From Zero to Production with Kubernetes

Daniel Lopez Ridruejo & Angus Lees

bitnami

About Us

Gus Lees, Principal Engineer, Bitnami

Past: Google, Openstack, Debian

Daniel Lopez, Co-founder, Bitnami

Past: Devicescape, Covalent Tecnologies, Apache Software Foundation

Bitnami and Kubernetes

We package software for any platform

PACKAGING

- Founding member of Helm project
- Founding member of Ksonnet project
- Maintain many of the stable helm charts
- Training and "how-to" guides

AUTHORING

- Stacksmith
- Kubeapps
- Kubeless
- Sealed-secrets
- kubewatch
- helm-crd

…

USING

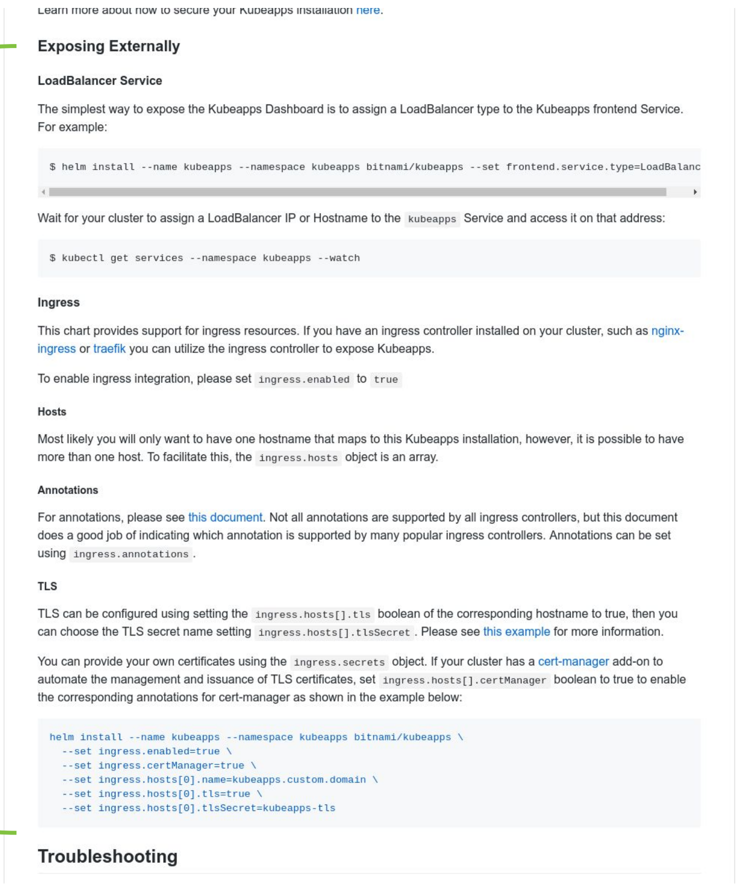
- Multiple clusters
- Multiple providers
- Multiple years

bitnami

Recurring themes

Kubeapps

3rd party Ingress / TLS configuration



Learn more about how to secure your Kubeapps installation here.

Exposing Externally

LoadBalancer Service

The simplest way to expose the Kubeapps Dashboard is to assign a LoadBalancer type to the Kubeapps frontend Service. For example:

```
$ helm install --name kubeapps --namespace kubeapps bitnami/kubeapps --set frontend.service.type=LoadBalanc
```

Wait for your cluster to assign a LoadBalancer IP or Hostname to the `kubeapps` Service and access it on that address:

```
$ kubectl get services --namespace kubeapps --watch
```

Ingress

This chart provides support for ingress resources. If you have an ingress controller installed on your cluster, such as nginx-ingress or traefik you can utilize the ingress controller to expose Kubeapps.

To enable ingress integration, please set `ingress.enabled` to `true`

Hosts

Most likely you will only want to have one hostname that maps to this Kubeapps installation, however, it is possible to have more than one host. To facilitate this, the `ingress.hosts` object is an array.

Annotations

For annotations, please see this document. Not all annotations are supported by all ingress controllers, but this document does a good job of indicating which annotation is supported by many popular ingress controllers. Annotations can be set using `ingress.annotations`.

TLS

TLS can be configured using setting the `ingress.hosts[].tls` boolean of the corresponding hostname to true, then you can choose the TLS secret name setting `ingress.hosts[].tlsSecret`. Please see this example for more information.

You can provide your own certificates using the `ingress.secrets` object. If your cluster has a cert-manager add-on to automate the management and issuance of TLS certificates, set `ingress.hosts[].certManager` boolean to true to enable the corresponding annotations for cert-manager as shown in the example below:

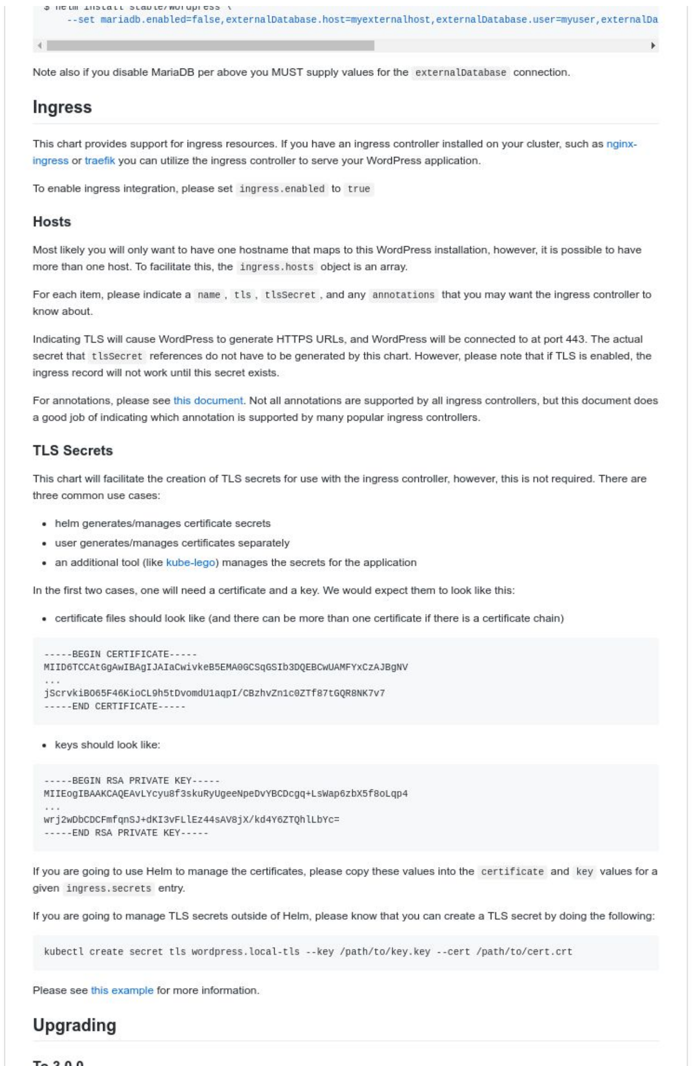
```
helm install --name kubeapps --namespace kubeapps bitnami/kubeapps \
  --set ingress.enabled=true \
  --set ingress.certManager=true \
  --set ingress.hosts[0].name=kubeapps.custom.domain \
  --set ingress.hosts[0].tls=true \
  --set ingress.hosts[0].tlsSecret=kubeapps-tls
```

Troubleshooting

Recurring themes

Wordpress helm chart

3rd party Ingress / TLS configuration



```
$ helm install stable/wordpress \
    --set mariadb.enabled=false,externalDatabase.host=myexternalhost,externalDatabase.user=myuser,externalDa
```

Note also if you disable MariaDB per above you MUST supply values for the `externalDatabase` connection.

Ingress

This chart provides support for ingress resources. If you have an ingress controller installed on your cluster, such as nginx-ingress or traefik you can utilize the ingress controller to serve your WordPress application.

To enable ingress integration, please set `ingress.enabled` to `true`

Hosts

Most likely you will only want to have one hostname that maps to this WordPress installation, however, it is possible to have more than one host. To facilitate this, the `ingress.hosts` object is an array.

For each item, please indicate a `name`, `tls`, `tlsSecret`, and any `annotations` that you may want the ingress controller to know about.

Indicating TLS will cause WordPress to generate HTTPS URLs, and WordPress will be connected to at port 443. The actual secret that `tlsSecret` references do not have to be generated by this chart. However, please note that if TLS is enabled, the ingress record will not work until this secret exists.

For annotations, please see this document. Not all annotations are supported by all ingress controllers, but this document does a good job of indicating which annotation is supported by many popular ingress controllers.

TLS Secrets

This chart will facilitate the creation of TLS secrets for use with the ingress controller, however, this is not required. There are three common use cases:

- helm generates/manages certificate secrets
- user generates/manages certificates separately
- an additional tool (like kube-lego) manages the secrets for the application

In the first two cases, one will need a certificate and a key. We would expect them to look like this:

- certificate files should look like (and there can be more than one certificate if there is a certificate chain)

```
-----BEGIN CERTIFICATE-----
MIID6TCCAtGgAwIBAgIJAIaCwivkeB5EMA0GCSqGSIb3DQEBCwUAMFYxCzAJBgNV
...
jScrvkiBO65F46KioCL9h5tDvomdU1aqpI/CBzhvZn1c0ZTf87tGQR8NK7v7
-----END CERTIFICATE-----
```

- keys should look like:

```
-----BEGIN RSA PRIVATE KEY-----
MIIEogIBAAKCAQEAvLYcyu8f3skuRyUgeeNpeDvYBCDcgq+LsWap6zbX5f8oLqp4
...
wrj2wDbCDCFmfqnSJ+dKI3vFLlEz44sAV8jX/kd4Y6ZTQhlLbYc=
-----END RSA PRIVATE KEY-----
```

If you are going to use Helm to manage the certificates, please copy these values into the `certificate` and `key` values for a given `ingress.secrets` entry.

If you are going to manage TLS secrets outside of Helm, please know that you can create a TLS secret by doing the following:

```
kubectl create secret tls wordpress.local-tls --key /path/to/key.key --cert /path/to/cert.crt
```

Please see this example for more information.

Upgrading

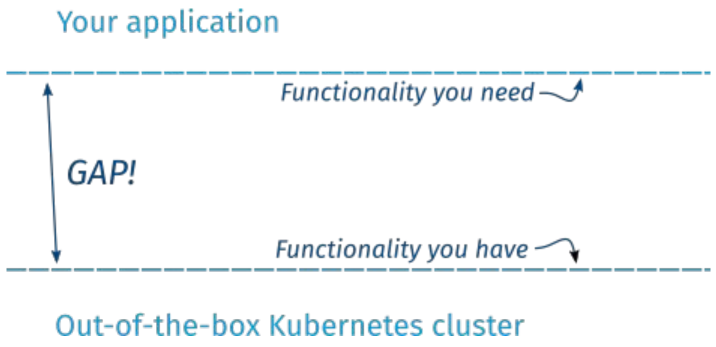
bitnami

BKPR: Bitnami Kubernetes Production Runtime

What is it?

Problem:

- Applications need “more” than bare k8s
- Unclear “next steps” after cluster install
- Ad-hoc solutions all different



Solution:

- **Standard set of manifests** for “expected” k8s infrastructure
- **Opinionated** to remove unnecessary deviation
- … but **customisable**, because cluster admins still need full control

bitnami

[DEMO]

BKPR 1.0

Monitoring/alerting stack:

- prometheus
- alertmanager
- node-exporter
- kube-state-metrics

Logging stack:

- elasticsearch
- fluentd
- kibana

Ingress stack:

- nginx-ingress
- cert-manager (letsencrypt)
- external-dns
- oauth2-proxy

Conservative selection

Will evolve to follow the broader ecosystem

Configured for production

- persistent volumes
- N+1 high availability
- restrictive RBAC
- etc

BKPR

Details

Three user-visible parts of BKPR:

- **Platform-neutral specification** of exposed annotations, features, etc.
- **Kubernetes manifests** for each target platform
- **Installer tool** (`kubeprod`)

Each BKPR release supports two consecutive versions of Kubernetes:

- Kubernetes and BKPR can be upgraded separately
- BKPR 1.0 release targets Kubernetes 1.9 & 1.10 on AKS/GKE

Apache-2 license. https://github.com/bitnami/kube-prod-runtime

BKPR: Usage

(Simple version)

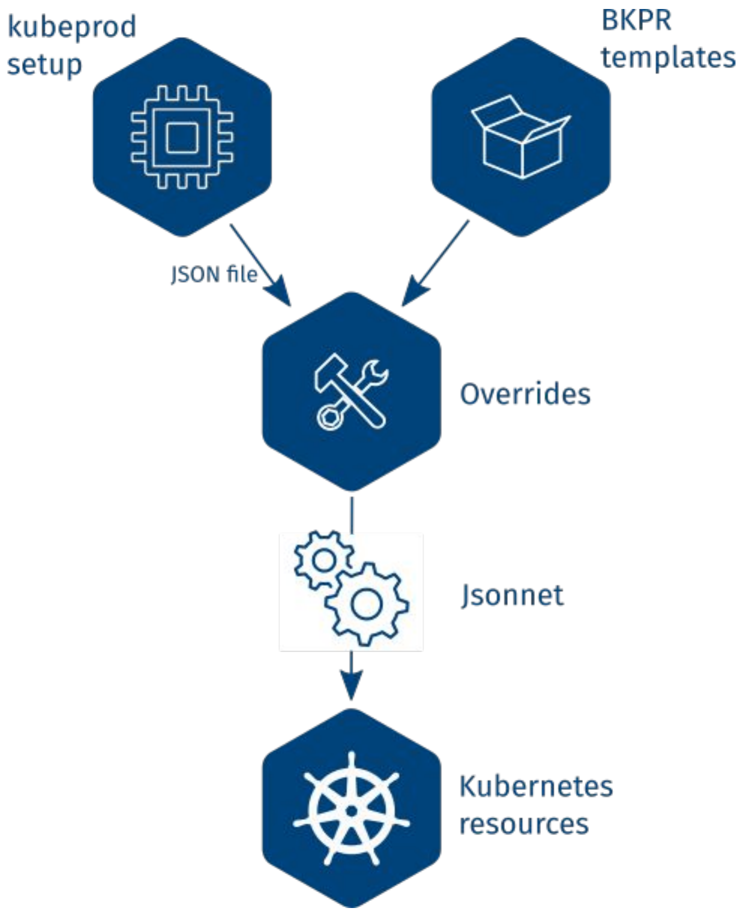
1. Create a fresh/empty Kubernetes cluster
2. Run: `kubeprod install {aks|gke} --dns-zone=example.com <other flags>`
3. Use https://kibana.example.com, https://prometheus.example.com, etc.
4. Upgrades: goto (2)

BKPR: Usage

(Simple version)

1. Create a fresh/empty Kubernetes cluster
2. Run: `kubeprod install {aks|gke} --dns-zone=example.com <other flags>`
    a. configure underlying cloud
    b. write a JSON file with cluster-specific parameters
    c. generate Kubernetes manifest
    d. push manifests to cluster
3. Use https://kibana.example.com, https://prometheus.example.com, etc.
4. Upgrades: goto (2)
    a. Re-uses local JSON file from 2.b.

bitnami


kubeprod setup
BKPR templates
JSON file
Overrides
Jsonnet
Kubernetes resources

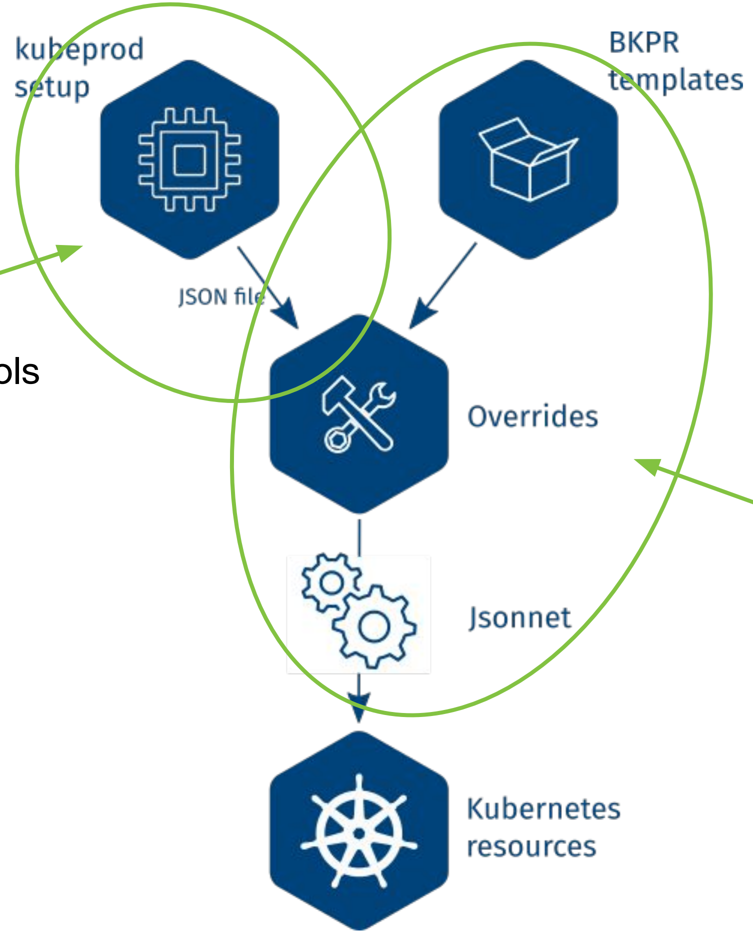
bitnami

BKPR Templates

Not just another pile of YAML

bitnami

kubeprod install
- JSON
- Replace with local tools (eg: terraform)



Manifest templates
- Jsonnet
- Replace with kubecfg

BKPR: Usage

(Advanced version)

1. Do the non-Kubernetes setup manually
    a. Hand-generate the same JSON file
2. Create a **jsonnet overlay file** with any local customisations
3. Push manifests using `kubecfg`
4. Use https://kibana.example.com, https://prometheus.example.com, etc.

```
local bkpr = import
  "https://github.com/bitnami/kube-prod-runtime/raw/v1.0.0/manifests/platforms/gke.jsonnet";

bkpr {
  config:: import "local-parameters.json",
  // Place your overrides here
}
```

bitnami

BKPR: Usage

Overrides

Full control of configuration using jsonnet “overlay”

Social contract:

- We support/test the base manifests
- You support/test your overrides

No *customization cliff*:

- Simple customization is simple
- Complex customization is possible
- Effort proportional to deviation
- Overlay is clearly separate → easy to “rebase” onto new releases

BKPR: Usage

Example override

```
local bkpr = import
  "https://github.com/bitnami/kube-prod-runtime/raw/v1.0.0/manifests/platforms/gke.jsonnet";

bkpr {
  config:: import "kubeprod-autogen.json",

  nginx_ingress+: {
    controller+: {
      spec+: {
        template+: {
          spec+: {
            terminationGracePeriodSeconds: super.terminationGracePeriodSeconds * 2,
          },
        },
      },
    },
  },
}
```

Add/remove/modify *anything*

See https://jsonnet.org/ for full syntax

BKPR: Integration

One part of a much larger ecosystem

Building block, not an end-to-end solution

Version control, build, CI/CD

- kubecfg
- kubectl apply
- Jenkins
- GitLab Auto DevOps
- Argo

Application management

- Helm
- Kubeapps
- Ksonnet
- kubectl

… and *so* many more logos

BKPR: Integration

Knative Comparison

Knative

- elasticsearch
- fluentd
- kibana
- prometheus
- grafana
- node-exporter
- zipkin
- Istio
- **Knative builder**
- **Knative event queue**
- **Knative serving infrastructure**

Opinionated, featureful, developer-focused,
codifies a set of serving best-practices

BKPR

- elasticsearch
- fluentd
- kibana
- prometheus
- node-exporter
- alertmanager
- nginx-ingress
- cert-manager
- external-dns

Conservative, follows community, infrastructure-focused,
designed as a base to build on

Bitnami Kubernetes Production Runtime

Summary

- Avoid repeating ourselves across the community
- Known, consistent reference point
- Raise the water level beyond bare Kubernetes
  - (TLS is not optional!)

#gitops on kubernetes.slack.com

Give it a try: https://kubeprod.io/

Thank You

For more information
visit kubeprod.io

bitnami