Comparison self-made EKS and 4wheels

Angelegt von Simon Dreher, zuletzt geändert am 17. Mär. 2021

Category	self-made EKS	4wheels
Initial effort (on feature parity) (+maintenance effort, if it differs significantly)	for feature parity: add some options, kured helm chart, if needed CNI with NetworkPolicies (HPA with monitoring?,registry with gitlab?) deployment: one terraform apply (currently two applies, because of a bug)	for feature parity: add autoscaler (easy); IaC would quasi require writing it yourself deployment: click in manage.aws portal, some manual steps (replace cert, configure Kibana,...)
Day 2 operations (updates, scaling, ...)	• Updates: read changelogs, bump helm chart version, terraform apply • Scaling: cluster is auto-scaled • Disaster Recovery: terraform apply, deploy workload	• Updates: k8s itself: follow documented steps manually; tools: no documentation available (yet) • Scaling: cluster needs manual scaling (or simply deploying cluster autoscaler) • Disaster Recovery: click in manage.aws portal, do most/some manual changes again, deploy workload
Freedom to customize	As far as helm charts allow or if you want to manage without helm: complete freedom	Since 4wheels won't manage them anymore (as far as I understood), you can do with them whatever you want
Authentication (integration/SSO)	kubectl: BMW SSO via IAM monitoring etc: TBD	kubectl: BMW SSO via IAM monitoring etc: default basic auth, can be integrated with WebEAM
Available core services	• cluster autoscaler • ~~ebs-csi-driver~~ in-tree aws-ebs provider • external-dns • (dashboard - auth missing)	• manual resizing (due to difficulties with single worker node group and EBS volumes) • in-tree aws-ebs provider • Horizontal Pod Autoscaler • NetworkPolicy Enforcement (Calico) • KURED reboot manager • private container registry (ECR with node role; repos still need to be provisioned), public via nexus
other features / peculiarities	➕ whitelisted public access ➕ Infrastructure-as-Code	➕ Encryption at rest and for secrets (root_encrypted, StorageClass parameters, EKS KMS encryption) ➕ Extensive documentation ➕ Multiple storage classes out-of-the-box ➕ Support volume resizing ❓ 2nd CIDR for POD deployment ❓ Proxied internet access ❓ AWS Resouce Group ❓ Public image paths need to be modified (e.g. quay.io → lpnexus.bmgroup.net:18086; loses information on which host I want ...) ❓ Integration with ATC Jenkins (documented, but pretty manual; CI Template for working with ECR)
Monitoring (Prometheus/Alertmanger/Grafana)	➕ CI/CD Workflow ➕ Open for a (full or partial) managed solution (ℹ Should be updated after TOOLOPS-6775 is done)	➖ Manual editing of k8s resources (one-shot deployment) ➖ No external availability by default, self-signed certificate which needs to be replaced manually ➖ No discovery of alerts ➖ No Service/Ingress probes, only scraping ➖ Only basic auth by default ➕ metrics-server for HPA ➕ node-exporter ℹ Setup guide for oauth-proxy with BMW-Auth available (also possible with other deployments) ℹ Basic deployment with no managed services
Logging	ℹ will be defined in TOOLOPS-6773	➕ low cost 6$/month if sizing fits ➕ Log shipping preconfigured ➖ Manual initial setup steps required for Kibana ➖ Single node setup ➖ Manual editing and updates (one-shot deployment) ℹ only for pod logging; additional logs e.g. audit logging must be enabled manually for the EKS cluster ℹ sized for 2 GB log ingest/day (64GB)
Ingress / network connectivity	➖ No access to or from BMW internal network ➕ Internet-facing ingress ➕ Automated DNS and Certificates (LetsEncrypt) ➖ No support for NetworkPolicies out-of-the-box	➖ Only access from BMW internal network ❓ External ingress via manual creation and VPC hopping ➖ Certificates and (public) DNS via BMW processes ➖ Internet only accessible via proxy ➖ Manual editing and updates (one-shot deployment)

Summary

4Wheels provides an easy to use and cheap solution built with various good practices in mind. It is especially useful if you want to get started quickly and don't have to take care of long-term-maintenance e.g. experimental setups.

The primary drawback is the lack of day 2 operations support since it is intended as one-shot deployment. Provisioned services (e.g. Elasticsearch, Prometheus) are plain deployments in the cluster which need to be maintained manually.
Since we need to manage multiple stages, we strongly recommend using a solution that can be controlled programmatically.

Keine Stichwörter