Sig API Machinery Deep Dive

Daniel Smith (Google) – Stefan Schimanski (Red Hat)

kubernetes

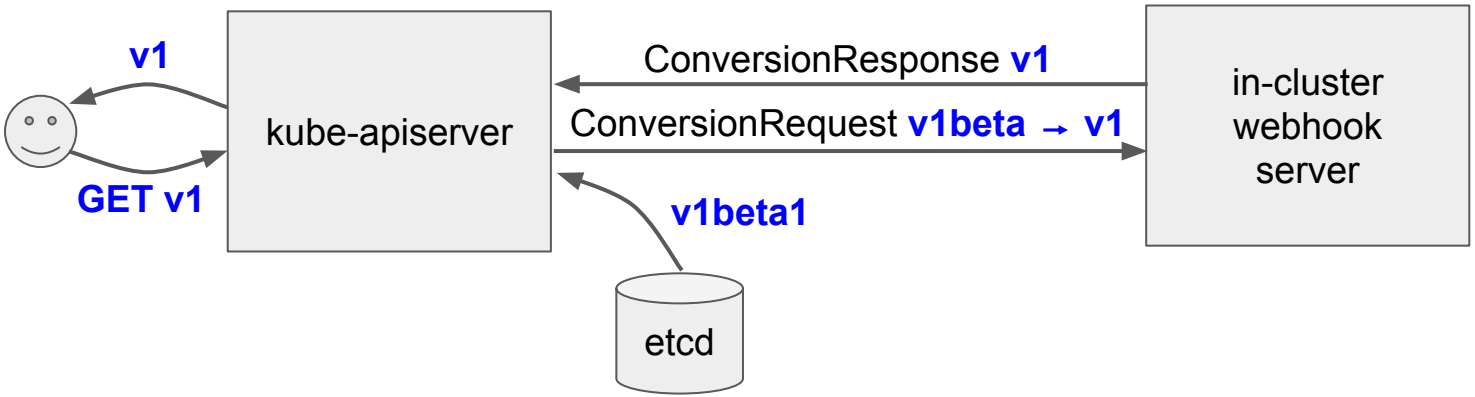
CR[!D] Versioning

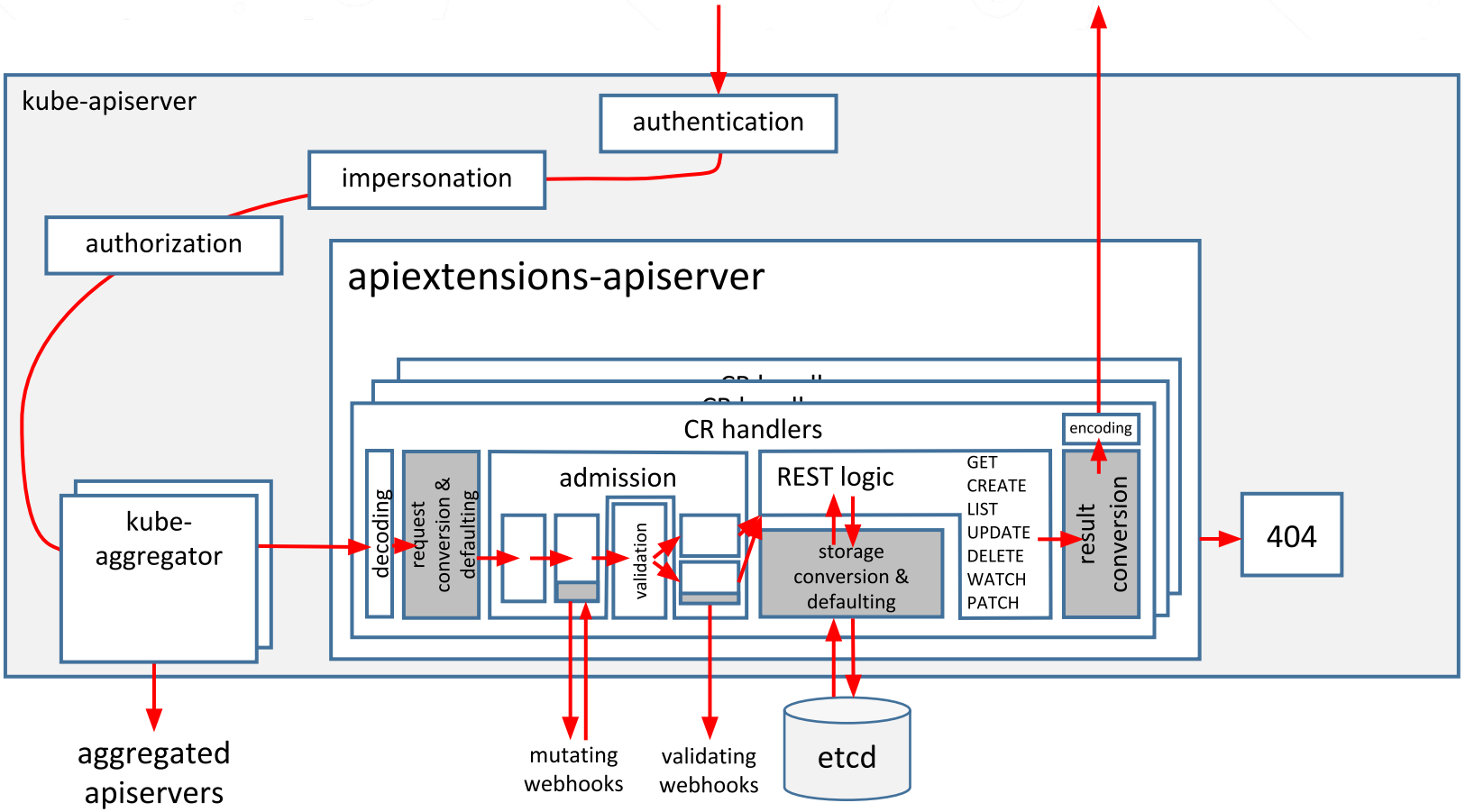
& Conversion

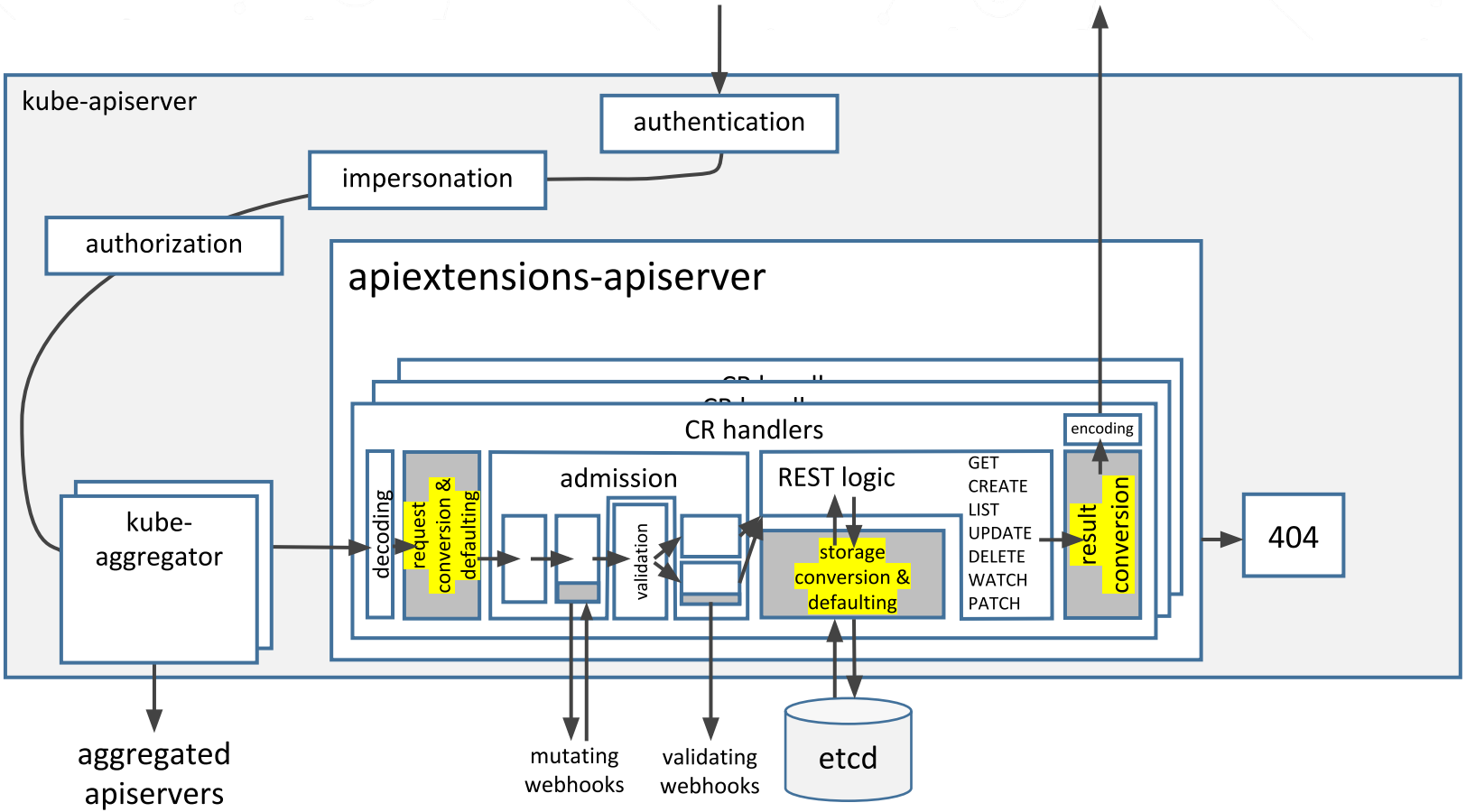
deep dive

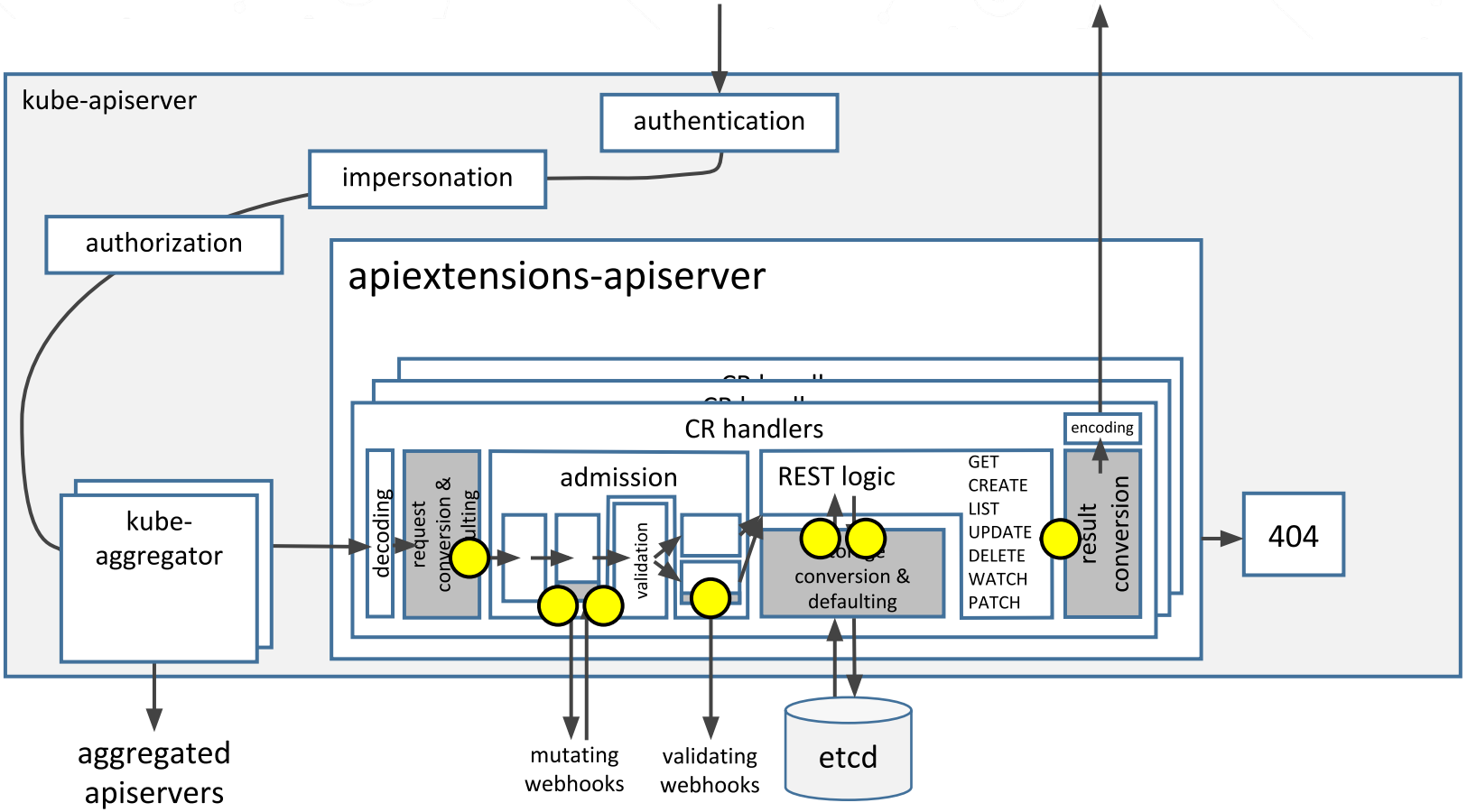
CR Conversion

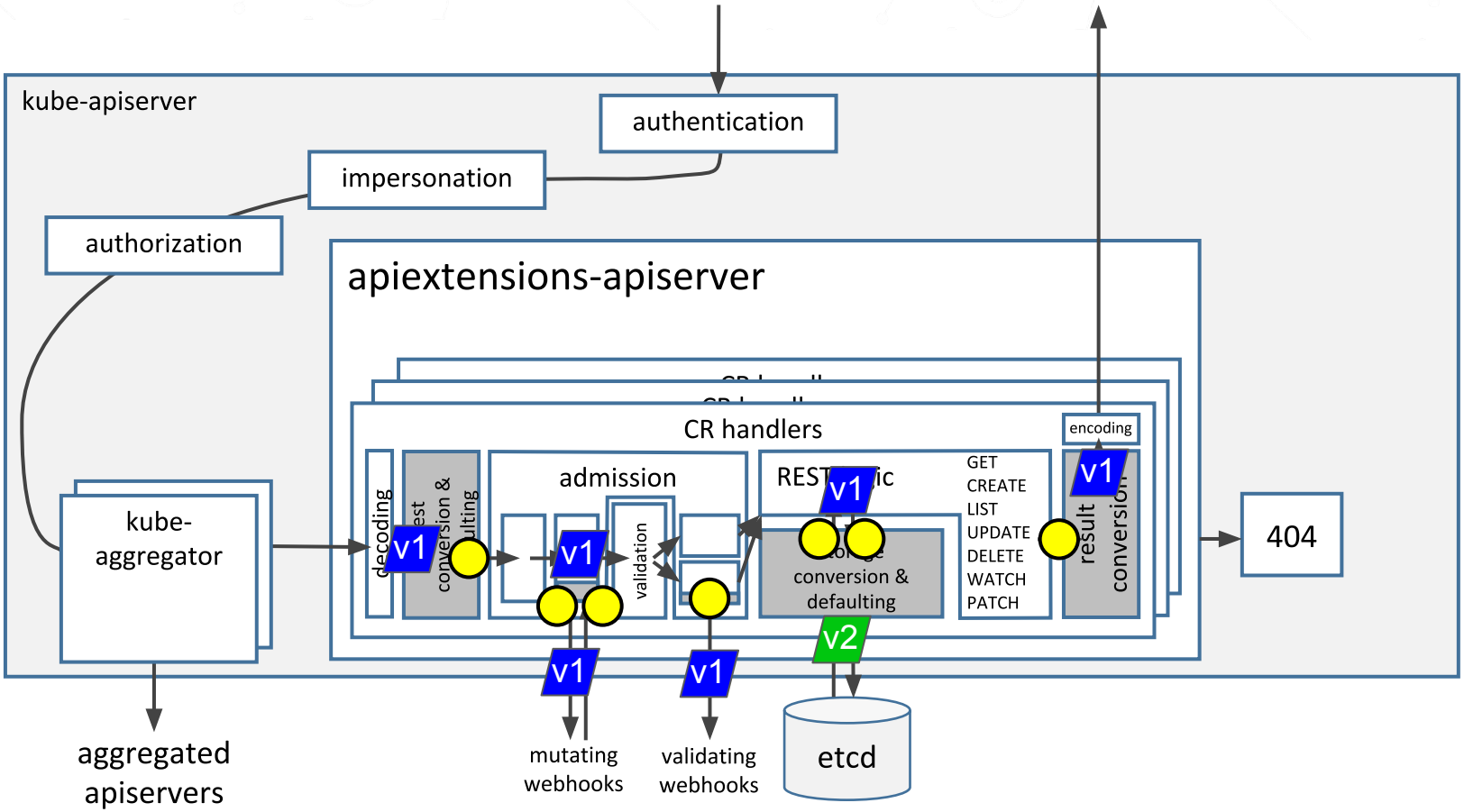
- Since 1.13 as alpha: Webhook Conversion
  - similar to admission webhooks, via **apiextensions.k8s.io/v1beta.ConversionReview**

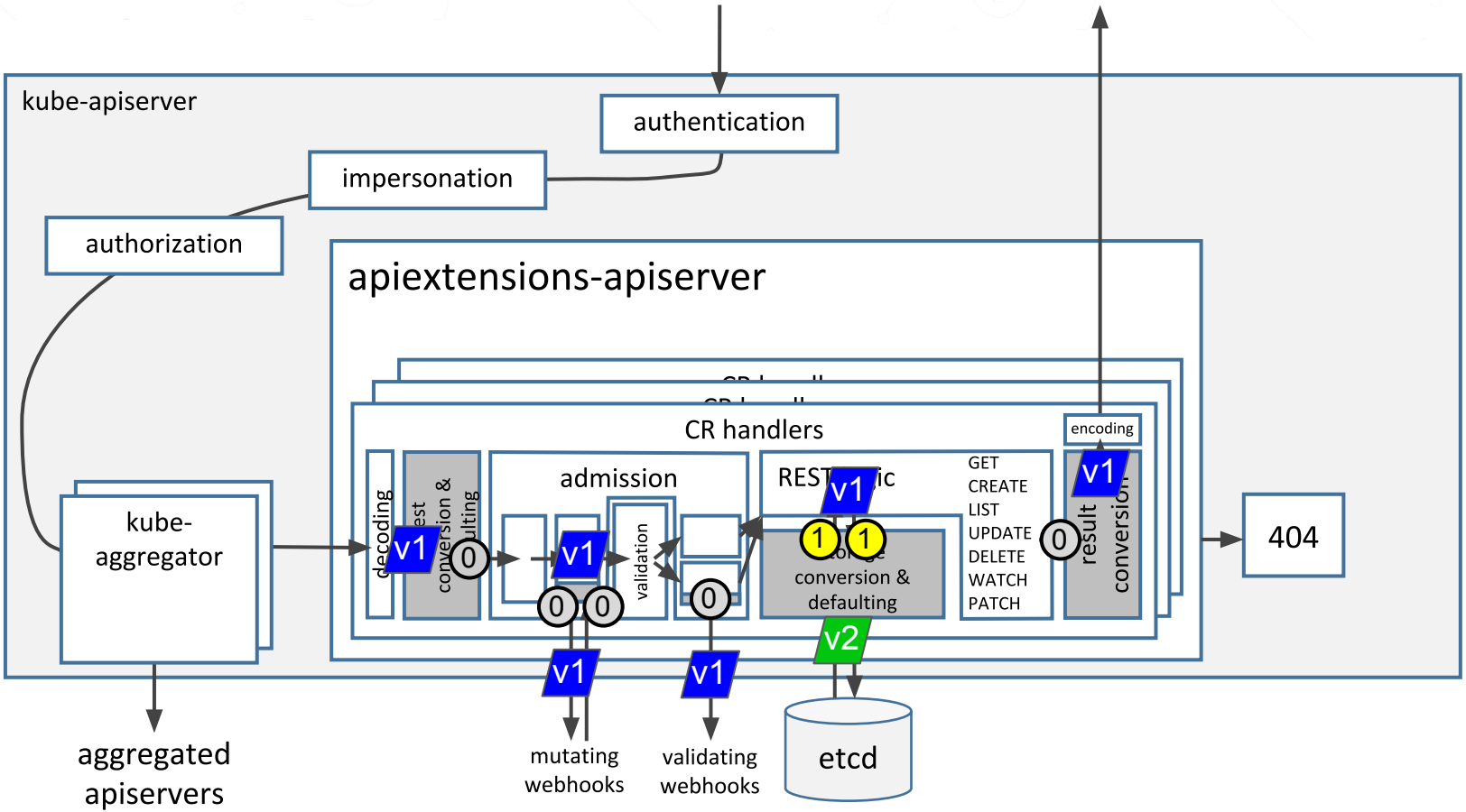


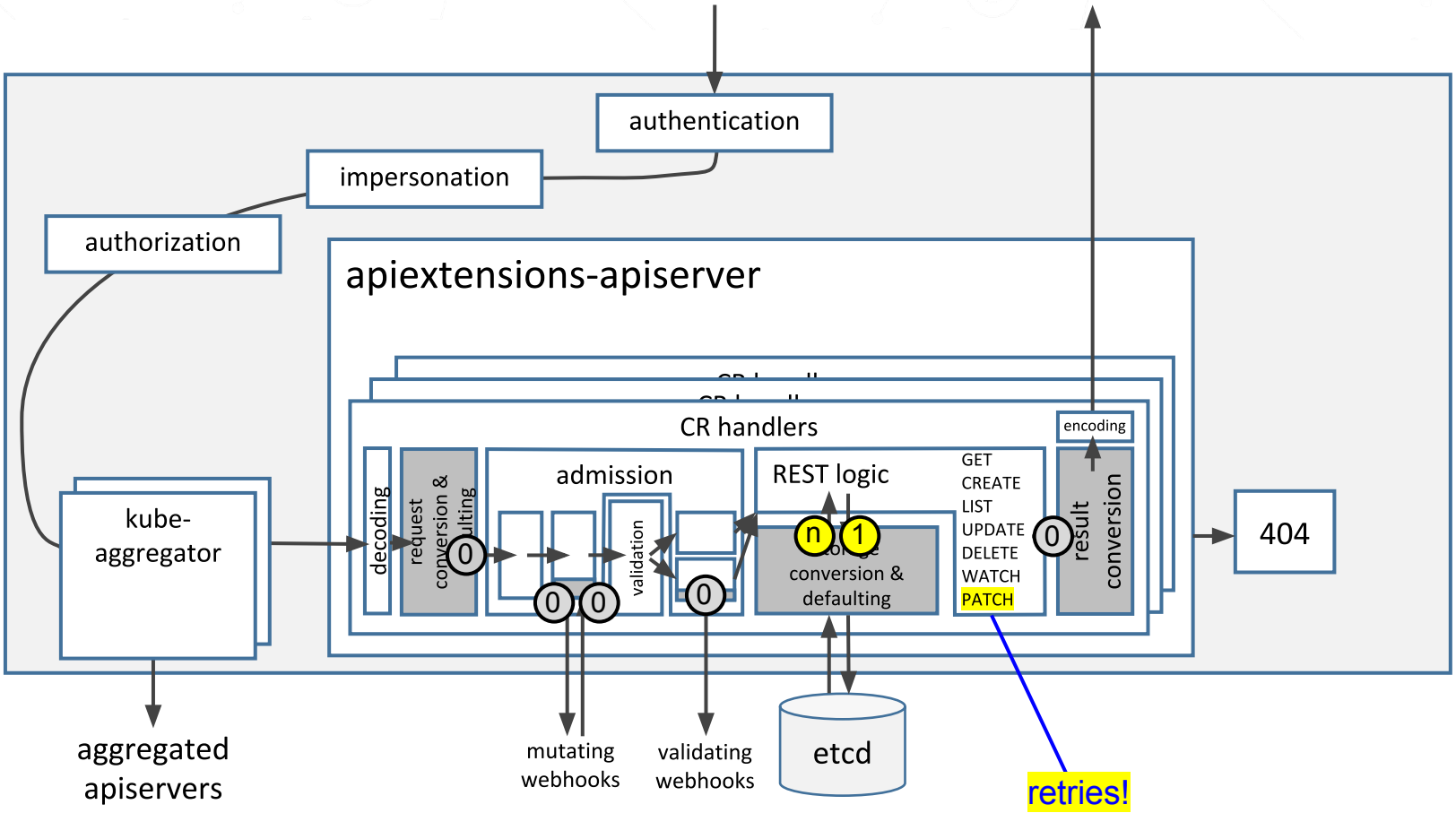

kube-apiserver
authentication
impersonation
authorization
apiextensions-apiserver
CR handlers
encoding
decoding
request conversion & defaulting
admission
validation
REST logic
storage conversion & defaulting
GET
CREATE
LIST
UPDATE
DELETE
WATCH
PATCH
result conversion
404
kube-aggregator
aggregated apiservers
mutating webhooks
validating webhooks
etcd

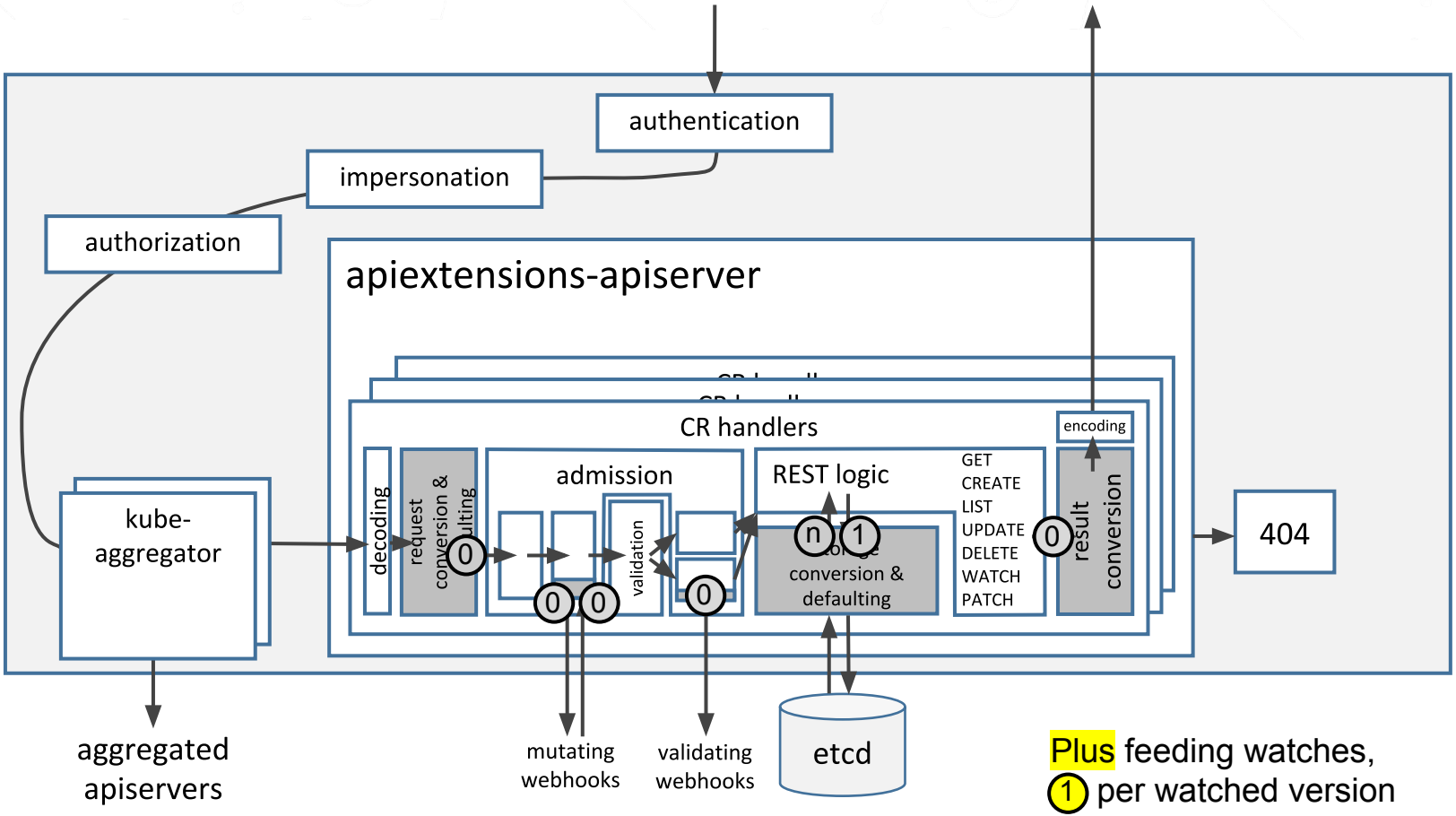

kube-apiserver
authentication
impersonation
authorization
apiextensions-apiserver
CR handlers
encoding
admission
REST logic
GET
CREATE
LIST
UPDATE
DELETE
WATCH
PATCH
decoding
request
conversion &
defaulting
validation
storage
conversion &
defaulting
result
conversion
kube-
aggregator
404
aggregated
apiservers
mutating
webhooks
validating
webhooks
etcd

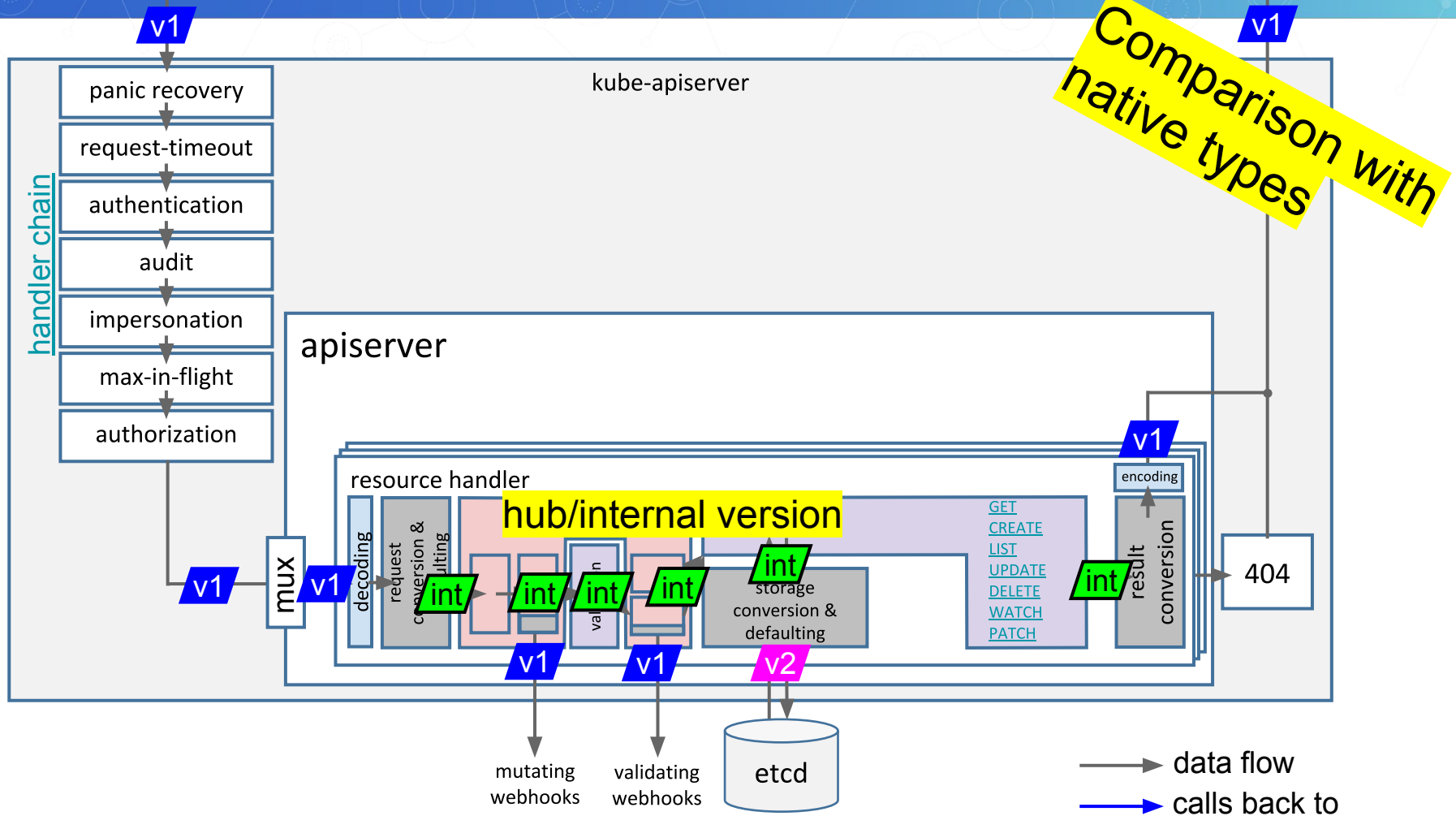

kube-apiserver
authentication
impersonation
authorization
apiextensions-apiserver
CR handlers
encoding
decoding
request conversion & defaulting
admission
validation
REST logic
conversion & defaulting
GET
CREATE
LIST
UPDATE
DELETE
WATCH
PATCH
result conversion
404
kube-aggregator
aggregated apiservers
mutating webhooks
validating webhooks
etcd


kube-apiserver
authentication
impersonation
authorization
apiextensions-apiserver
CR handlers
encoding
admission
validation
GET
CREATE
LIST
UPDATE
DELETE
WATCH
PATCH
conversion &
defaulting
result
conversion
v1
v2
kube-
aggregator
404
aggregated
apiservers
mutating
webhooks
validating
webhooks
etcd


kube-apiserver
authentication
impersonation
authorization
apiextensions-apiserver
CR handlers
encoding
admission
validation
conversion &
defaulting
GET
CREATE
LIST
UPDATE
DELETE
WATCH
PATCH
v1
v2
0
1
kube-
aggregator
404
aggregated
apiservers
mutating
webhooks
validating
webhooks
etcd


authentication
impersonation
authorization
apiextensions-apiserver
CR handlers
encoding
kube-
aggregator
decoding
request
admission
REST logic
GET
CREATE
LIST
UPDATE
DELETE
WATCH
PATCH
result
conversion
404
validation
conversion &
defaulting
aggregated
apiservers
mutating
webhooks
validating
webhooks
etcd
retries!


authentication
impersonation
authorization
apiextensions-apiserver
CR handlers
encoding
decoding
request conversion & defaulting
admission
validation
REST logic
conversion & defaulting
GET
CREATE
LIST
UPDATE
DELETE
WATCH
PATCH
result conversion
404
kube-aggregator
aggregated apiservers
mutating webhooks
validating webhooks
etcd
Plus feeding watches,
1 per watched version

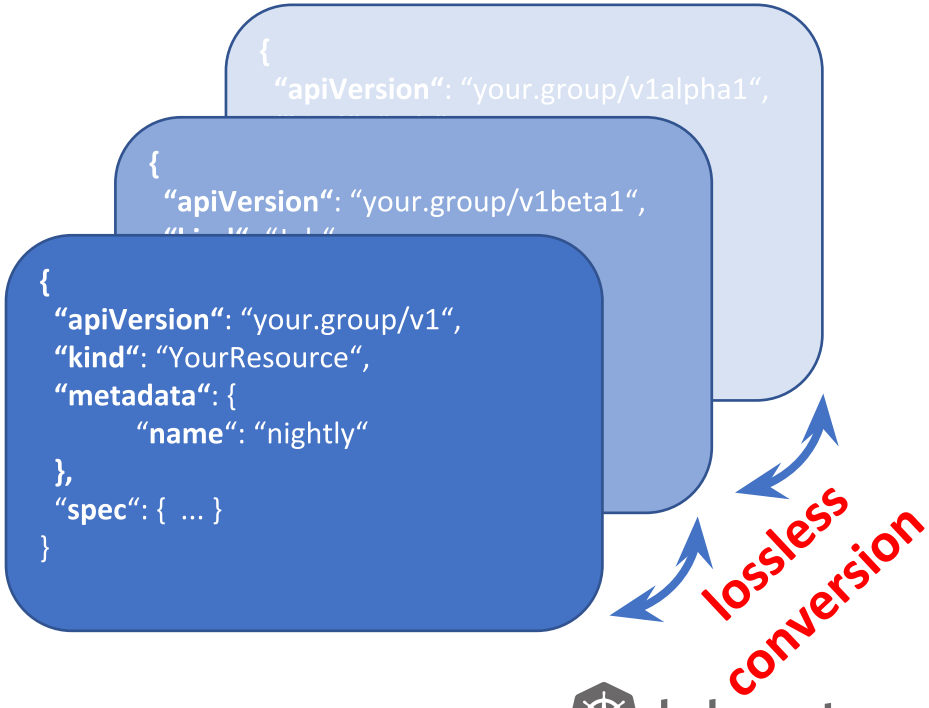

Comparison with native types
kube-apiserver
handler chain
panic recovery
request-timeout
authentication
audit
impersonation
max-in-flight
authorization
apiserver
resource handler
hub/internal version
mux
decoding
request conversion & defaulting
storage conversion & defaulting
GET
CREATE
LIST
UPDATE
DELETE
WATCH
PATCH
encoding
result conversion
404
v1
int
v2
mutating webhooks
validating webhooks
etcd
data flow
calls back to

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



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

Roundtrip testing – how we do that in upstream

use the **gofuzz** library + k8s.io/apimachinery/pkg/api/apitesting/roundtrip:

```
func Funcs(codecs runtimeserializer.CodecFactory) []interface{} {
    return []interface{}{
        func(e *audit.Event, c fuzz.Continue) {
            c.FuzzNoCustom(e)
            doAuditEventDefaulting()
            enforceAnyOtherTypeInvariant()
        }
    }
}

func TestRoundTrip(t *testing.T) {
    roundtrip.RoundTripTestForAPIGroup(t, Install, fuzzer.Funcs)
}
```

Example: pkg/apis/audit/fuzzer/fuzzer.go & pkg/apis/audit/install/roundtrip_test.go:

CRD – Outlook for 2019 and beyond

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

The other big 2019 theme:

Server-side Apply

Serverside Apply

- Plumbing: mostly done
- Pivot!
  - From x = a + (b - c) to field “managers”
- Low level operations: mostly done
- API: in progress!
  - Looks to be a big improvement over the old last-applied annotation
    - Shorter
    - Omits values
- Alpha in 1.14
- Next steps
  - Perfect place to check general schema concerns (e.g, add discriminated unions)

API Machinery Q & A