Kubernetes Extension Points

With Deep Dive into Custom Resource Definitions

Jungho Kim, Architech

December 11, 2018

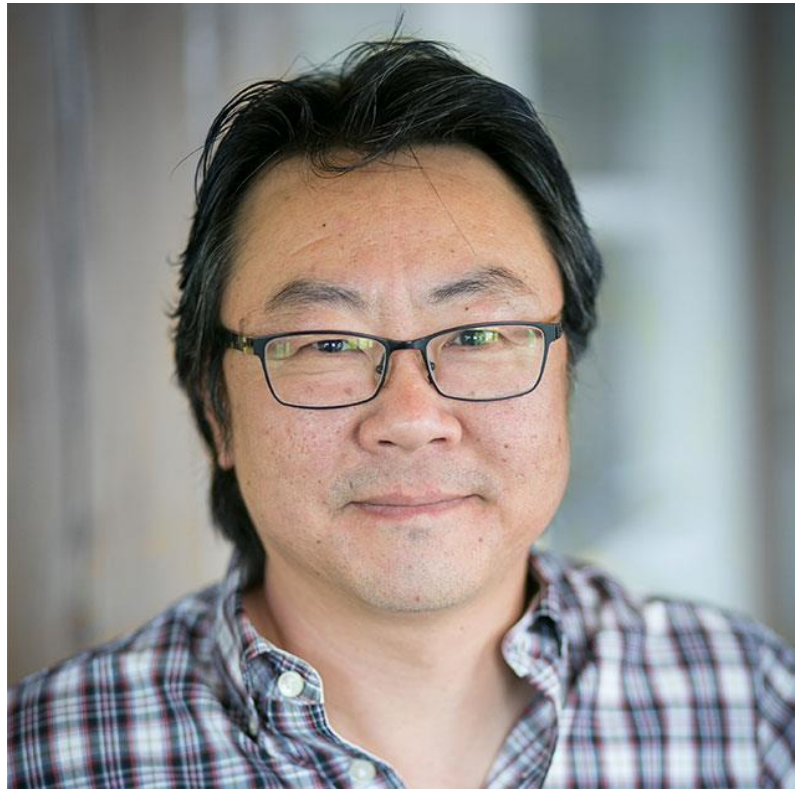
ARCHITECH

In ~~90~~ 60 minutes…

- Quick overview of Kubernetes extensions points then focus on Custom Resource Definitions (CRDs) and Custom Controllers
- Deep dive into CRDs, the basis for “Operators”
  - Use-cases from the community
    - Prometheus Operator
  - Roll your own example to get the concept
  - Using Kubebuilder
  - Using Operator SDK

https://github.com/jungho/k8s-crds

https://github.com/jungho/k8s-bootcamp-ms



Twitter: @jungho_kim

LinkedIn: @junghokim

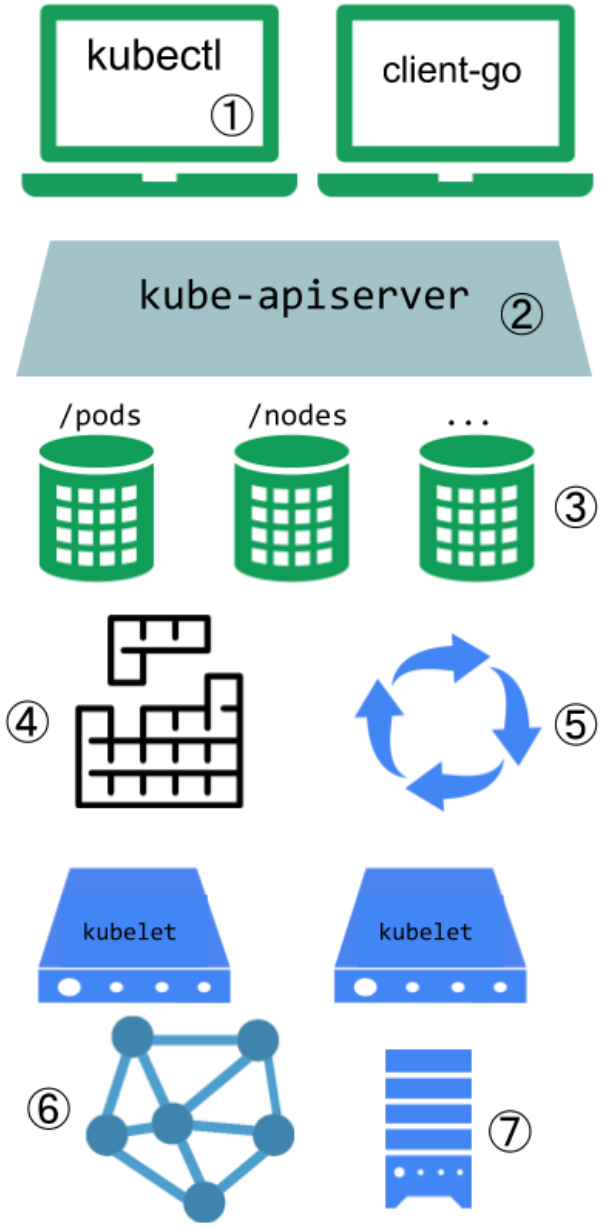
GitHub: @jungho

Desired Outcome

- Learn how extensible K8S can be
- Understand better the inner workings of Kubernetes
- Spark your imagination to leverage Kubernetes in more powerful ways
- Start your journey to becoming Kubernetes ninjas!

Kubernetes Extension Points

- K8S is incredibly extensible
- Designed with well-defined extension points to extend/customize the platform
  1. Plugins for kubectl to add additional kubectl commands (you can't overwrite existing commands)
  2. Authn/Authr, Webhooks, Admission Controllers, Dynamic Admission Control, API Aggregation
  3. **Custom Resource Definitions**
  4. Scheduler extensions. You can replace the default scheduler, use multiple schedulers, extend scheduler behaviours via webhooks
  5. **Controllers that reconciles current-state to desired state.**
  6. Network Plugins e.g. Container Network Interface (CNI) plugins such as Celium, Calico, Azure CNI
  7. Storage Plugins e.g. Flex Volumes



https://bit.ly/2r6qKTp

Custom Resource Definitions

- Means to define new API resources to model your domain
- Can leverage kubectl, helm to work with your CRDs

```yaml
apiVersion: apiextensions.k8s.io/v1beta1
kind: CustomResourceDefinition
metadata:
  #the name must be the plural form + api group
  name: websites.extensions.example.com
spec:
  # Can be namespaced or cluster scope
  scope: Namespaced
  #All resources have a version and are part of an api group
  group: extensions.example.com
  version: v1
  # The names of the resource when using kubectl
  names:
    kind: Website
    singular: website
    plural: websites
    shortNames: ['ws']
```

The CRD

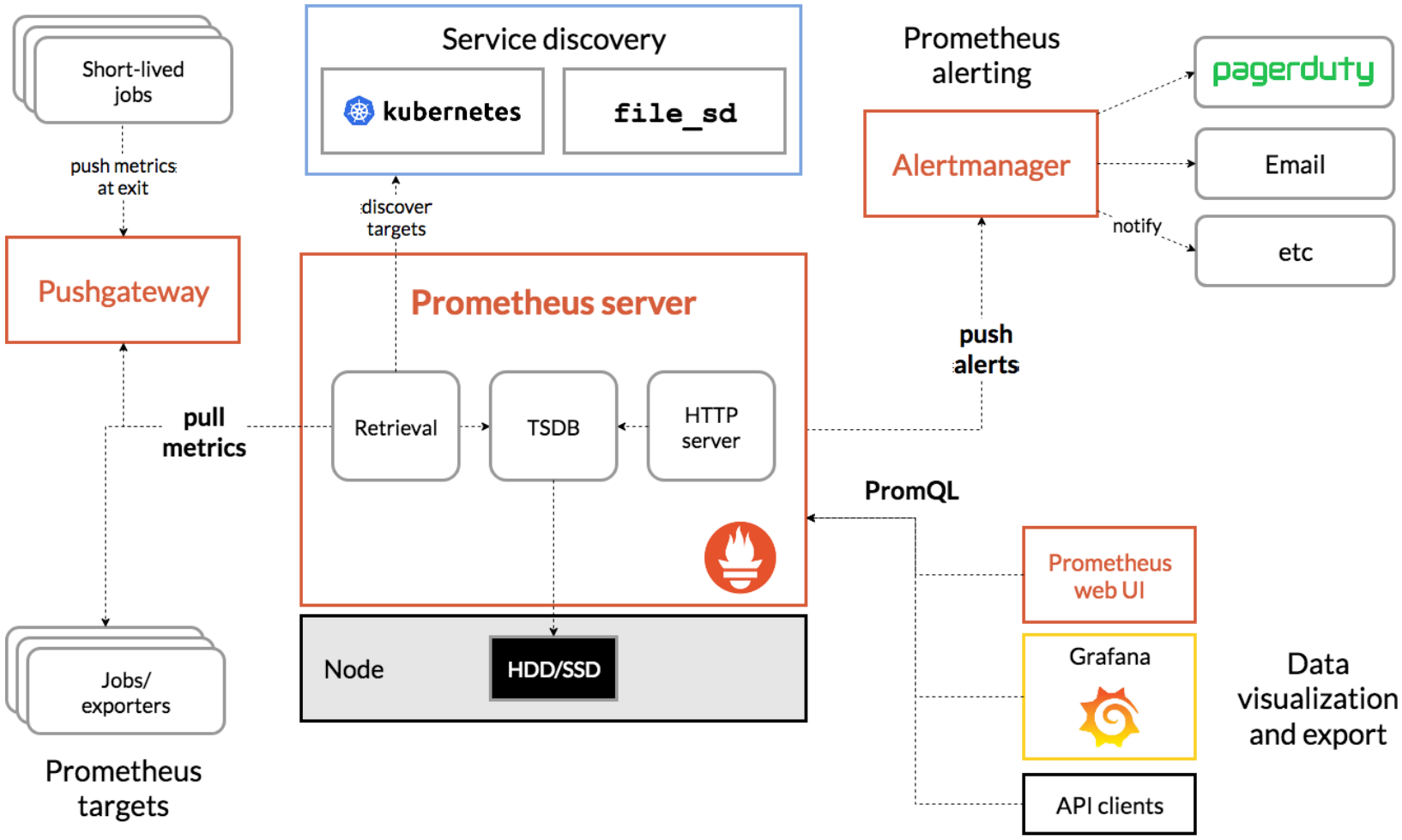
```yaml
apiVersion: extensions.example.com/v1
kind: Website
metadata:
  name: kubia
spec:
  gitRepo: https://github.com/luksa/kubia-website-example.git
```

Instance of CRD

Custom Contollers

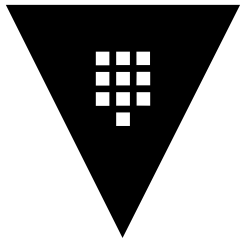
- CRDs alone don’t do anything
- Something needs to consume instances of a CRD and take action, that is the **Controller**
- In simple terms, a controller is responsible for:
  - Watching for specific resource types e.g. Website
  - Reconciling the desired state, and doing so continuously (this is what makes K8S so resilient!!)

Prometheus



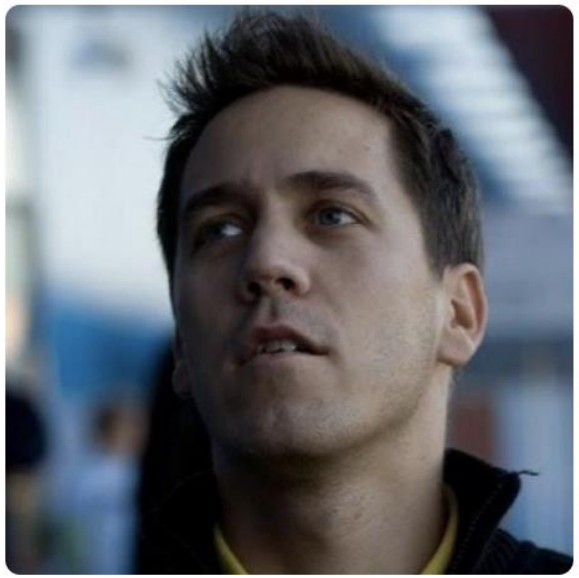
https://prometheus.io/docs/introduction/overview/

Operators



https://github.com/operator-framework/awesome-operators

First CRD Example…. Thank You Lukša!



Marko Lukša
luksa

Software engineer at Red Hat.
Author of Kubernetes in Action.

Overview | Repositories 105 | Stars 36 | Fol

Popular repositories

kubernetes-in-action

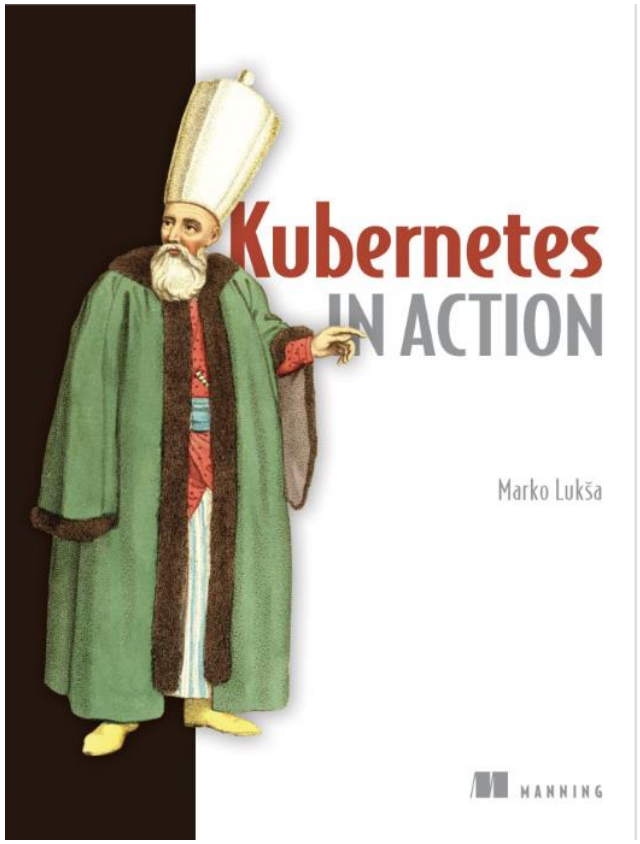
Code from the Kubernetes in Action book

JavaScript ★ 146 106

k8s-website-controller

A barely working example of a Kubernetes controller

Go ★ 11 2



https://github.com/luksa/k8s-website-controller

Kubebuilder

- Part of the apimachinery SIG (Special Interest Group)
  - https://github.com/kubernetes-sigs
- GA version v1.0.5, so stable
- Excellent documentation at https://book.kubebuilder.io/
- Provides scaffolding to quickly get started including:
  - Generate CRD, CRD instance, webhooks
  - Golang code for Controller, Manager, CRD Types, Reconciler, tests
  - Deployment manifests
  - RBAC manifests
  - Makefile to build, test, deploy your CRD and custom controller
  - Annotations to generate CRDs OpenAPI v3 schema validation to generate RBAC Roles

Kubebuilder Workflow

```
#Must run within $GOPATH
kubebuilder init --domain architech.ca --owner "Jungho Kim”

#You will be asked the following, answer 'y’.
Run `dep ensure` to fetch dependencies (Recommended) [y/n]?

#Now create a new API resource and Controller
#answer 'y' to both Resource and Controller
kubebuilder create api --group example --version v1beta1 --kind Website
```

Kubebuilder Workflow

```
#Modify the generated yaml and golang code then to run locally
make test
make install
make run
kubectl create -f config/samples #Deploy our CRD instance

#To deploy the controller to K8S
export IMG=architechbootcamp/website-kubebuilder-controller:1.0.0
Make docker-build
Make docker-deploy
kubectl create -f config/samples
```

Operator SDK

- Created by the CoreOS team that coined ”Operator”
- Still “alpha” but used for many published operators
- Supports both golang and Ansible implementations
- Leverages controller-runtime package which is a sub-project of kubebuilder
- Excellent documentation here https://bit.ly/2U8RXCt
- Provides scaffolding to quickly get started including:
  - Generate CRD, CRD instance
  - Golang code for Controller, Manager, CRD Types, Reconciler
  - Deployment manifests
  - RBAC manifests
- Does not provide a nice Makefile to help with workflow so you need to write some scripts

Operator SDK Workflow

```
#Create new project, will generate website-operator-sdk directory
operator-sdk new website-operator-sdk --skip-git-init

#Create API, will generate types and deployment yaml files
operator-sdk add api
    --api-version=example.architech.ca/v1beta1
    --kind=Website

#Add a controller
operator-sdk add controller
    --api-version=example.architech.ca/v1beta1
    --kind=Website
```

Operator SDK Workflow

```
#Modify the generated go code and yaml files.
#Execute within the website-operator-sdk directory
operator-sdk build $image

#Create API, will generate types and deployment yaml files
kubectl create -f ./deploy/crds/example_v1beta1_website_crd.yaml

#To run the controller locally
export OPERATOR_NAME="$operatorName"
operator-sdk up local --namespace "$default"
```

Which One?

- Kubebuilder for workflow and general stability
- It also supports creating custom Webhooks and Admission Controllers
- Operator-SDK if you want to implement your Controllers using Ansible
- If you cannot use Golang, then there is also MetaController which allows you to use the programming language of your choice. https://metacontroller.app/

We use open source, cloud-native technologies to modernize systems.

ARCHITECH

Integrated Development Studio

Strategy

Design

Engineering

Analytics

Support

Digital Transformation

Product Innovation

Legacy Modernization

Team Enablement

Modern Product Development

Lean Startup

Design Thinking

Agile Engineering

14+ years

300+ projects

100+ people

40+ systems modernized

Digital Transformation Journeys

ARCHITECH

"We're trying to become an organization that's constantly rethinking the problems we're solving in the market today and looking to bring new value to clients in ways that they haven't thought of yet. Architech is instrumental to us throughout this journey.

Hugh Cumming, Chief Technology Officer, Finastra ($2B+ fintech)

Open Source Experts in the Cloud

- Reduced Costs
- Enhanced Supportability
- Massive Global Adoption
- Avoid Vendor Lock-In
- Availability, Resiliency, Security
- Achieve True Business Agility

ARCHITECH

70 Bond St., Suite 400
Toronto, ON, Canada
M5B 1XB

P: 416.607.5618
F: 416.352.1768
www.architech.ca