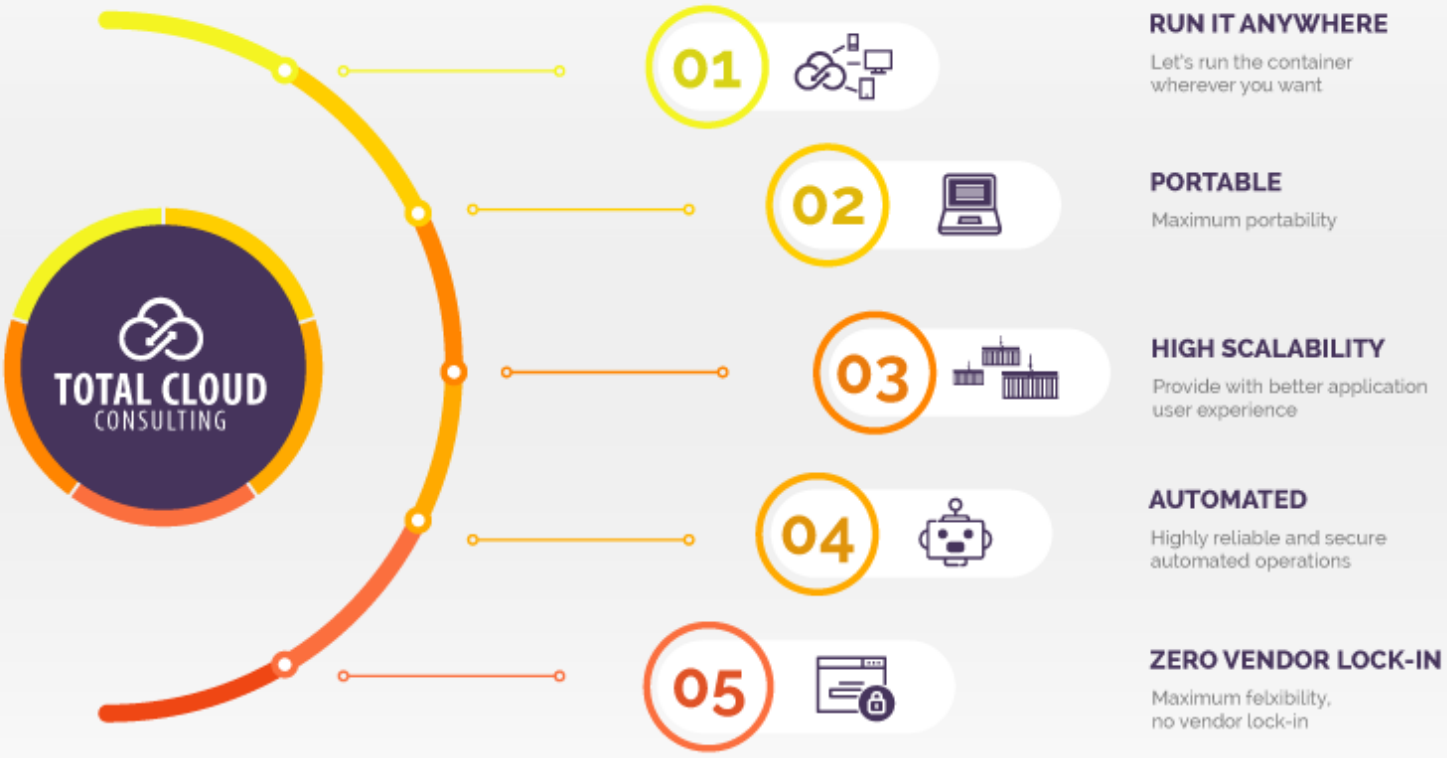
Abstract for **production grade Kubernetes deployment on AWS Cloud**

from Total Cloud Consulting

CONTAINERIZATION



All rights to this document are the property of Total Cloud Consulting Ltd. The use, duplication, unchanged or modified form of any part of this document other than the intention of Total Cloud Consulting Ltd. is forbidden unless the total consent of Total Cloud Consulting Ltd. is expressly granted.

© Total Cloud Consulting Ltd. 2015-2018. All rights reserved.

TOTAL CLOUD CONSULTING

Index

1. About This Guide

This deployment guide describes how easy to deploy a highly available, fault tolerant, full-scale Kubernetes environment on Amazon Web Services (AWS) Cloud, using Kubernetes Operations (kops) and AWS CloudFormation (CFN) templates together that automate the process. The result is 100% Kubernetes (and "kops") compatible deployment, what you can manage from either the Bastion host or HTTPS API endpoint remotely.

We use the popular tool "kops" because it is the easiest and most elegant way to get a production grade Kubernetes cluster up and running. We keep focus on security and transparency for the whole deployment process. The guide is for IT architects, administrators, and DevOps professionals who are planning to implement their Kubernetes workloads on AWS.

Combination of AWS Systems Manager (SSM) and AWS Lambda help with graceful cluster tear-down.

2. Overview

The deployment process bootstraps a scalable, full-scale Kubernetes cluster in a new, separated Virtual Private Cloud (VPC) environment. This is an automatically scaled cluster (Node level) using multiple AWS Availability Zones (AZs).

Our solution's main purpose is to automate, speed up and simplify Kubernetes Full-Scale AWS deployment. The cluster deployment finishes less than 20 minutes.

2.1. Kubernetes Cluster on AWS

All Kubernetes instances in the cluster are running in private VPC subnets (masters and nodes). The cluster API is accessible via public / internal load balancer (via Bastion Host locally, SSH port forward connections, or using AWS Virtual Private Gateway - IPSec VPN tunnel - which one is optional, not part of our current solution).

All Kubernetes Docker nodes and Bastion hosts send the logs to AWS CloudWatch Log Groups. The CFN template provides flexibility to choose VPC IP address ranges to be able to use multiple Kubernetes deployments (e.g. branches: dev/stage/production...) at the same time with on-site routing setup flexibility. The cluster IP separation helps to connect our AWS Kubernetes clusters easily via VPC peering connections as well if needed. In VPC, we use free Private Link (VPC Endpoint) setup to access AW S3 and DynamoDB resources without public internet access of private network resources. The templates also provide the option to choose the Kubernetes host image type:

- CoreOS (latest),
- Ubuntu 16.04 LTS,
- Debian Jessie (latest),
- CentOS 7.x (latest)
- Red Hat Enterprise Linux 7.x (latest)

We use Ubuntu 16.04LTS for Bastion host. It is fixed for now, later we will provide more options for other OS types. We keep focus on AWS CFN and code separation as well to provide clean templates and transparent, modularized bootstrap process.

The stack supports upgrades and rebuild for new versions (via kops utility). The "helm" Kubernetes package manager utility also comes pre-installed, so you can use it to deploy "charts" easily.

With **kops** our automated deployment provides:

- Built on a state-sync model for dry-runs and automatic idempotency
- Supports custom Kubernetes add-ons
- YAML Manifest Based API Configuration
- Supports upgrading from kube-up
- Capability to add containers, as hooks, and files to nodes via a cluster manifest

This guide doesn't cover general guidance and best practices for Kubernetes. For general open-source reference, check the Kubernetes documentation. For more Kubernetes on AWS best practices, use cases, support please consult with Total Cloud Consulting engineers.

We use Calico - Kubernetes CNI networking which is fixed in our configuration. More: https://github.com/kubernetes/kops/blob/master/docs/networking.md

The CloudFormation template gives option to pre-install some popular Kubernetes Add-On:

- Kubernetes Cluster Autoscaler
- Kubernetes Dashboard
- Kubernetes Heapster Monitoring
- HELM

A commonly used abbreviation for Kubernetes is "k8s" ("kates"), where the middle eight letters of the word are replaced with number "8".

2.2. TC2 GitHUB Repository

Our templates and bootstrap files published under Apache 2.0 are open source license.

Check our GitHUB repository for different versions:
https://github.com/totalcloudconsulting/kubernetes-aws

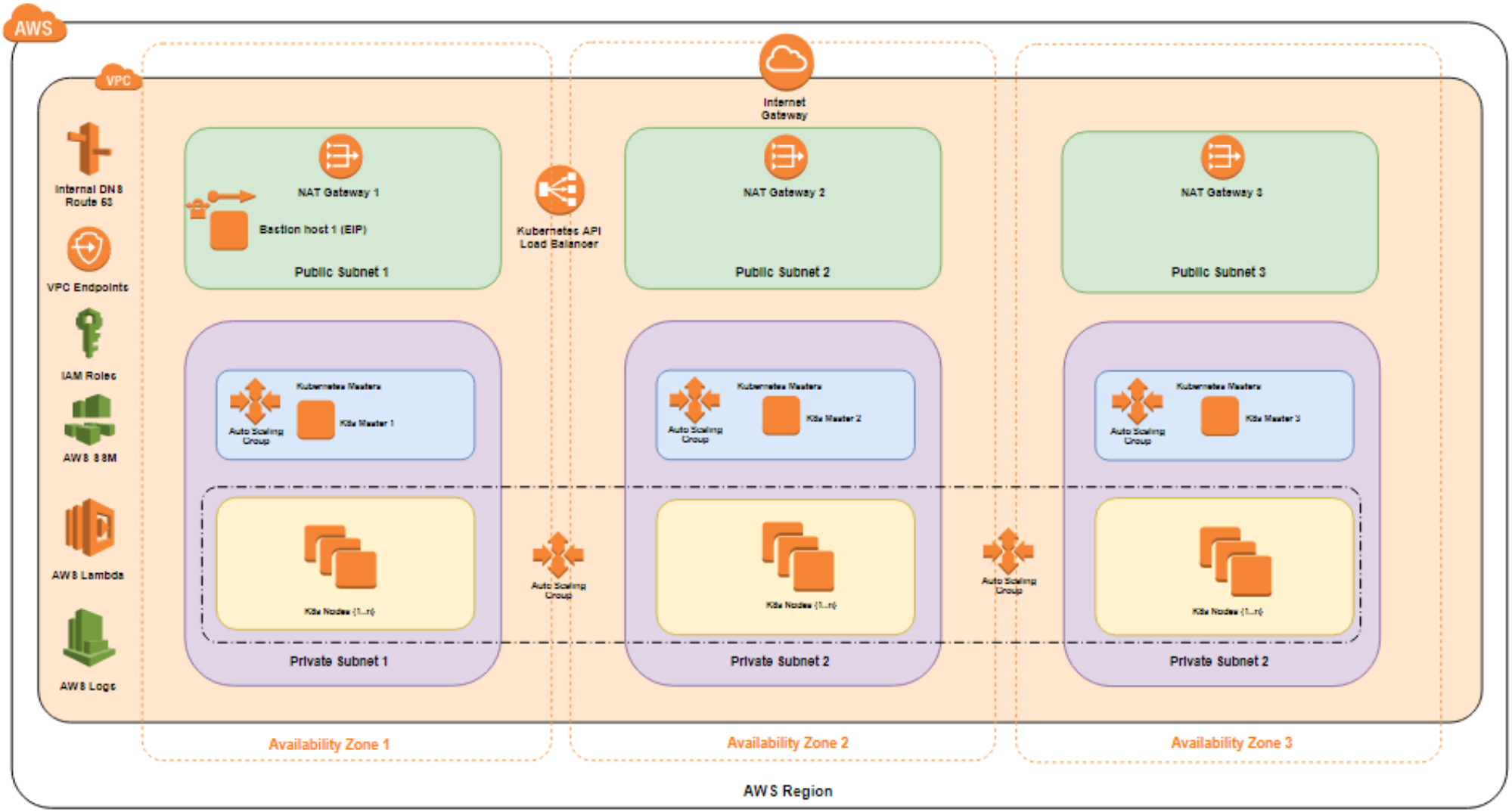
2.3. AWS Costs and software licenses

You are responsible for the cost of the AWS services used while running the resources described in this guide. We do not charge extra costs for using this documentation and programs, templates. Kubernetes and KOPS are free to deploy and uses the open-source Apache 2.0 license. **Total Cloud Consulting is not responsible for the costs you have in AWS using the above deployment methods!** By default, the deployment will be **six t2.medium** instances for the cluster and **one t2.small** instance for a bastion host. For cost estimates, see the pricing pages. You may need to open a an AWS Support Ticket to increases EC2 service limits.

2.4. Kubernetes Production Grade Architecture

The CloudFormation template sets up these core components:

- 1 VPC: 3 private and 3 public subnets (6) in 3 different Availability Zones, Private Link routes to S3 and DynamoDB (free)
- 3 NAT gateways in each public subnet in each 3 Availability Zones
- 3 self-healing Kubernetes Master instances in each Availability Zone's private subnet AutoScaling group (separate ASGs)
- 3 Node instances in AutoScaling groups, in each Availability Zone (one ASG)
- 1 self-healing Bastion host in 1 Availability Zone's public subnet, fixed Ubuntu 16.04 LTS,
- 5 Elastic IP Addresses: 3 for NAT Gateways, 1 -1 for Bastion hosts
- One internal or public ELB load balancer for HTTPS access to the Kubernetes API
- 2 CloudWatch Logs group for Bastion hosts and Kubernetes Docker images
- a Lambda function for graceful teardown through SSM
- 2 security groups 1 for Bastion host, 1 for Kubernetes Hosts (Master and Nodes)
- IAM roles for Bastion hosts, Nodes and Master instances
- An S3 bucket for kops state store
- 1 Route53 private zone for VPC


AWS
VPC
Internal DNS
Route 53
VPC Endpoints
IAM Roles
AWS S3M
AWS Lambda
AWS Logs
Internet
Gateway
NAT Gateway 1
Bastion host 1 (EIP)
Public Subnet 1
Kubernetes API
Load Balancer
NAT Gateway 2
Public Subnet 2
NAT Gateway 3
Public Subnet 3
Kubernetes Masters
Auto Scaling
Group
K8s Master 1
K8s Master 2
K8s Master 3
K8s Nodes (1..n)
Private Subnet 1
Private Subnet 2
Private Subnet 2
Availability Zone 1
Availability Zone 2
Availability Zone 3
AWS Region

1. Figure

The architecture on "1. Figure" is recommended for production. It provides the highest availability because each (3) Availability Zones contains Master/Node instances. This deployment stack has the highest deployment costs.

Kubernetes relies on a key-value store service called "etcd". "Etcd" uses the Quorum approach to keep consistency, which means that at least 51% of the nodes are required to be available to keep the system HA and avoid service interruption. The stack contains 3 Master instances deployed into 3 different Availability Zones, ensuring to have full HA and resiliency against an outage of one of the AZs, and to avoid losing "etcd" quorum.

We recommend to use AWS Regions with at least 3 Availability Zones nevertheless you can start the template with 2 AZs, but in case of a Zone failure, the Kubernetes etcd may turn to be inconsistent by losing 75% of quorum in the region.

3. Best Practices

The architectures built by this deployment guide supports AWS best practices for security. All of these setups actively use a Multi-AZ solution by deploying a Kubernetes cluster in each template into multiple zones, with automatic recovery AutoScaling groups and auto-recovery Bastion hosts (by CloudWatch).

The full-scale deployment (1. Figure) architecture provides the highest level of availability for Kubernetes services with **NO SPOF (single point of failure)**, but it uses the largest amount of resources. We recommend it for production use.

You can take full control over the whole cluster:

- access and control the cluster from Bastion host
- The kops, kubectl and helm binaries are pre-installed and configured on the Bastion host.
- via Kubernetes Master external (public ELB) API access, you can use "kops" and "kubectl" immediately from any remote machine after getting the configuration from the Bastion host.

You can take advantage of security because of the Kubernetes API and Router 53 zone is available only in the VPC or via VPN channels. The Kubernetes applications can be exposable to the public internet via public ElasticLoadBalancer (ELBv2) or Application Load Balancer (ALB).

All of the deployments above directly logs into AWS CloudWatch Logs (aws logs) into two Log Groups.

We provide advanced configurations with pre-installed and pre-configured OpenVPN service on bastion host.

3.1. AWS Log

- K8s-Bastion-${K8sClusterName}
  - This Log Group is created for Bastion Host(s), to have a full view of events happening during the cluster bootsrap.
- K8s-ALL-Docker-Logs-${K8sClusterName}
  - This Log Group is created to host Docker container's system logs

4. Deployment Steps

Follow these steps to deploy Kubernetes with KOPS on AWS. For detailed instructions, follow the links for each step.

- Step 1. Prepare Your AWS account
  - Sign up for an AWS account. Choose your region, create a key pair, and request increases for account limits, if necessary.
- Step 2. Launch the Kubernetes Cluster
  - Choose and launch the AWS CloudFormation template, specify parameter values, and create the stack. The deployment provides 3 different templates for different level of HA and purpose

****Note:** You are responsible for the cost of the AWS services used while running this deployment steps There is no additional cost for using this documentation and source codes. The AWS CloudFormation template for this document includes configuration parameters that you can customize. Some of these settings, such as instance type, will affect the cost of deployment.

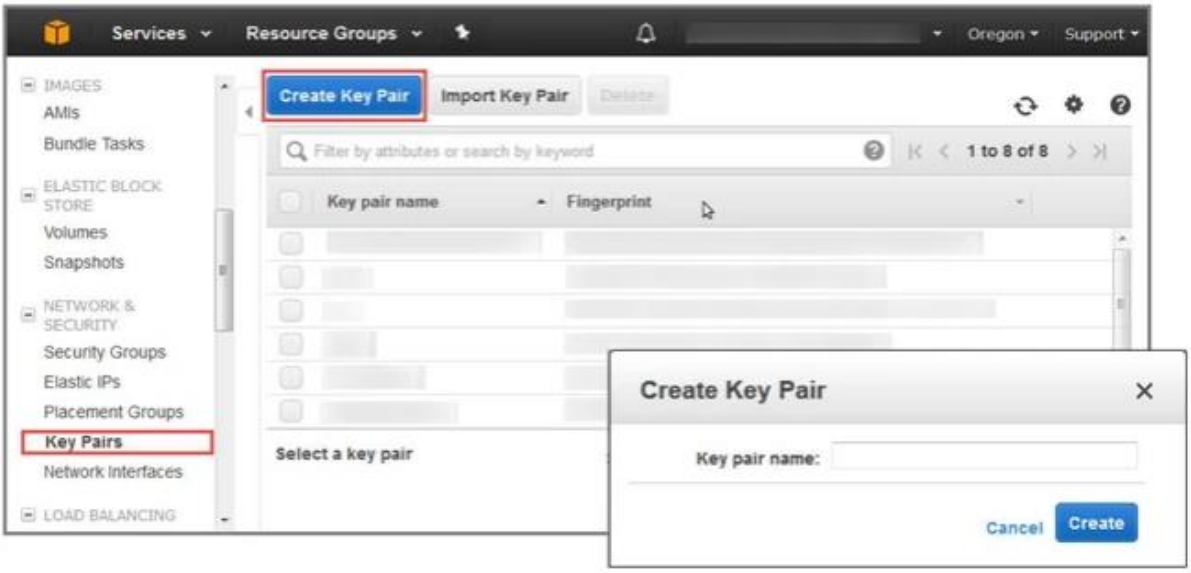
4.1. Prepare Your AWS account

***If you already have an AWS account, skip to step 2.*

1. Create your AWS account at https://aws.amazon.com by following the on-screen instructions. Part of the sign-up process involves receiving a phone call and entering a PIN using the phone keypad. We recommend to create and use IAM user with elevated (Administrator) privileges to deploy the Kubernetes Cluster because the process needs rights to create and access wide range of resources (including IAM role creation). Log in with your new IAM user on AWS WEB console.

2. Use the region selector in the navigation bar to choose the AWS Region where you want to deploy Kubernetes on AWS. For more information, see Regions and Availability Zones. Regions are dispersed and located in separate geographic areas. Each Region includes at least two Availability Zones that are isolated from one another but connected through low-latency links. Consider choosing a region closest to your data center or corporate network to reduce network latency between systems running on AWS and the systems and users on your corporate network.

3. Create an Ec2 Key Pair in your preferred region. To do this, in the navigation pane of the Amazon EC2 console, choose Key Pairs, Create Key Pair, type a name, and then choose Create.



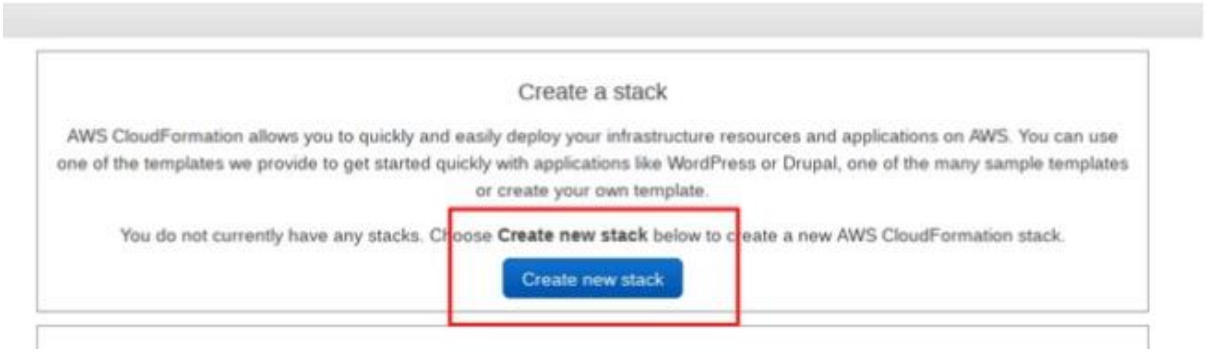
2. Figure

Amazon EC2 uses public-key cryptography to encrypt and decrypt login information. To log in to your Bastion and Kubernetes Master/Node instances, you must create a key pair. On Linux, the key pair is used to authenticate SSH login.

4.2. Launch the Kubernetes Cluster

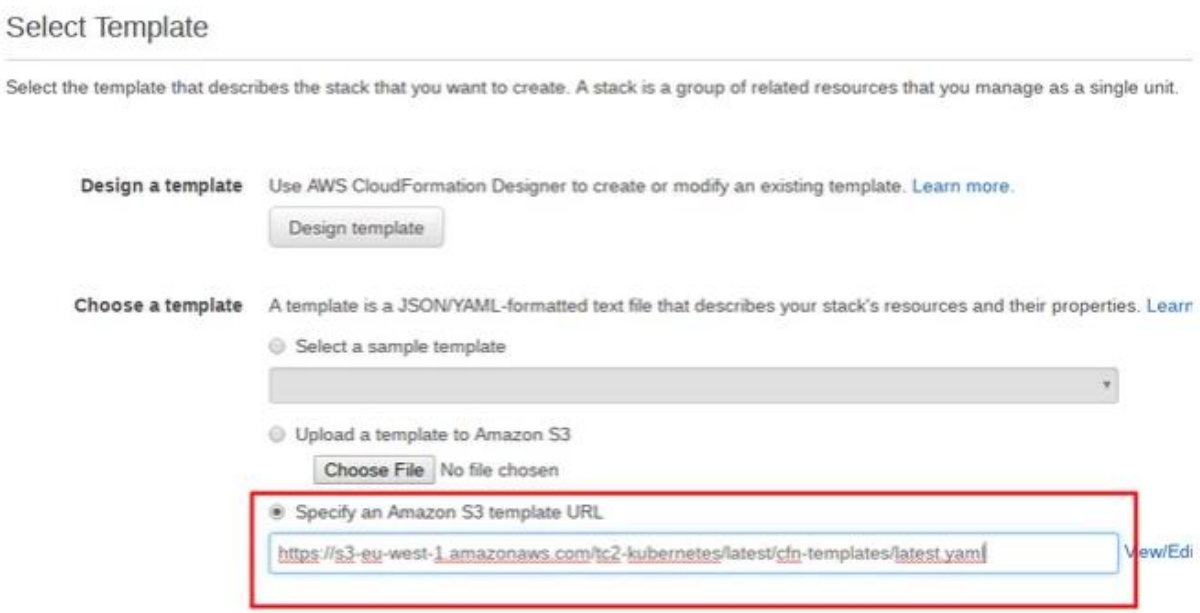
One-Click Launch (new VPC)

Go to the AWS region's AWS CloudFormation stack page and choose "Create new stack"
(example URL for Ireland:
https://eu-west-1.console.aws.amazon.com/cloudformation/home?region=eu-west-1)



3. Figure

Choose "Specify an Amazon S3 template URL":



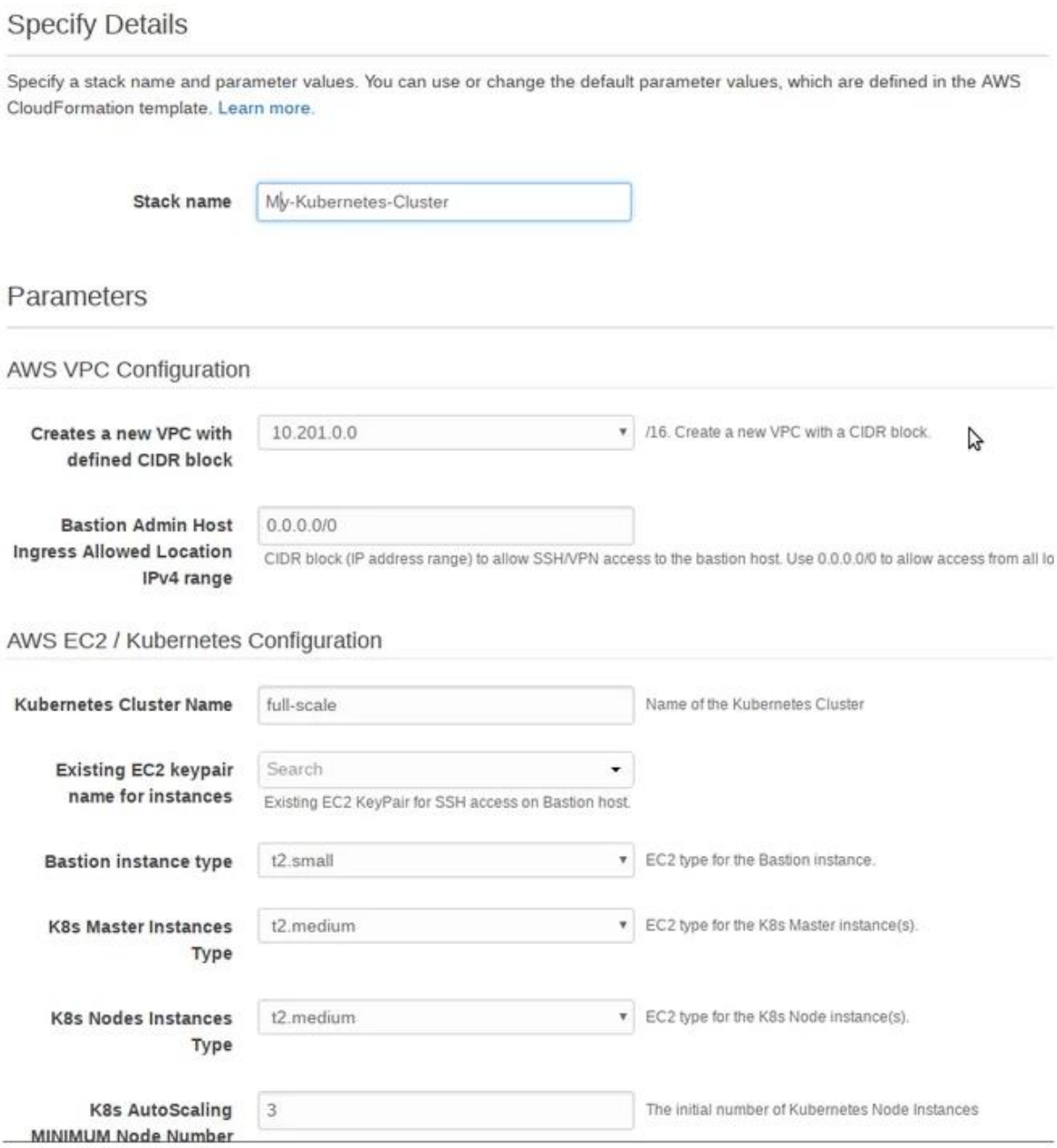
4. Figure

We provide a public S3 bucket for templates and bootstrap:

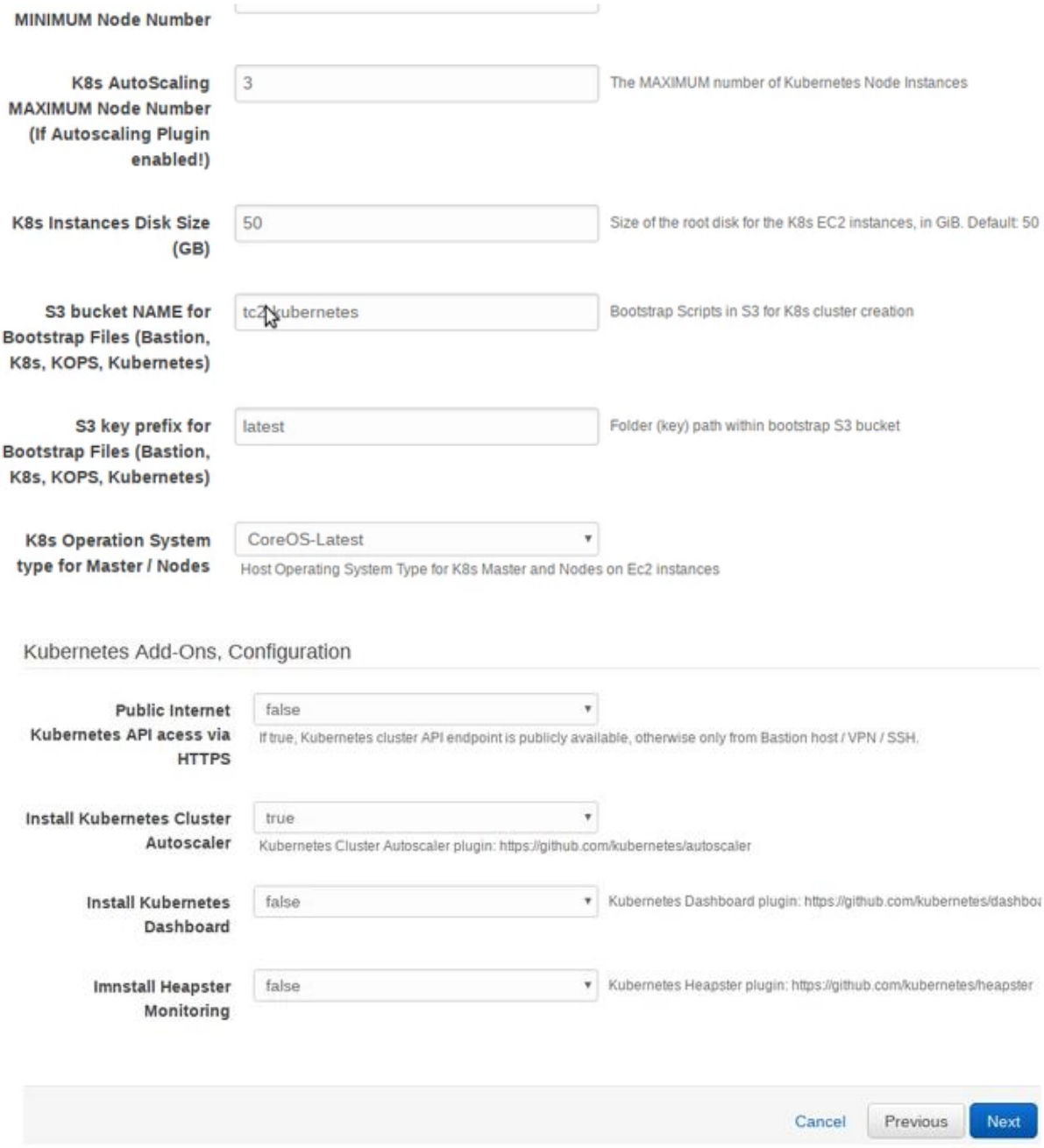
COPY + PASTE LINK THERE: https://s3-eu-west-1.amazonaws.com/tc2-kubernetes/latest/cfn-templates/latest.yaml

Then click "Next" ...

Parameters for deploying Kubernetes



5. Figure



6. Figure

On “5. Figure” and “6. Figure” you can see the default parameters for Kubernetes CFN template.

4.2.1. Step 1. AWS CloudFormation Required Parameters

Stack Name:

- name of the CFN stack

Existing EC2 keypair name for instances:

- AWS EC2 Key Pair: which is used to create and then access Bastion / K8s Master and K8s Node Ec2 instances

4.2.2. Step 2. AWS VPC Configuration

Creates a new VPC with defined CIDR block:

- Choose IPv4 "/16" CIDR block to the newly created VPC: 10.{201|202|203|204|205}.0.0/16 (Default: 10.201.0.0/16)
- You have choices here to avoid IP address collision in case of routing / VPC Peering.

Bastion Admin Host Ingress Allowed Location IPv4 range:

- Limit the public IP address from where you can connect to the Bastion instance (recommended to keep 0.0.0.0/0: from everywhere)

4.2.3. Step 3. AWS EC2 / Kubernetes Configuration

Kubernetes Cluster Name:

- name of your Kubernetes cluster

Existing EC2 keypair name for instances:

- Required! Choose an EC2 Key-Pair you created previously.

Bastion instance type / K8s Master Instances Type / K8s Nodes Instances Type:

- EC2 instance types for the different instances (check Figure #1-3). If necessary, request a service limit increase for the types you choose. Default: t2.micro / t2.small

K8s AutoScaling MINIMUM Node Number:

- Kubernetes cluster / Node instance initial size. Default: 3 (1-1 for 3 AZs)

K8s AutoScaling MAXIMUM Node Number (If Autoscaling Plugin Installed!):

- Kubernetes cluster: if you choose (by default: yes) Kubernetes AutoScaler plugin to be installed, it is effective to limit maximum number of Kubernetes Nodes Ec2 instances in the deployment. (Default: 3, 1-1 for 3 AZs)

K8s Instances Disk Size (GB):

- Attached EBS GP2 disk size (in GB) to the Kubernetes Ec2 machines (both Master / Nodes). (Default: 50GB)

S3 bucket NAME for Bootstrap Files (Bastion, K8s, KOPS, Kubernetes):

- **IMPORTANT!!! the default value: "tc2-kubernetes" contains Total Cloud Consulting public templates and bootstrap scripts. If you change it, please create a bucket for yourself with your own files otherwise the stack startup fails.**

S3 key prefix for Bootstrap Files (Bastion, K8s, KOPS, Kubernetes):

- **IMPORTANT!! keep it "latest" for Total Cloud Consulting public template and bootstrap files. If you use your own bucket, you need to create the path give here contains the bootstrap files.**

K8s Operation System type for Master / Nodes:

- the host operation system type for Kubernetes Master / Nodes. Now you can choose:
  - CoreOS-Latest (Default)
  - Debian-Jessie
  - Ubuntu-1604
  - CenOs-7-Latest**
  - RHEL-7-Latest (Red Hat 7.x)**

****NOTE**: on Red Hat 7.x and CentOs 7.x the operation is not guaranteed because of different setup and security model (SELinux): https://github.com/kubernetes/kops/blob/master/docs/images.md

We tested all above Kubernetes host operation systems, all works well and stable.

4.2.4. Step 4. Kubernetes Add-Ons, Configuration

Public Internet Kubernetes API access via HTTPS:

- if 'true', you can access the Kubernetes API (with kubectsl, helm, etc...) from your remote machine directly. It is widely opened!
- if 'false' you can access the Kubernetes API only from the bastion host (or optional VPN configuration)

Install Kubernetes Cluster Autoscaler:

- Kubernetes Cluster Autoscaler plugin: https://github.com/kubernetes/autoscaler (Default: true)
- File path: *bootstrap/cluster-autoscaler.yml*

Install Kubernetes Dashboard:

- Kubernetes Dashboard plugin: https://github.com/kubernetes/dashboard (Default: false)
- File path: *bootstrap/kubernetes-dashboard.yaml*

Install Heapster Monitoring:

- Kubernetes Heapster plugin: https://github.com/kubernetes/heapster (Default: false)
- File path: *bootstrap/monitoring-standalone.yaml*

4.2.5. Step 5. AWS CloudFormation Custom Parameters - "Finish"

Click on "Next"

On the **Options page**, you can specify tags (key-value pairs) for resources in your stack and set advanced options. The template automatically names EVERY RESOURCE it creates as appropriate.

On the **Review page**, review and confirm the template settings. Under Capabilities, select the check box to acknowledge that the template will create IAM resources.

Choose "**Create**" to deploy the stack. Monitor the status of the stack. When the status is CREATE_COMPLETE, the Kubernetes cluster is ready.

5. Connecting to Kubernetes cluster

5.1. Validation the Kubernetes cluster

For remote SSH users, check the CFN Output and look for:

- BastionHostPublicIp

You need the Ec2 Key Pair - private one - you created and used to initiate the cluster previously.

The easiest way to validate the cluster is using SSH login to the Bastion host.

```
# Login to Bastion host
ssh -i /path/to/ec2/keypair/KeyPairPrivate ubuntu@{BastionHostPublicIp1}

# example:
# ssh -i /home/user/.ssh/kubernetes-keypair.pem ubuntu@52.50.51.74

#check Kubernetes cluster
kops validate cluster
```

If you look something similar above, the K8s cluster is up and running

****NOTE**: on Windows use WinSCP utility to get the keys.

5.2. Access public Kubernetes cluster from remote machine

Download and install "kops" and "kubectl" and AWS CLI binaries:

KOPS: https://github.com/kubernetes/kops/blob/master/docs/install.md

KUBECTL: https://kubernetes.io/docs/tasks/tools/install-kubectl/

AWS CLI**: https://aws.amazon.com/cli/

****NOTE**: set up AWS CLI with previously generated IAM user you use (AWS AccessKey ID and AccessKey): https://docs.aws.amazon.com/cli/latest/reference/configure/index.html (use --profile to have different profile for new IAM credentials)

If we SET "**Public Internet Kubernetes API access via HTTPS**" CFN option "**true**" we can access the Kubernetes API via public ELB HTTPS endpoint.

Download kops / kubectl configuration files from Bastion host (check CFN Output)

```
# download configuration files for Kops / Kubectl from Bastion host
scp -i /path/to/ec2/keypair/KeyPairPrivate ubuntu@{BastionHostPublicIp}:/opt/kops-state/* ./

# example:
# scp -i /home/tobi/.ssh/kubernetes-keypair.pem ubuntu@52.50.51.74 ubuntu@52.50.51.74:/opt/kops-state/* ./
#.....

ls -l KOPS*
-rw-r--r-- 1 root root 31 jan   23 19:26 KOPS_AWSLOGS
-rw-r--r-- 1 root root 38 jan   23 19:26 KOPS_CLUSTER_NAME
-rw-r--r-- 1 root root 48 jan   23 19:26 KOPS_PRIVATE_SUBNETS
-rw-r--r-- 1 root root 48 jan   23 19:26 KOPS_PUBLIC_SUBNETS
-rw-r--r-- 1 root root 15 jan   23 19:26 KOPS_R53_PRIVATE_HOSTED_ZONE
-rw-r--r-- 1 root root 12 jan   23 19:26 KOPS_SECURITY_GROUP
-rw-r--r-- 1 root root 40 jan   23 19:26 KOPS_STATE_STORE
```

Connect to Kubernetes cluster via public HTTPS endpoint

```
# set up environment variables
export KOPS_STATE_STORE=`cat ./KOPS_STATE_STORE`
export KOPS_CLUSTER_NAME=`cat ./KOPS_CLUSTER_NAME`

# NOTE: if you use different AWS CLI profile, or different AWS REGION set it up:
# export AWS_DEFAULT_PROFILE=your-k8s-profile-name
# export AWS_PROFILE=your-k8s-profile-name
# export AWS_DEFAULT_REGION=your-region

# export kubectl config
kops export kubecfg --name $KOPS_CLUSTER_NAME
kops has set your kubectl context to demo-public.k8s.local

# validate cluster with kops from local machine

kops validate cluster
...
```

We have full control over the Kubernetes cluster with kops and kubectl via HTTPS ELB endpoint.

6. Troubleshooting

Sometimes the RHEL and CentOS host OS has problems starting Kubernetes PODs. In that case the CFN stack created but the "kops validate cluster" command shows the problematic pod. You can check logs:

https://eu-west-1.console.aws.amazon.com/cloudwatch/home?region=eu-west-1#logs:

Bastion host(s) logs: *K8s-Bastion-{your cluster name}*, e.g.: K8s-Bastion-demo-public

Kubernetes Docker container logs: K8s-ALL-Docker-Logs-*{your cluster name}*, e.g.: K8s-ALL-Docker-Logs-demo-public

List Kubernetes instances

```
kubectl get nodes --all-namespaces
```

Login users for different Kubernetes OS hosts (Masters & Nodes):

- **RHEL**: *ec2-user*
- **Ubuntu**: the user name is *ubuntu*
- **Centos**: the user name is *centos*
- **Debian**: the user name is *admin*
- **CoreOs**: *core*

****NOTE**: The CFN stack after deletion MAY leave back S3 kops-state-* buckets and AWS Log groups, although it initiates cleanup via Lambda+SSM.

7. Security

This stack exposes port 22 and 1194 on the bastion host and port 443 on the API ELB load balancer (ONLY if you choose public Internet access for K8s API). You can limit the IP address range(s) of access to Bastion host. The optional public access of Kubernetes HTTPS API is opened to the wild. you can limit it to IP address ranges by editing the ELB's security group. All nodes within the stack can reach one another on all ports. All nodes are running in private subnets, without public IP. To expose more ports to Bastion host (and/or to API ELB), we recommend using Elastic Load Balancing. Via OpenVPN, the routing is one direction and NAT-ed from client side. Open VPN clients can initiate connections, the Kubernetes hosts cannot see back to the VPN network. For general AWS security considerations, please visit the AWS Security Center.

8. Additional Resources

8.1. AWS services

- Amazon EC2 user guide for Linux: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/
- AWS CloudFormation: https://aws.amazon.com/documentation/cloudformation/
- Amazon VPC: https://aws.amazon.com/documentation/vpc/
- AWS SSM: https://docs.aws.amazon.com/systems-manager/latest/APIReference/Welcome.html
- AWS Route53: https://aws.amazon.com/route53/
- AWS Lambda: https://aws.amazon.com/lambda/

8.2. Kubernetes documentation

- Kubernetes Open-Source Documentation: https://kubernetes.io/docs/
- Calico Networking: http://docs.projectcalico.org/
- KOPS documentation: https://github.com/kubernetes/kops/blob/master/docs/aws.md , https://github.com/kubernetes/kops/tree/master/docs
- Kubernetes Host OS: https://github.com/kubernetes/kops/blob/master/docs/images.md

8.3. OpenVPN

- https://community.openvpn.net/openvpn/wiki/FAQ

9. Send Us Feedback

We welcome your questions and comments!

Please reach out to us at Total Cloud Consulting WEB page

You can visit TC2's GitHub repository to download the templates and scripts for this public release of the deployment guide. Total Cloud Consulting will be updating this guide on a regular basis.

10. Document Revisions

1. 2018.01.23 / Initial Release
2. 2018.02.08 / Abstract for full-scale Kubernetes deployment