
KubeCon
CloudNativeCon
North America 2023


KubeCon | CloudNativeCon

North America 2023

The Future of Kubernetes Auth and Policy Config: Common Expression Language

Mo Khan, Microsoft & Jordan Liggitt, Google

Shout outs!

- @aramase KMS v2, structured authentication config
- @ahmedtd Cluster trust bundle
- @munnerz Bound SA token improvements
- @nilekhc KMS v2
- @palnabarun Structured authorization config
- @ritazh KMS v2, structured authorization config
- @yt2985 Reduction of secret based SA tokens

What is Common Expression Language (CEL)?

CEL is a non-Turing-complete language
designed for simplicity, speed, safety, and portability.

```
// Check whether all resource names in a list match a given filter.

claims.email_verified && resources.all(r, r.startsWith(claims.email))
```

Why use CEL?

- Straightforward syntax similar to expressions in Go/C/C++/Java
- Lightweight (in comparison to something like JavaScript or Rego)
- Ability to do deep type-checking integrations at validation time
- Integrator can define custom environments, variables, functions available
- Can calculate and bound execution time at compile time and runtime

CEL usage outside SIG-Auth

- KEP-2876: CRD Validation Expression Language
  Beta in v1.25, GA in v1.29
- KEP-3488: CEL for Admission Control
  Beta in v1.28
- KEP-3716: Admission Webhook Match Conditions
  Beta in v1.28

KEP-3331: Structured Authentication Config

Alpha in v1.29

Questions:

- Can we limit the scenarios that require an authentication webhook?
- What user stories can we support if we standardize around JWTs (JSON Web Tokens)?

KEP-3331: Structured Authentication Config

With this JWT payload:

```
{
 "sub": "119abc",
 "aud": "kubernetes",
 "username": "jane_doe",
 "roles": "admin,user",
 ...
}
```

And this CEL-based authentication config:

```
claimMappings:
 username:
   expression: 'claims.username + ":external-user"'
 groups:
   expression: 'claims.roles.split(",") + [ "idp1" ]'
 uid:
   claim: 'sub'
 extra:
 - key: "client_name"
   value: 'claims.aud'
```

We get the following user info:

```
username: "jane_doe:external-user"
uid: "119abc"
groups: ["admin", "user", "idp1"]
extra:
 client_name: ["kubernetes"]
```

We can validate the user never has a system username:

```
userInfoValidationRules:
- rule: "!userInfo.username.startsWith('system:')"
  message: "username cannot use system: prefix"
```

KEP-3221: Structured Authorization Config

Alpha in v1.29

New capabilities:

- Run more than one webhook
- Customize timeout
- Customize failure policy

Goals when running multiple webhooks:

- Limit the blast radius from an unavailable webhook
- Reduce performance impact of webhooks
  that only care about specific types of requests
- Make it safer to host authorization webhooks on the cluster

KEP-3221: Structured Authorization Config

We can limit which requests an authorization webook needs to intercept:

```yaml
matchConditions:
# only send resource requests to the webhook, i.e. ignore requests like /healthz
- expression: has(request.resourceAttributes)

# only intercept requests to kube-system
- expression: request.resourceAttributes.namespace == 'kube-system'

# don't intercept requests from kube-system service accounts
- expression: "!('system:serviceaccounts:kube-system' in request.groups)"
```

Questions?

https://git.k8s.io/community/sig-auth

Bi-weekly meetings (Wednesday at 11am Pacific Time)
Mailing list: kubernetes-sig-auth@googlegroups.com
Kubernetes Slack: #sig-auth

Please scan the QR Code above
to leave feedback on this session