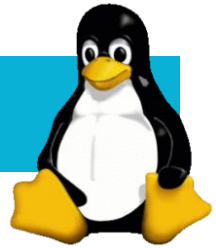


File
File owner
File permissions

Kubernetes

Resources
?
?

Linux



File

Kubernetes

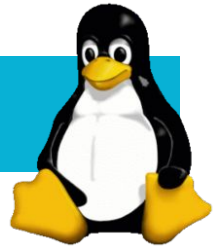
Resources

Example

Running *myapp*

Creating a pod to run *myapp*

Linux



File

File owner

File permissions

Kubernetes

Resources

Resources don’t have owners

Resources don’t have permissions

Role Based Access Control

Role Based Access Control

aqua

Role

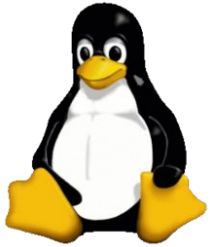
```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: my-role
  namespace: my-project
rules:
- apiGroups:
  - ""
  resources:
  - pods
  verbs:
  - create
  - get
  - list
```

Role

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: my-role
  namespace: my-project
rules:
- apiGroups:
  - ""
  resources:
  - pods
  verbs:
  - create
  - get
  - list
```

<verbs> you can do to <resources>

<verbs> you can do to a file called <name>

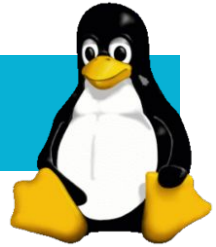


verbs

One specific, named resource

```
-rwxr-xr--  1 liz  staff  956  7 Mar 08:22 myapp
```

Linux



read

execute
write

Kubernetes

get
list
watch

create
delete
patch
update

use, bind, ...

<verbs> you can do to <resources>

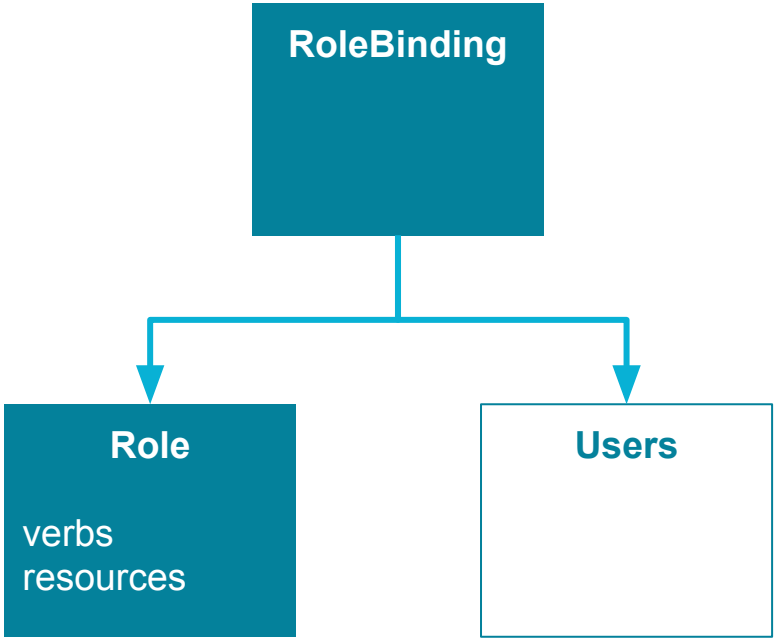
```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: my-role
  namespace: my-project
rules:
- apiGroups:
  - ""
  resources:
  - pods
  verbs:
  - get
  - list
```

<verbs> you can do to <resources> called <name>

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: my-role
  namespace: my-project
rules:
- apiGroups:
  - ""
  resources:
  - pods
  resourceNames:
  - myapp
  verbs:
  - get
  - list
```

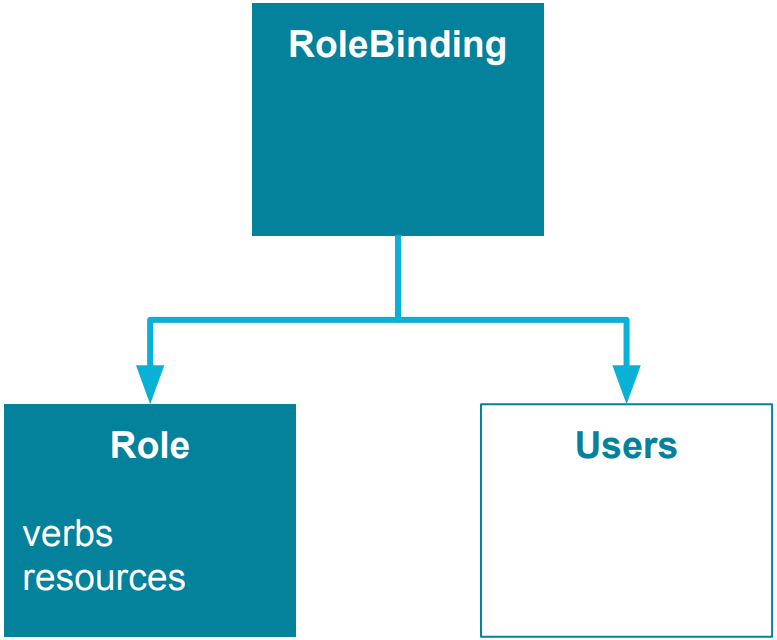
Role Based Access Control

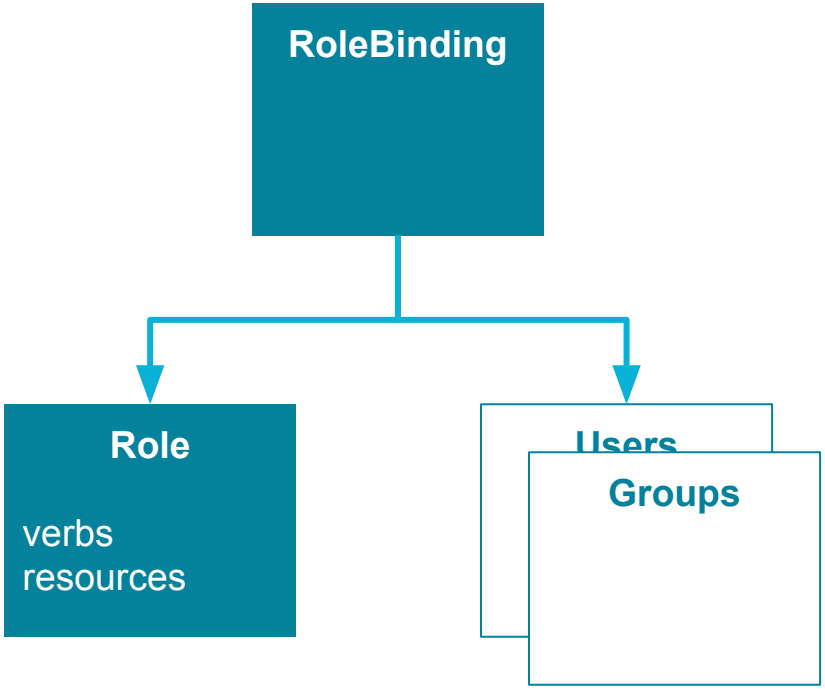

Role
verbs
resources
Users

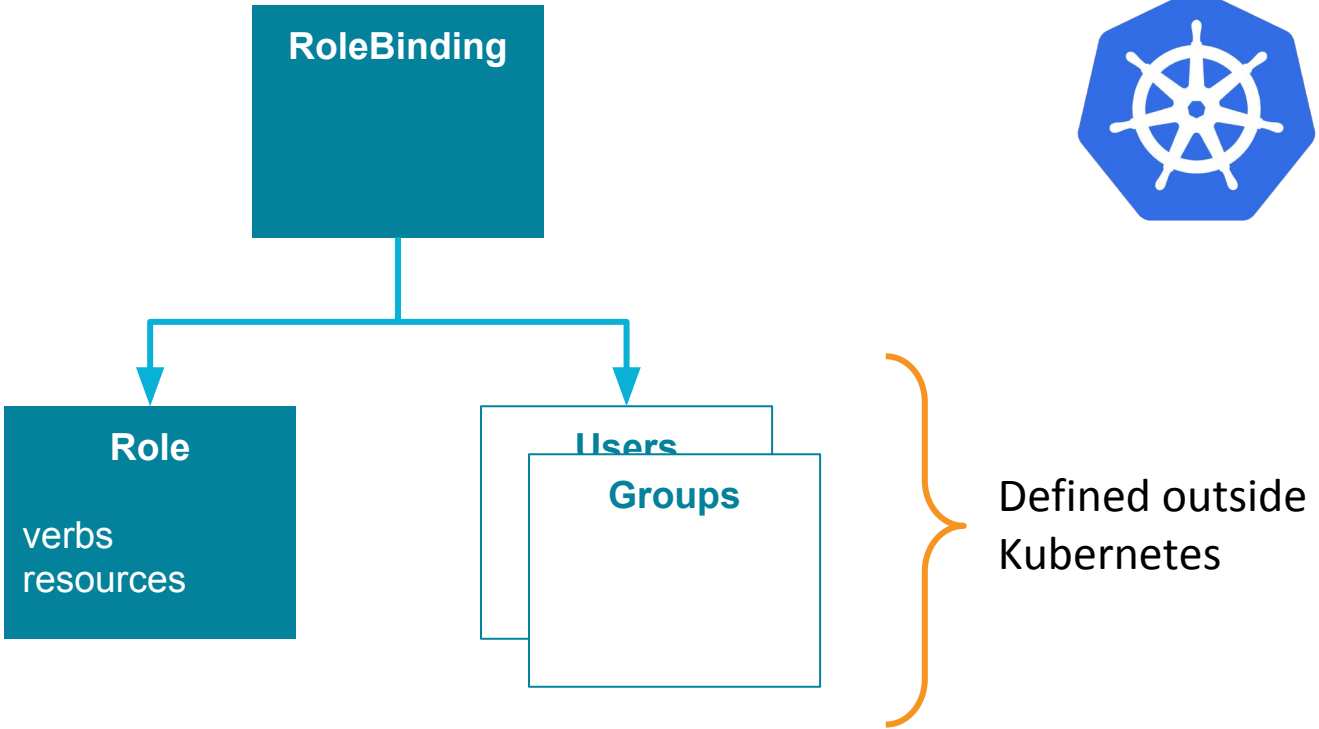

RoleBinding
Role
verbs
resources
Users

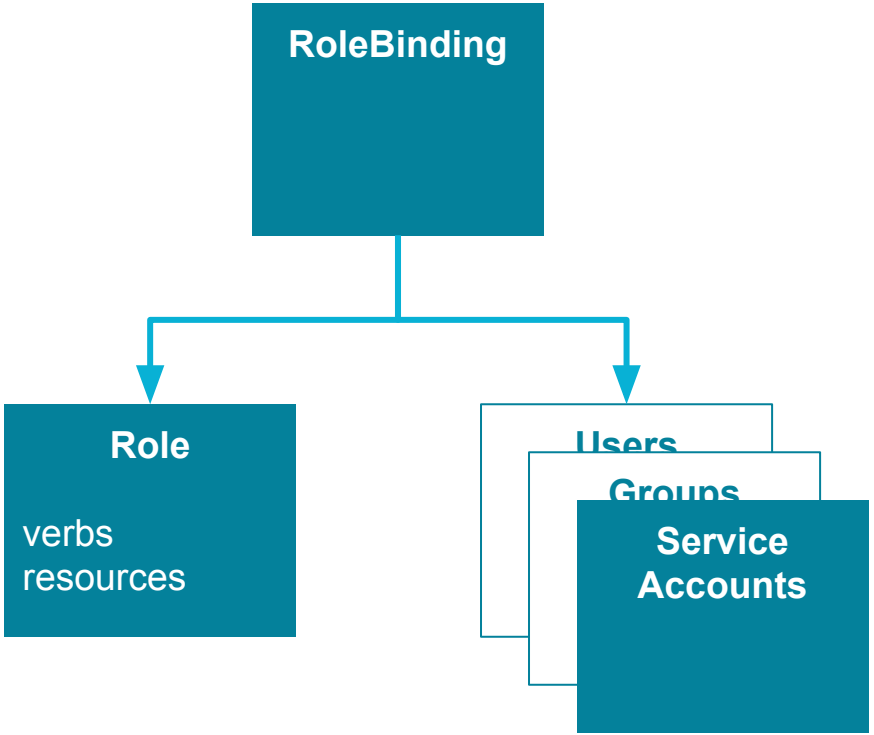
RoleBinding

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: my-role-binding
  namespace: default
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: my-role
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: User
  name: liz
```


RoleBinding
Role
verbs
resources
Users

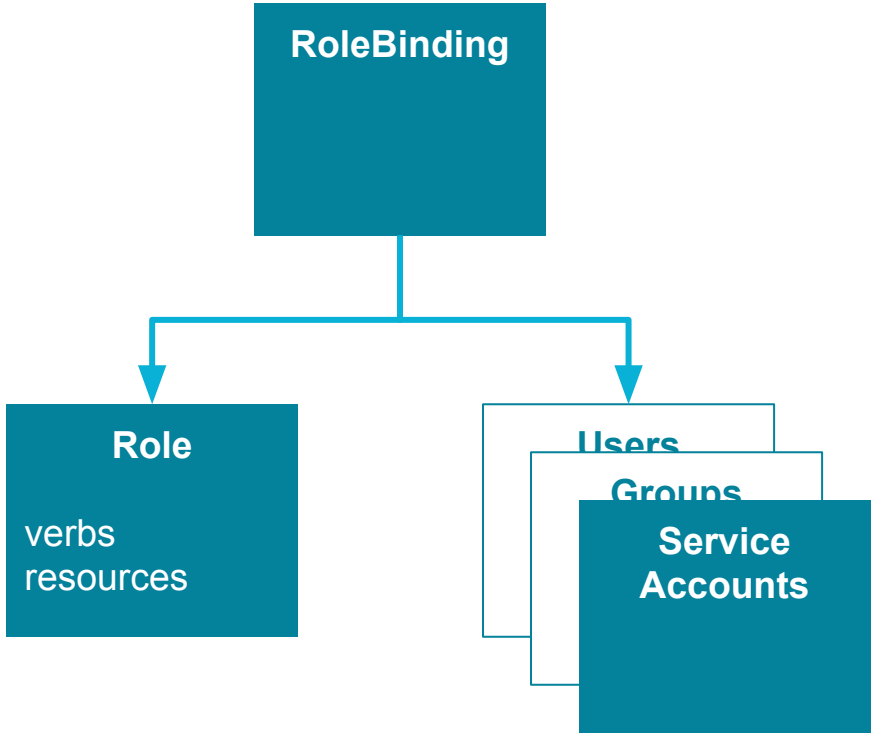

RoleBinding
Role
verbs
resources
Users
Groups

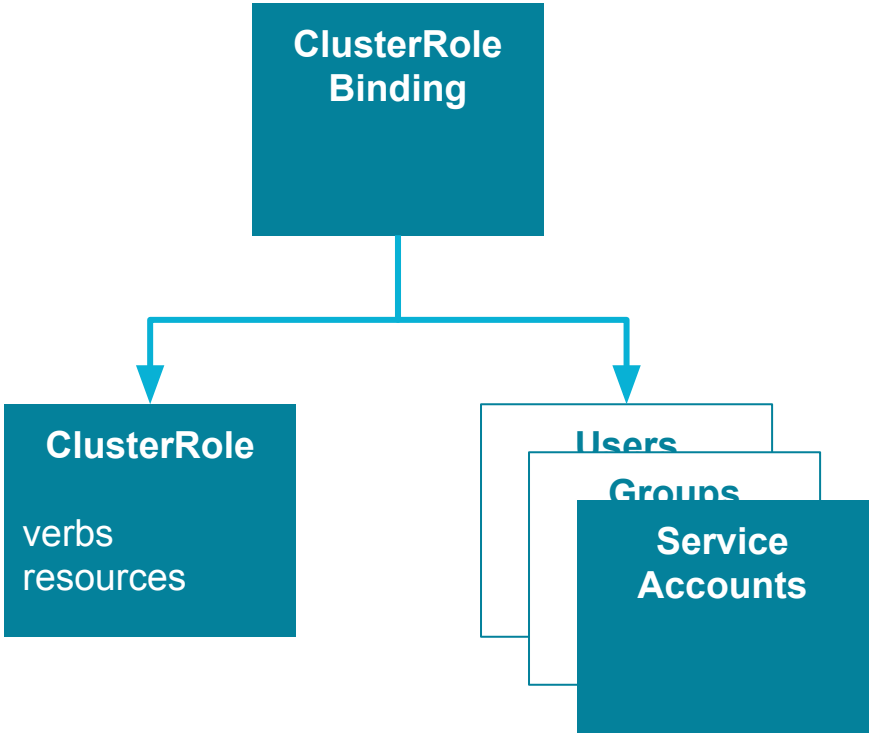

RoleBinding
Role
verbs
resources
Users
Groups
Defined outside
Kubernetes

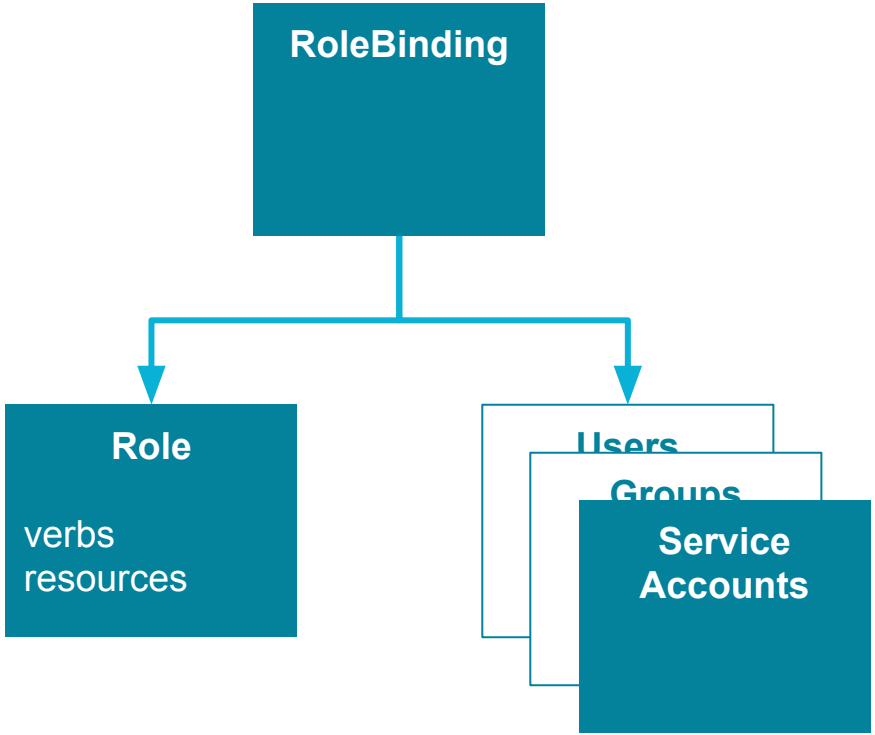


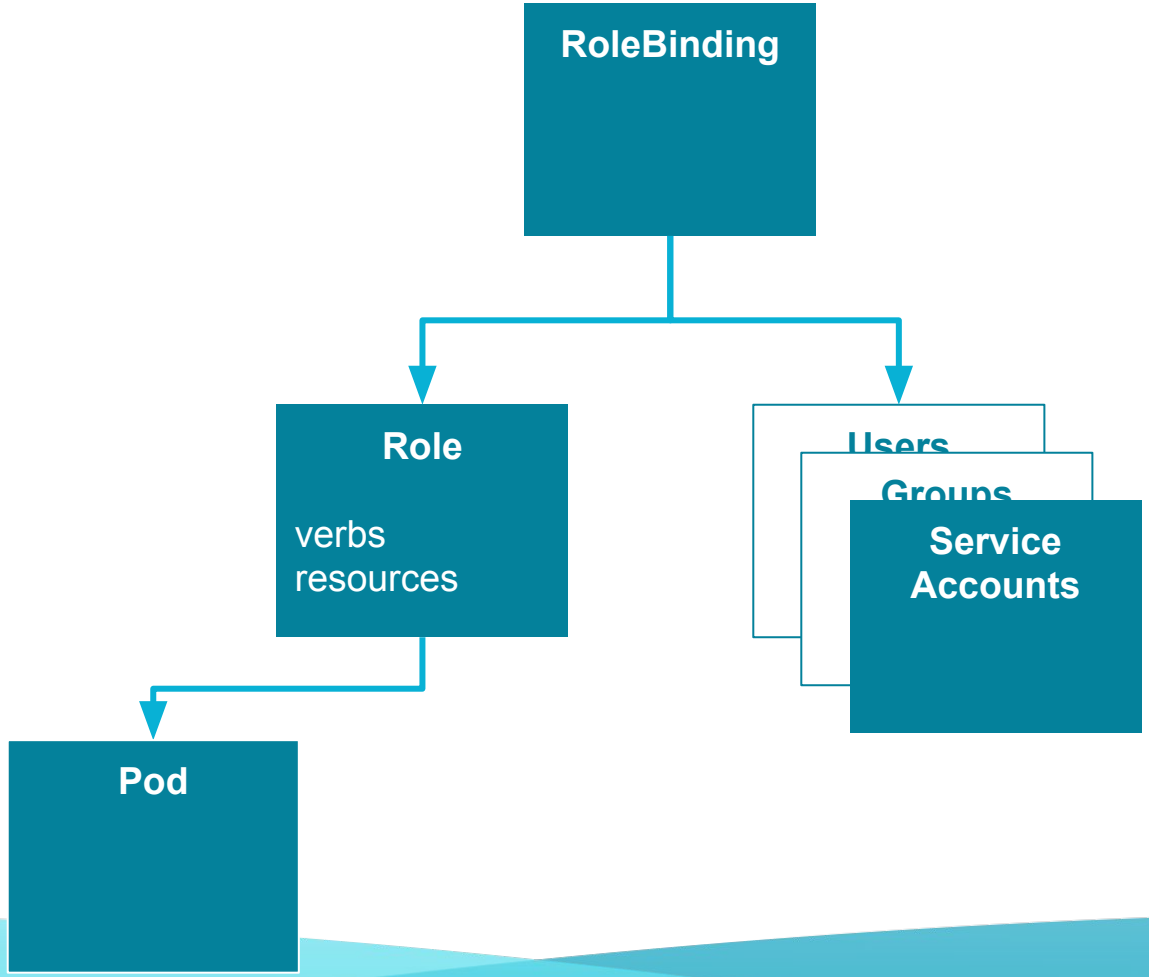
Identities within the cluster

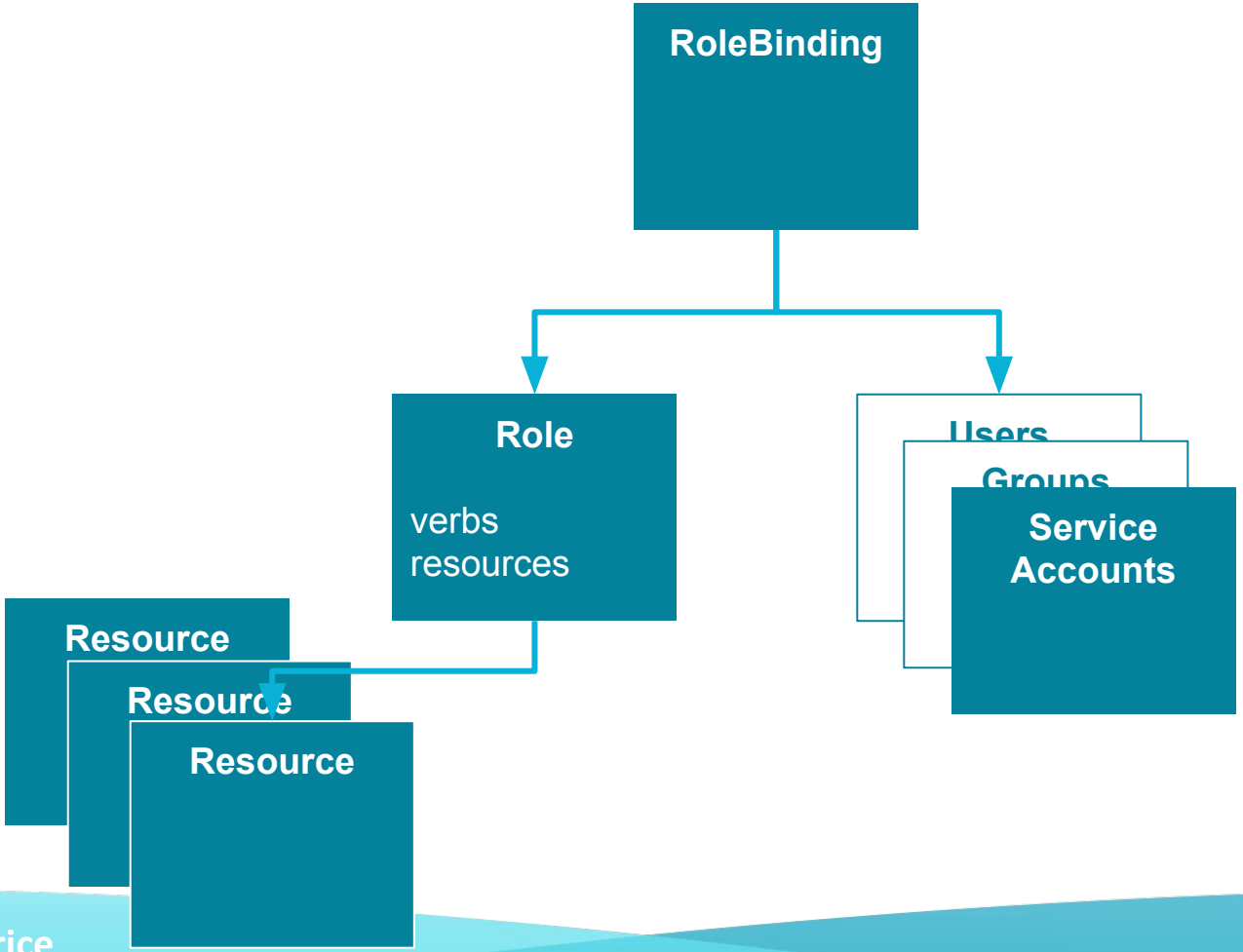
Roles are namespaced

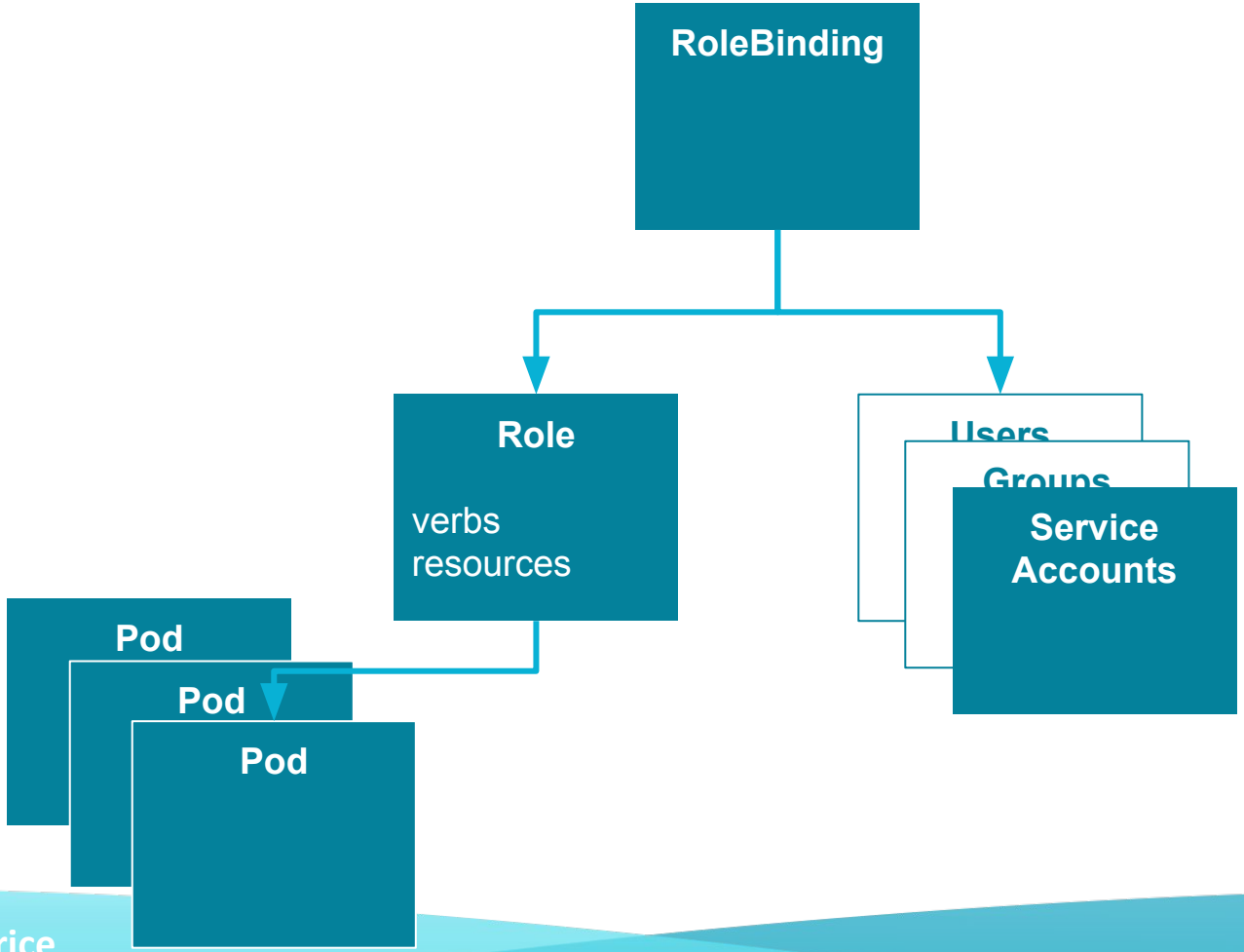


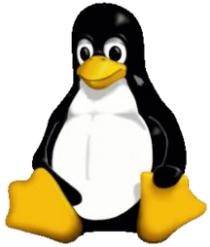

ClusterRole
Binding
ClusterRole
verbs
resources
Users
Groups
Service
Accounts


RoleBinding
Role
verbs
resources
Users
Groups
Service
Accounts


RoleBinding
Role
verbs
resources
Users
Groups
Service Accounts
Pod


RoleBinding
Role
verbs
resources
Users
Groups
Service Accounts
Resource
Resource
Resource


RoleBinding
Role
verbs
resources
Users
Groups
Service Accounts
Pod
Pod
Pod



File

Permissions

Owner

Entropy always increases with time

Second law of thermodynamics

Entropy of permissions always increases with time

Second law of RBAC

Many ClusterRoleBindings to start with

```
$ kubectl get clusterrolebindings
NAME                                                   AGE
...
kubeadm:kubelet-bootstrap                              20d
kubeadm:node-autoapprove-bootstrap                     20d
kubeadm:node-autoapprove-certificate-rotation          20d
kubeadm:node-proxier                                   20d
system:aws-cloud-provider                              20d
system:basic-user                                      20d
system:controller:attachdetach-controller              20d
system:controller:certificate-controller               20d
system:controller:clusterrole-aggregation-controller   20d
system:controller:cronjob-controller                   20d
system:controller:daemon-set-controller                20d
system:controller:deployment-controller                20d
...
```

Widely-scoped RoleBindings

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: pod-reader
  namespace: default
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: pod-reader
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: everyone
```

Widely-scoped Roles

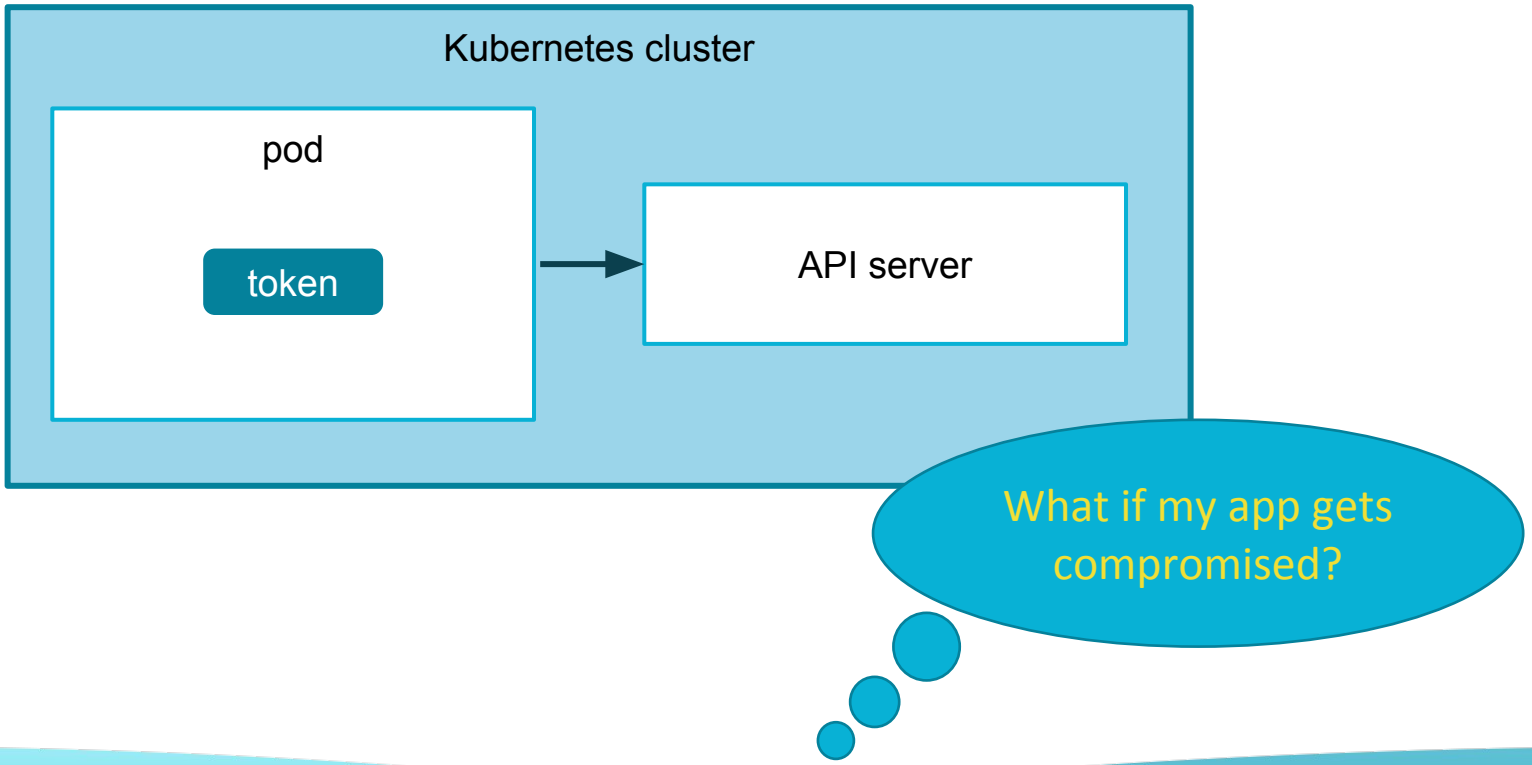
```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: my-role
  namespace: my-project
rules:
- apiGroups:
  - ""
  resources:
  - pods
  verbs:
  - “*”
```

Permissions are additive

If any role grants you a permission, you have that permission

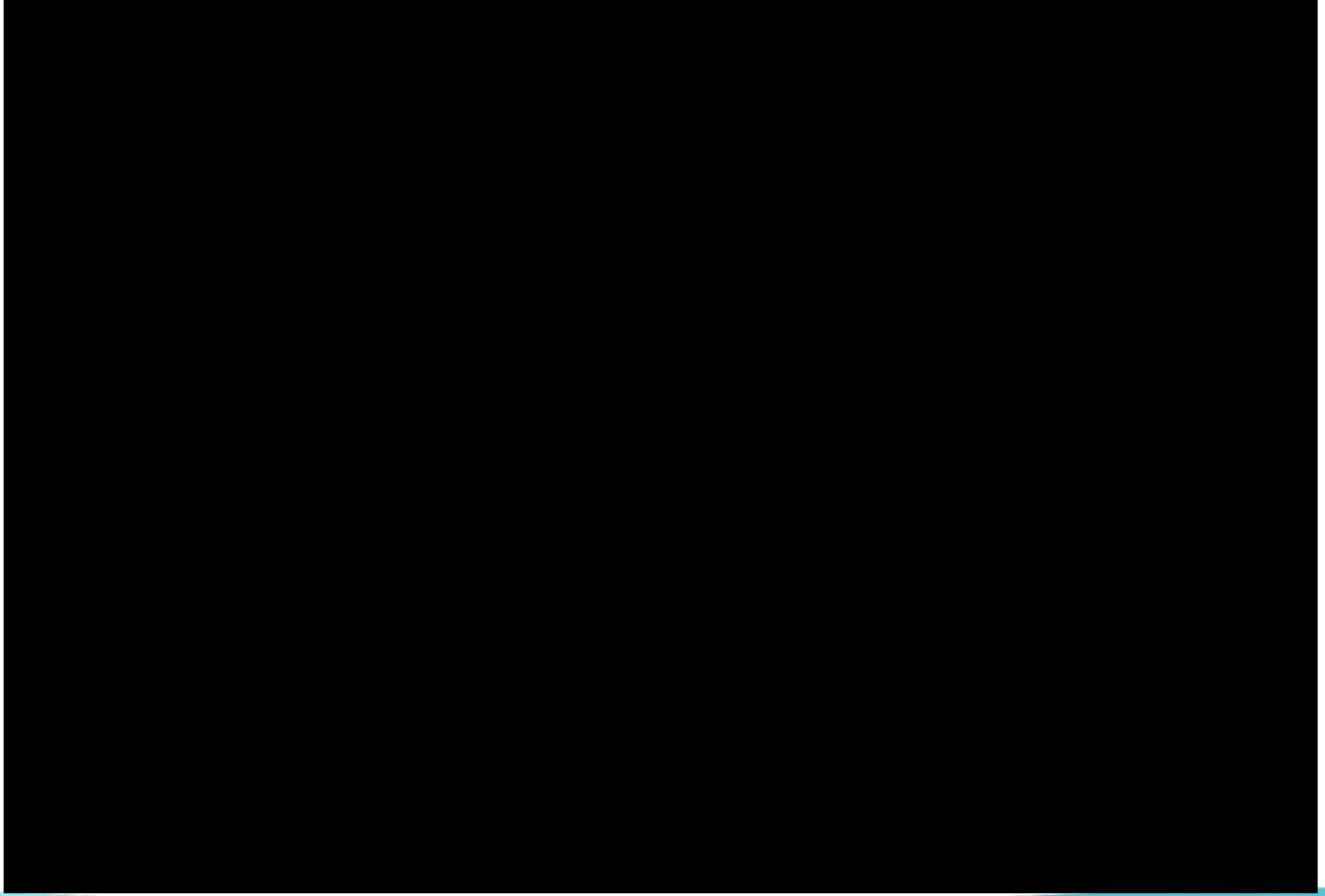
More subjects can do more things

Greater risk that one of those subjects is bad


Kubernetes cluster
pod
token
API server
What if my app gets compromised?

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: superpower
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: super-binding
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
- kind: ServiceAccount
  name: superpower
  namespace: default
```

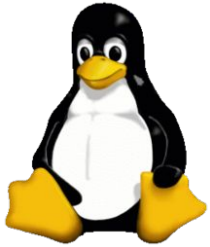
```
apiVersion: v1
kind: Pod
metadata:
  name: curl
spec:
  serviceAccountName: superpower
  containers:
  - name: curl
    image: tutum/curl
    command: ["sleep", "9999999"]
```



As an admin, how can you check permissions?

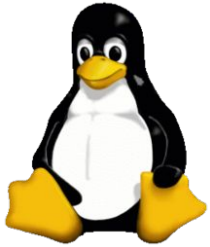
Can <subject> <verb> a <resource>?

Can I execute this file?



```
-rwxr-xr--  1 liz  staff  956  7 Mar 08:22 myapp
```

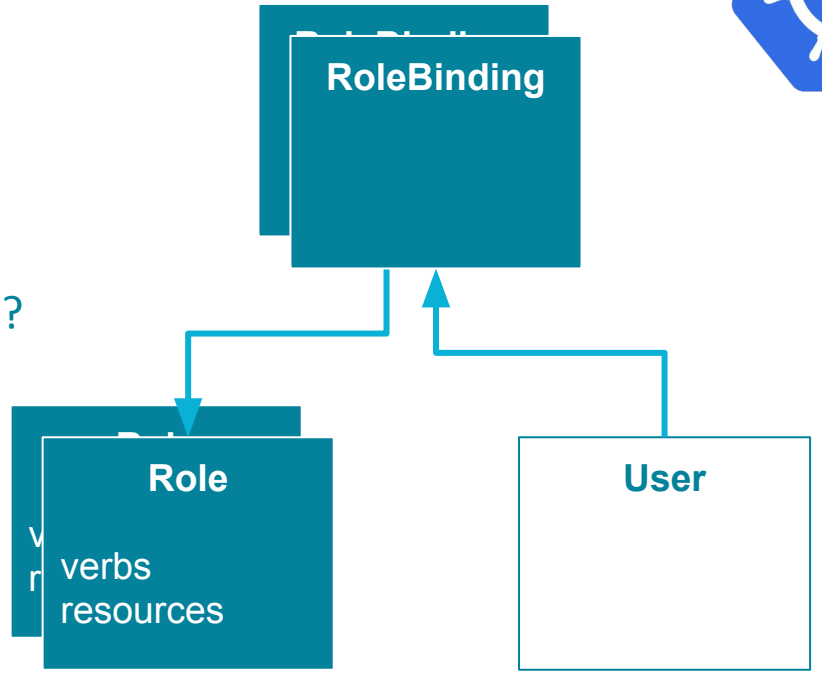
Am I liz, or a member of staff?



```
$ id
uid=501(liz) gid=20(staff)
groups=20(staff),501(access_bpf),12(everyone),80(admin)
```

Can I create a pod?

Which user am I?
What RoleBindings reference me?
What Roles do these bindings reference?
Do any of those Roles allow creating pods?



kubectl auth can-i

Can I (as <user>) <verb> a <resource>?

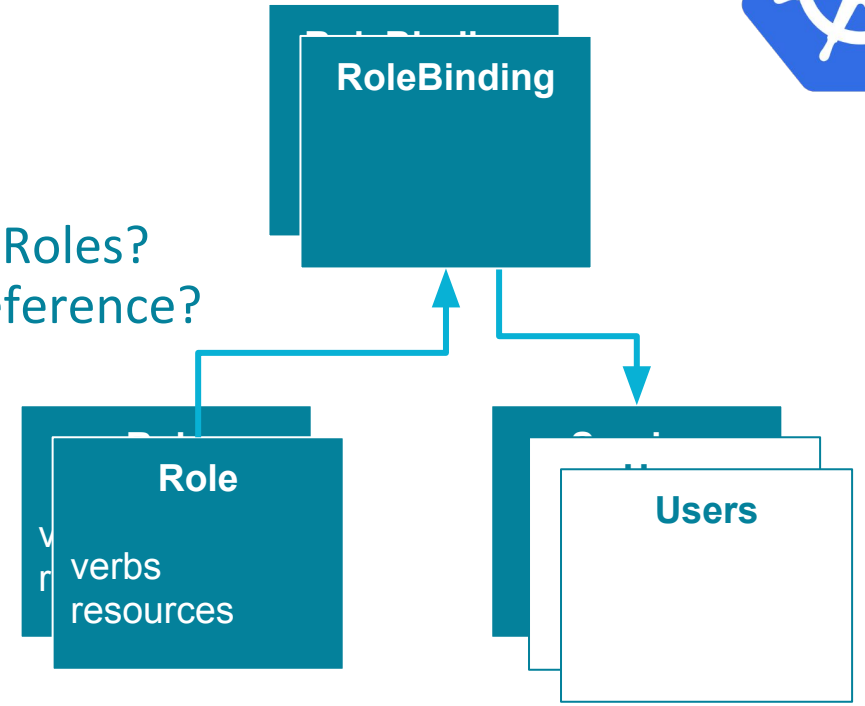
Can I create a pod?

```
$ kubectl auth can-i create pods --as=liz
no

$ kubectl auth can-i list pods --as=liz
yes
```

Who can <verb> <resources>?

Who can create a pod?

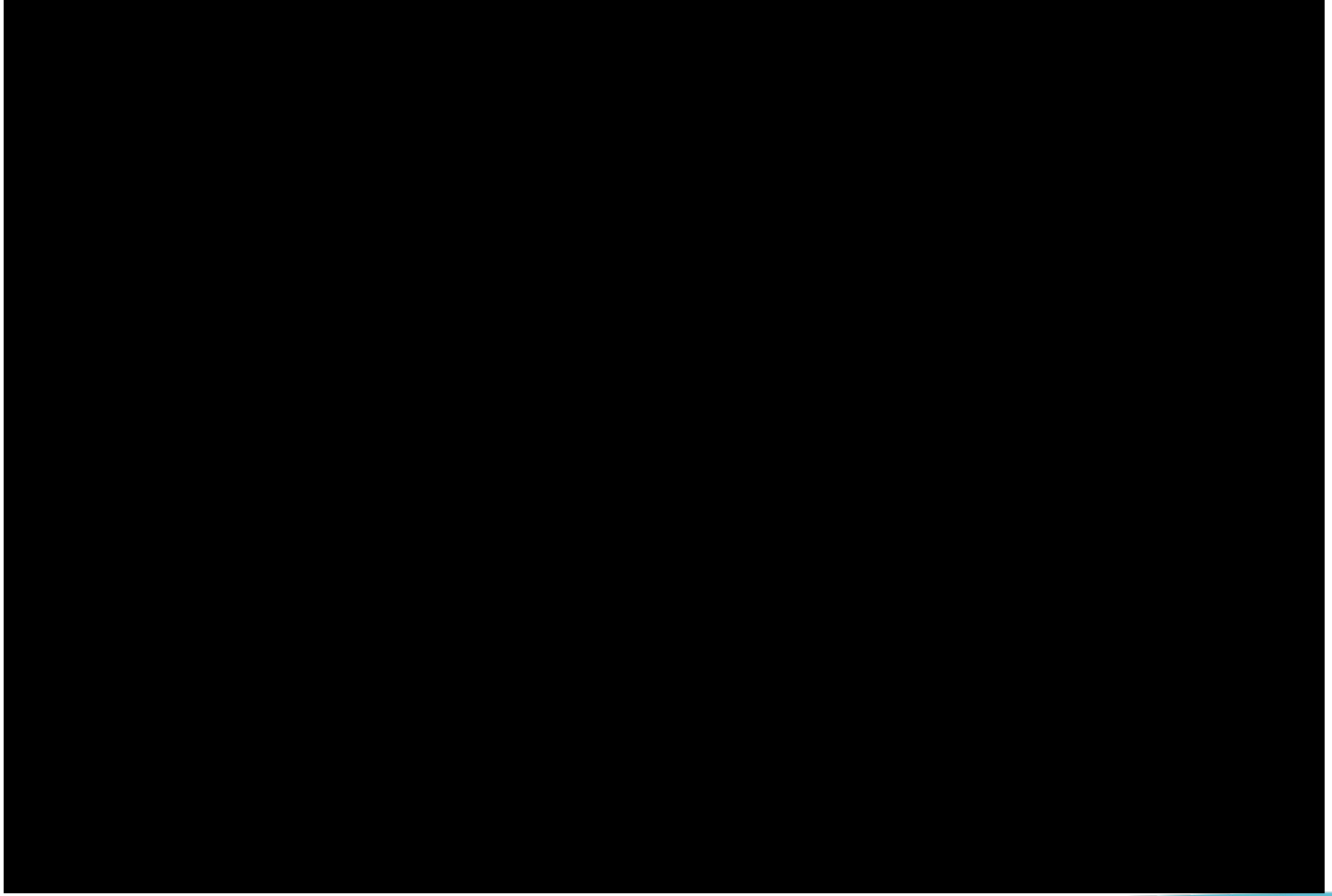
What Roles allow creating pods?
What RoleBindings reference those Roles?
Which subjects do these bindings reference?



kubectl auth who-can

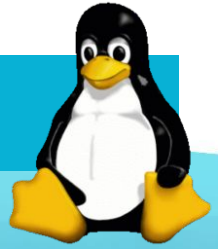
Who can <verb> <resources>?

Proof of concept:
`github.com/aquasecurity/kubectl-who-can`



Who can <verb> <resources>?

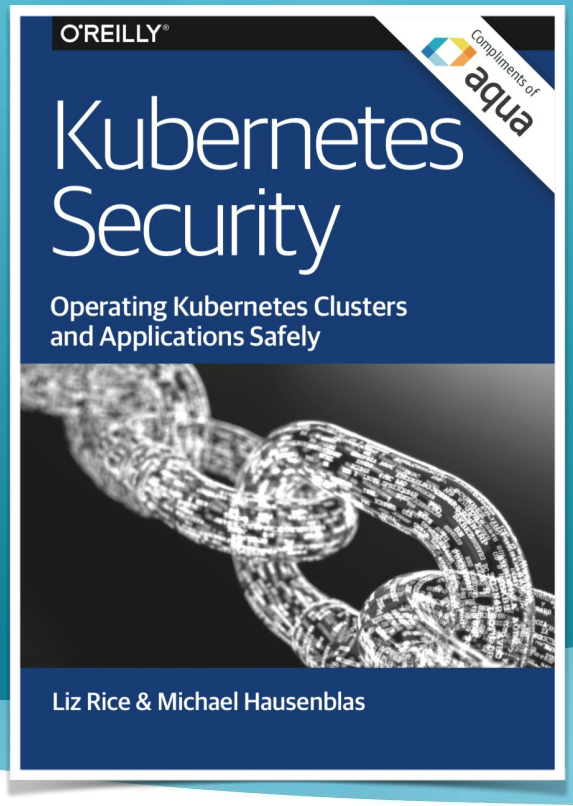
```
$ kubectl-who-can get pods
ROLEBINDING          NAMESPACE  SUBJECT  TYPE            SA-NAMESPACE
cross-namespaces     blah       default  ServiceAccount  default
read-pods            default    liz      User
read-specific-pods   default    liz      User

CLUSTERROLEBINDING                       SUBJECT                  TYPE
cluster-admin                            system:masters           Group
system:controller:deployment-controller
                                         deployment-controller    ServiceAccount
system:controller:endpoint-controller
                                         endpoint-controller      ServiceAccount
...
```

Linux		Kubernetes
File		**Many** resources
r w x verbs		**Many** verbs
Permissions attributes of file		**Abstracted** through Roles & RoleBindings
User		*User* (externally defined) & ServiceAccount
Group		*Group* (externally defined)

It’s easier to ask for forgiveness than permission

It’s easier to configure permissions
than ask for forgiveness
if you get attacked

info.aquasec.com/ kubernetes-security



aqua