Captain Kube's
Shake Shop
CRD

The Future of Your CRDs – Evolving an API

Stefan Schimanski (Red Hat) & Mehdy Bohlool (Google)

@the_sttts @mbohlool

kubernetes

CR[!D] Versioning

& Conversion

Custom Resources

- Extend kubernetes API by adding new resources
  - **Declarative** through Custom Resource Definition
  - **Imperative** through aggregated API server
- Focusing on Custom Resource Definitions (CRDs)
- Custom Resource (CR) refers to declarative CRs through CRD

CR Versioning

since 1.11
as beta

```
apiVersion: apiextensions.k8s.io/v1beta1
kind: CustomResourceDefinition
spec:
  versions:
  - name: v1
    served: true
    storage: true
  - name: v1beta1
    served: true
    storage: false
  - name: v1alpha1
    served: true
    storage: false
  …
```

→

/apis/your.group/**v1**/your-resources/**foo**

/apis/your.group/**v1beta1**/your-resources/**foo**

/apis/your.group/**v1alpha1**/your-resources/**foo**

Per-version CRD settings

- validation

```
spec:
  versions:
  - name: v1beta1
    served: true
    storage: false
    schema:
      openAPIV3Schema:
        properties:
          hostPort:
            type: string
  - name: v1alpha1
    served: true
    storage: false
    schema:
      openAPIV3Schema:
        properties:
          host:
            type: string
          port:
            type: string
  …
```

- subResources

```
spec:
  versions:
  - name: v1beta1
    served: true
    storage: false
    subresources:
      scale:
        specReplicasPath: "serverCount"
  - name: v1alpha1
    served: true
    storage: false
    subresources:
      scale:
        specReplicasPath: "clusterSize"
  …
```

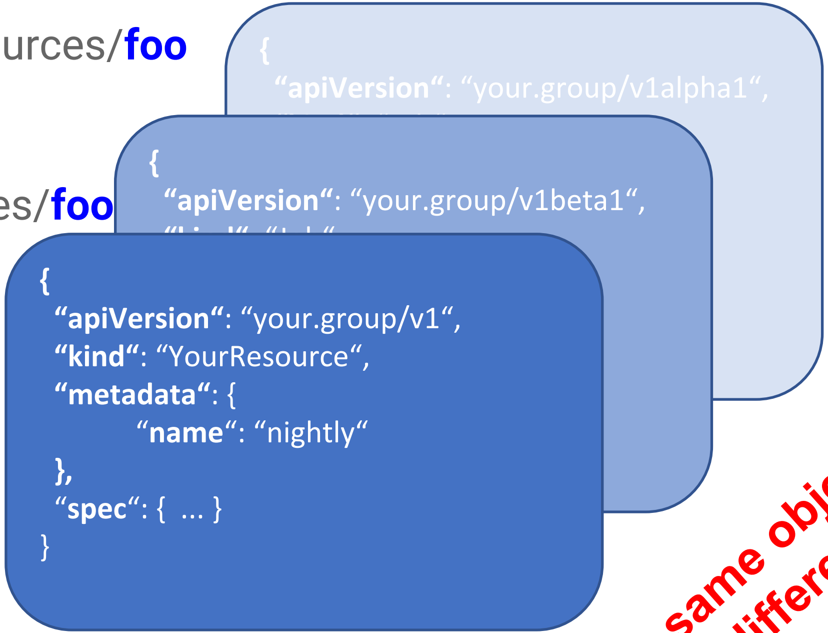
- additionalPrinterColumns

```
spec:
  versions:
  - name: v1beta1
    served: true
    storage: false
    additionalPrinterColumns:
    - JSONPath: "spec.nodes[*].name"
      name: "hosts"
      type: string
  - name: v1alpha1
    served: true
    storage: false
    additionalPrinterColumns:
    - JSONPath: "spec.servers[*].hosts"
      name: "hosts"
      type: string
  …
```

/apis/your.group/**v1alpha1**/your-resources/**foo**

/apis/your.group/**v1beta1**/your-resources/**foo**

/apis/your.group/**v1**/your-resources/**foo**



```
{
  "apiVersion": "your.group/v1alpha1",
```

```
{
  "apiVersion": "your.group/v1beta1",
```

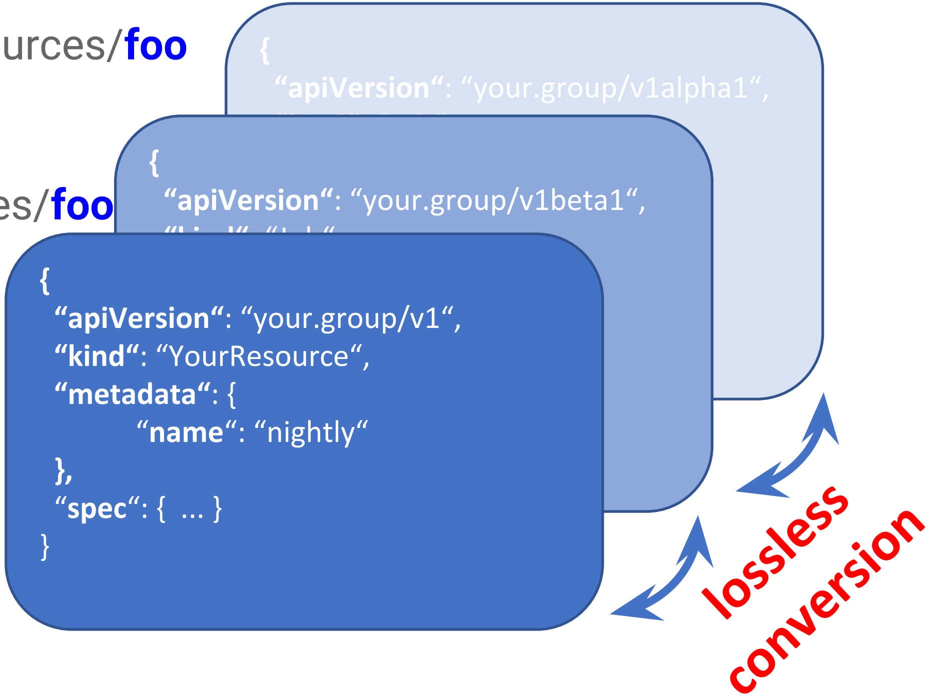
```
{
  "apiVersion": "your.group/v1",
  "kind": "YourResource",
  "metadata": {
          "name": "nightly"
  },
  "spec": {  ... }
}
```

the **same object foo**
in **three different versions**

/apis/your.group/**v1alpha1**/your-resources/**foo**

/apis/your.group/**v1beta1**/your-resources/**foo**

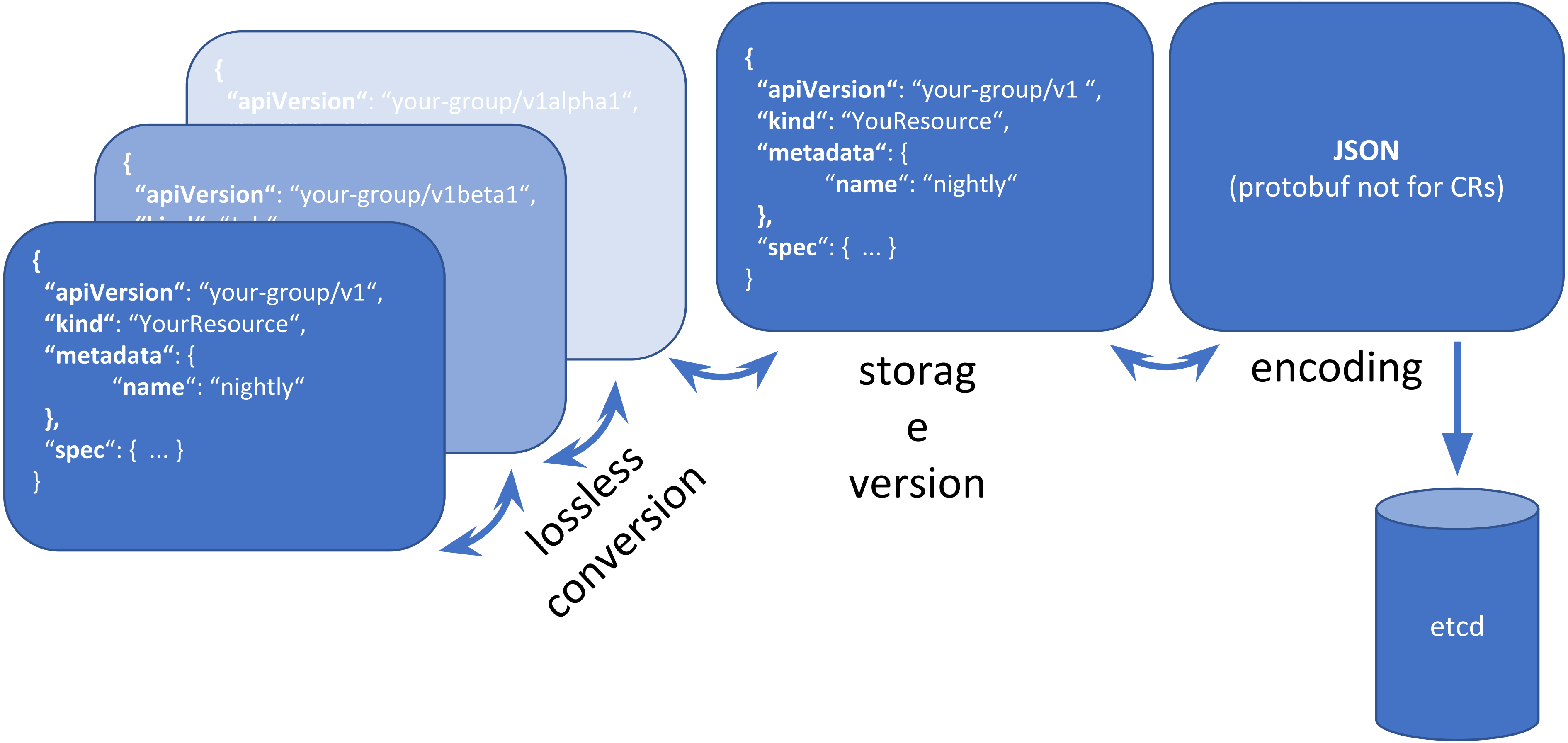
/apis/your.group/**v1**/your-resources/**foo**



```
{
  “apiVersion“: “your.group/v1alpha1“,
```

```
{
  “apiVersion“: “your.group/v1beta1“,
```

```
{
  “apiVersion“: “your.group/v1“,
  “kind“: “YourResource“,
  “metadata“: {
          “name“: “nightly“
  },
  “spec“: {  ... }
}
```



```
{
  "apiVersion": "your-group/v1alpha1",
```

```
{
  "apiVersion": "your-group/v1beta1",
```

```
{
  "apiVersion": "your-group/v1",
  "kind": "YourResource",
  "metadata": {
          "name": "nightly"
  },
  "spec": {  ... }
}
```

lossless conversion

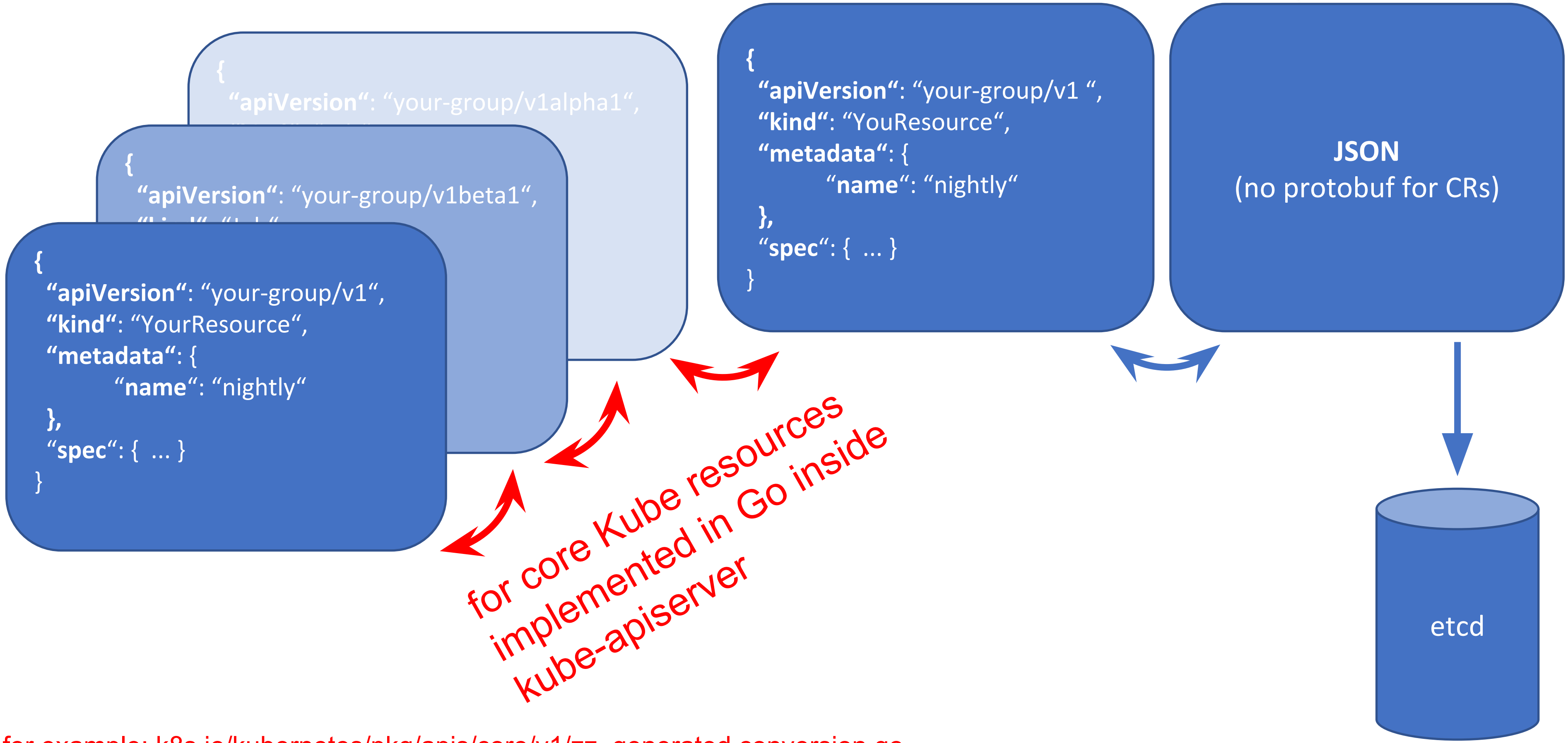
```
{
  "apiVersion": "your-group/v1 ",
  "kind": "YouResource",
  "metadata": {
          "name": "nightly"
  },
  "spec": {  ... }
}
```

storag
e
version

JSON
(protobuf not for CRs)

encoding

etcd



```
{
  “apiVersion“: “your-group/v1alpha1“,
```

```
{
  “apiVersion“: “your-group/v1beta1“,
```

```
{
  “apiVersion“: “your-group/v1“,
  “kind“: “YourResource“,
  “metadata“: {
        “name“: “nightly“
  },
  “spec“: {  … }
}
```

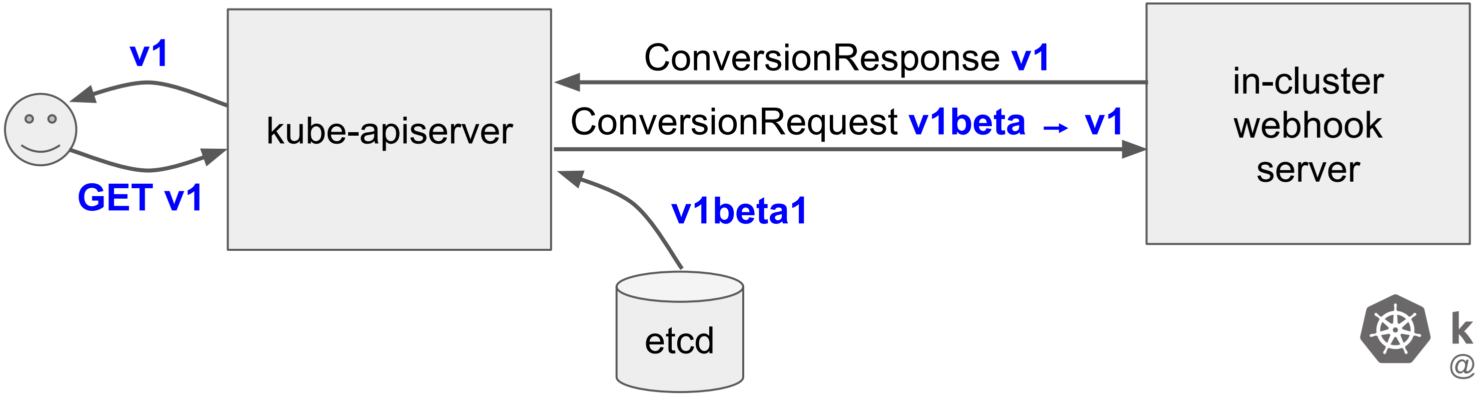
```
{
  “apiVersion“: “your-group/v1 “,
  “kind“: “YouResource“,
  “metadata“: {
        “name“: “nightly“
  },
  “spec“: {  … }
}
```

JSON
(no protobuf for CRs)
etcd
for core Kube resources implemented in Go inside kube-apiserver

for example: k8s.io/kubernetes/pkg/apis/core/v1/zz_generated.conversion.go

CR Conversion

- Since 1.11 as beta: None Conversion
  - Default implementation, “does nothing other than setting apiVersion”
- Declarative Conversion
  - Not implemented; was proposed but no consensus to implement
  - History/Discussions: bit.ly/k8s-crd-versioning and bit.ly/k8s-crd-conversion
- Since 1.13 as alpha: Webhook Conversion
  - similar to admission webhooks, but via **apiextensions.k8s.io/v1beta.ConversionReview**



kubernetes
@the_sttts @mbohlool

Webhook Configuration

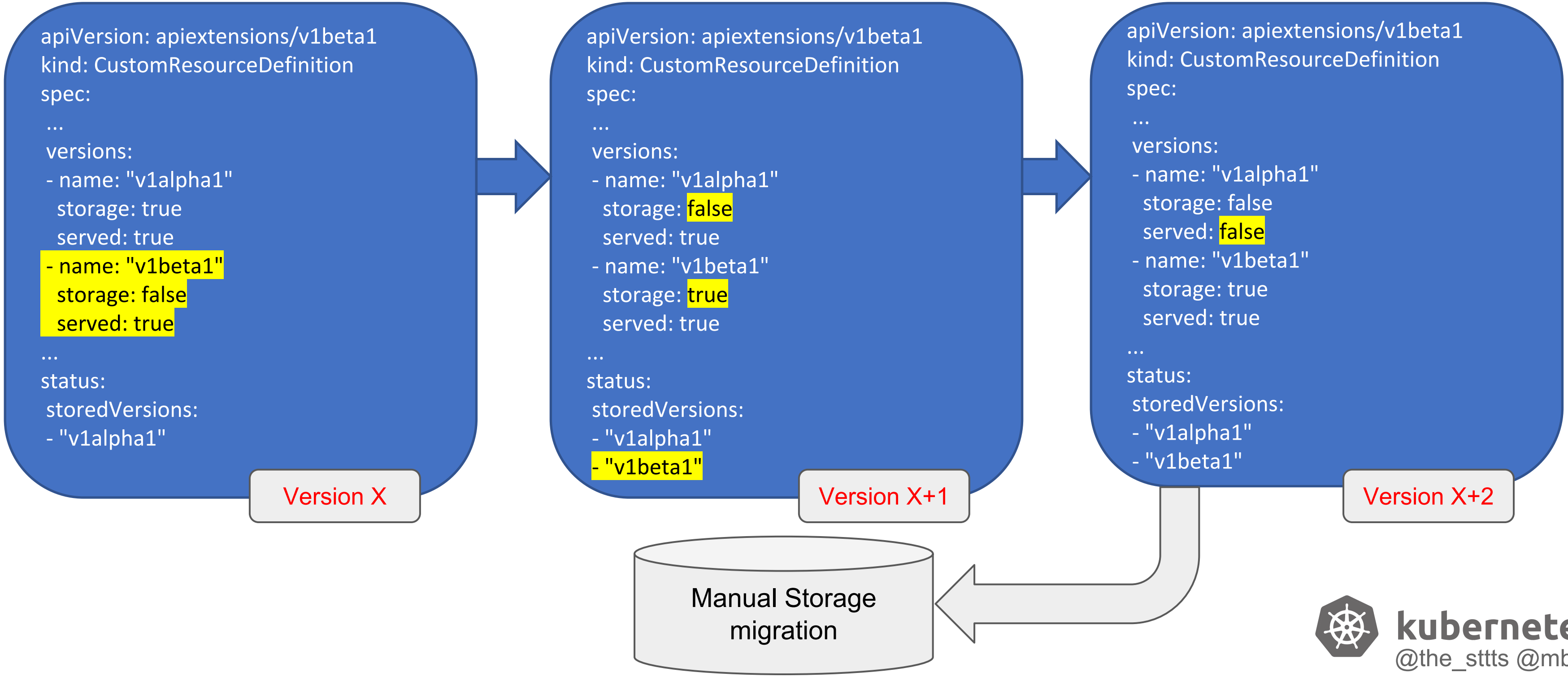
```
apiVersion: apiextensions.k8s.io/v1beta1
kind: CustomResourceDefinition
spec:
  versions:
  …
  conversion:
    Strategy: "webhook"
    webhookClientConfig:
      service:
        namespace: "mywebhook"
        name: "fooConverter"
    caBundle: ...

  …
```

should serve on port 443

pem encoded ca cert that signs the server cert

Demo

How To start adding versions



```
apiVersion: apiextensions/v1beta1
kind: CustomResourceDefinition
spec:
 ...
 versions:
 - name: "v1alpha1"
   storage: true
   served: true
 - name: "v1beta1"
   storage: false
   served: true
...
status:
 storedVersions:
 - "v1alpha1"
```

Version X

```
apiVersion: apiextensions/v1beta1
kind: CustomResourceDefinition
spec:
 ...
 versions:
 - name: "v1alpha1"
   storage: false
   served: true
 - name: "v1beta1"
   storage: true
   served: true
...
status:
 storedVersions:
 - "v1alpha1"
 - "v1beta1"
```

Version X+1

```
apiVersion: apiextensions/v1beta1
kind: CustomResourceDefinition
spec:
 ...
 versions:
 - name: "v1alpha1"
   storage: false
   served: false
 - name: "v1beta1"
   storage: true
   served: true
...
status:
 storedVersions:
 - "v1alpha1"
 - "v1beta1"
```

Version X+2

Manual Storage migration

Storage Versions and Rollback

- When should storage version be switched?
- **Be prepared for a rollback!**

In kube:

storage version lags 1 or 2 versions behind

```
apiVersion: apiextensions.k8s.io/v1beta1
kind: CustomResourceDefinition
spec:
  versions:
  - name: v1
    served: true
    storage: true
  - name: v1beta1
    served: true
    storage: false
  - name: v1alpha1
    served: true
    storage: false
  …
```

just introduced

Storage Migration

changing storage version in the CRD

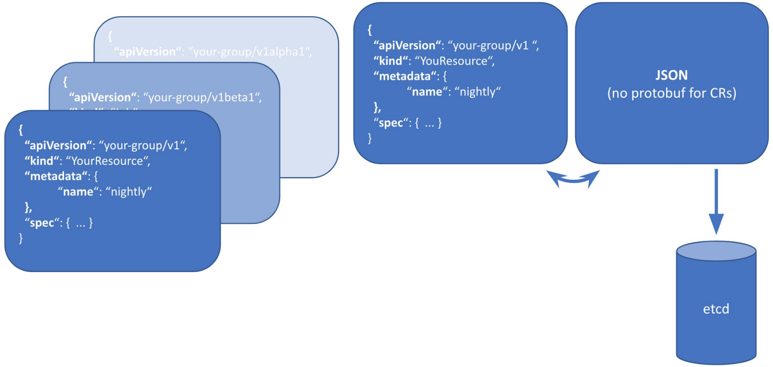
≠

version of objects in etcd magically changes

Storage version is used on WRITE to etcd.

Migration necessary:

- patch all objects with empty PATCH
- wait for storage migration mechanism
- Remove the old version from status.storedVersions



```
{
 "apiVersion": "your-group/v1alpha1",
```

```
{
 "apiVersion": "your-group/v1beta1",
```

```
{
 "apiVersion": "your-group/v1",
 "kind": "YourResource",
 "metadata": {
        "name": "nightly"
 },
 "spec": {  ... }
}
```

```
{
 "apiVersion": "your-group/v1 ",
 "kind": "YouResource",
 "metadata": {
        "name": "nightly"
 },
 "spec": {  ... }
}
```

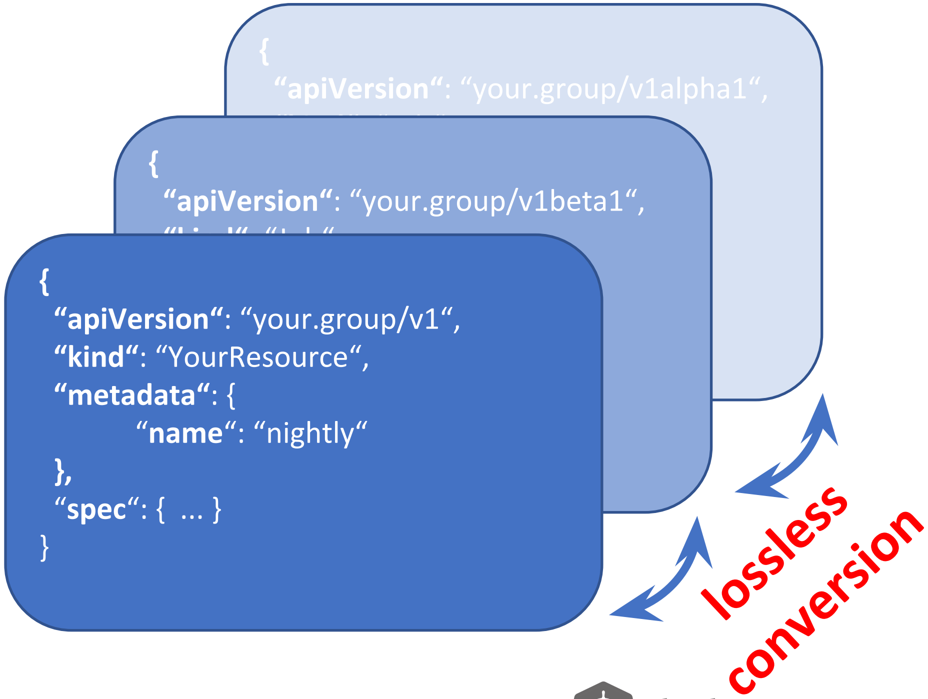
Roundtrip’ability

getting lossless conversion right is hard

established test pattern:

roundtrip test between all versions:

1. take YourResource e.g. in v1
2. fill with random values (“fuzzing”)
3. convert to e.g. v1beta1
4. convert back to v1
5. compare result with original
6. repeat



```
{
  “apiVersion“: “your.group/v1alpha1“,
```

```
{
  “apiVersion“: “your.group/v1beta1“,
```

```
{
  “apiVersion“: “your.group/v1“,
  “kind“: “YourResource“,
  “metadata“: {
          “name“: “nightly“
  },
  “spec“: {  … }
}
```

Example: k8s.io/sample-apiserver/blob/master/pkg/apis/wardle/install/roundtrip_test.go

Failure Modes

- What happens if your webhook fails completely?
  - **webhook call-out is on demand**
    ➪ other types continue working (standard or CRs)
    ➪ requests of your types in stored version still work
  - **for emergency:** patch/create/update/delete in the stored version
  - LIST request will fail if conversion is necessary for **even one** item
- If conversion fails for a subset of CRs?
- Webhook returns converted objects that are not round trippable
  - potentially **data loss**
  - silent failure which is the worst
  - **make sure it is round-trippable by running tests**

More about conversion and space for questions:

Sig API Machinery Deep Dive

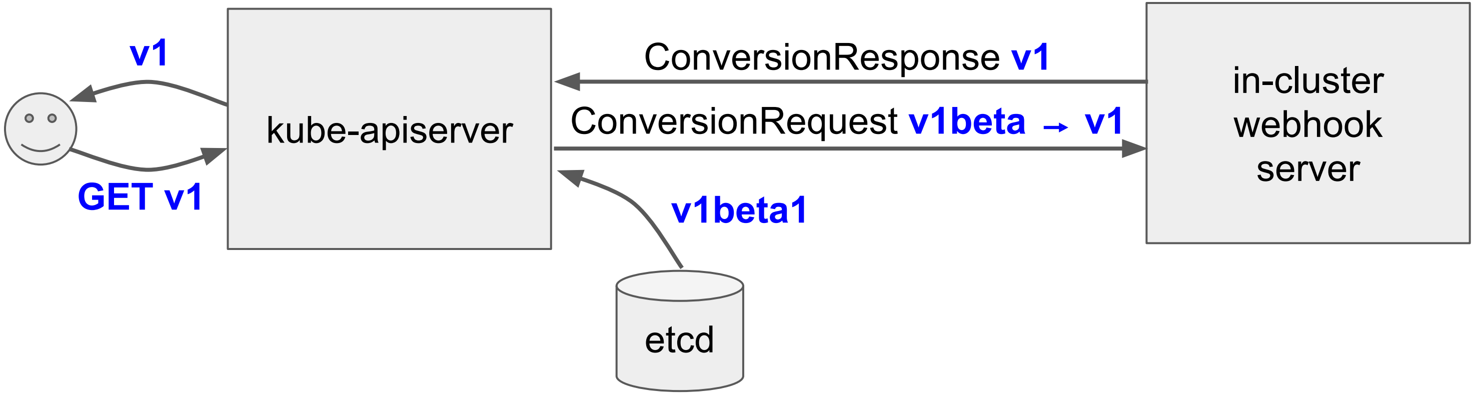
Thursday 11:40am 615-617

Outlook for 2019 and beyond

- **critical issues** identified for moving **CRDs to GA** (issue #58682):
  - **generate OpenAPI** spec from validation schema
➡ kubectl-side validation, kubectl explain
  - **quota** support
  - **defaulting** and **pruning**
  - … a number of smaller topics
- Other open, undecided topics:
  - validation schema language: alternative language, OpenAPI references, Go-types to schema
  - graceful deletion
  - validation webhook
  - arbitrary field selectors
  - protobuf encoding
  - namespace-local CRDs

CR Conversion

- Since 1.13 as alpha: Webhook Conversion
  - similar to admission webhooks, but via **apiextensions.k8s.io/v1beta.ConversionReview**



- Conversion is about user data
- Mistakes in conversion result in data loss
- Test **round-trip'ability**