

Les éléments à télécharger sont disponibles à l'adresse suivante :
<http://www.editions-eni.fr>
Saisissez la référence de l'ouvrage **EPKUB** dans la zone de recherche
et validez. Cliquez sur le titre du livre puis sur le bouton de téléchargement.

Avant-propos

1. Présentation de Kubernetes	39
1.1 Un peu d'histoire	39
1.2 Qu'est-ce qu'un container ?	40
1.3 Les containers avant Docker	40
1.4 Pourquoi utiliser des containers ?	41
1.5 Problèmes introduits avec les containers	42
1.6 À quoi va servir Kubernetes ?	42
1.7 Ressources externes	43
2. Un mot sur l'application	43
2.1 Rien ne sert de courir	43
2.2 Les douze facteurs applicatifs	44
2.3 Microservices vs Monolithes	44

Chapitre 1

Introduction

1. Cibles et objectifs de l'ouvrage	45
2. Prérequis techniques et ressources documentaires	46
2.1 Prérequis techniques	46
2.2 Ressources documentaires	46
2.3 Récupération des fichiers d'exemples	47
3. Présentation générale	47
3.1 Prérequis	48
3.2 Utilisation de Kubernetes	48
3.3 Installation et configuration de Kubernetes	48

2 **Kubernetes**

Plateforme de déploiement de vos applications

3.4	Extension du cluster Kubernetes et notions avancées	49
3.5	Déploiement et intégration continue	49
3.6	Conventions utilisées	50

Chapitre 2

Installation de l'environnement Kubernetes

1.	Objectifs du chapitre et prérequis	51
2.	Alternative à l'installation en local	52
2.1	Pourquoi ces alternatives ?	52
2.2	Utilisation d'un service managé	52
2.3	Service Katacoda	53
3.	Mise en place de la commande kubectl	53
3.1	À quoi sert kubectl ?	53
3.2	Installation de kubectl	54
3.2.1	Installation sous Debian/Ubuntu	54
3.2.2	Installation sous CentOS/RHEL ou Fedora	54
3.3	Vérification de l'installation	55
3.4	Configuration de l'autocomplétion	55
3.4.1	Présentation du mécanisme d'auto-complétion	55
3.4.2	Fichier profile à modifier	56
3.4.3	Autocomplétion sur kubectl	56
3.4.4	Utilisation de la variable SHELL	57
4.	Mise en place de Minikube	57
4.1	Pourquoi faire appel à Minikube ?	57
4.2	Téléchargement et installation de Minikube	57
4.3	Vérification de l'installation de Minikube	58
4.4	Mise en place de l'auto-complétion	59
5.	Installation du cluster Kubernetes avec Minikube	59
5.1	Choix de l'hyperviseur	59
5.2	Préparation de l'hyperviseur VirtualBox	59
5.2.1	Installation de VirtualBox	59

- 5.3 Préparation de l’hyperviseur KVM/libvirt 60
 - 5.3.1 Installation de KVM et libvirt 60
 - 5.3.2 Installation du driver KVM pour Minikube 60
- 5.4 Configuration de l’utilisateur courant 61
- 5.5 Déploiement du cluster avec Minikube 62
 - 5.5.1 Création de la machine Minikube 62
 - 5.5.2 Arrêt/démarrage de la machine Minikube 65
 - 5.5.3 Container runtime alternatif 66
 - 5.5.4 Extensions de Minikube 67
 - 5.5.5 Suppression de la machine Minikube 68
- 6. Quelques notions sur le format YAML 68
 - 6.1 Déclaration de couples clés/valeurs 69
 - 6.2 Les tableaux en YAML 70
 - 6.3 Les structures clé/valeur ou table de hachage 71
 - 6.4 Tableau de table de hachage 72

Chapitre 3

Tableau de bord et ligne de commande

- 1. Objectifs du chapitre et prérequis 75
- 2. Préambule 76
 - 2.1 Origine du nom et du logo 76
 - 2.2 Pourquoi utiliser Kubernetes ? 76
 - 2.3 Origine de Kubernetes 77
 - 2.4 Fondation CNCF 77
 - 2.5 Les orchestrateurs du marché 78
- 3. Le tableau de bord de Kubernetes (dashboard) 78
 - 3.1 Présentation 78
 - 3.2 Tableau de bord Kubernetes sur service managé 79
 - 3.3 Déploiement du dashboard sur Minikube 79
 - 3.4 Accès au dashboard sur Minikube 79
 - 3.5 Structure du tableau de bord 80

4 **Kubernetes**

Plateforme de déploiement de vos applications

3.6	Création d'un déploiement	81
3.6.1	Un petit mot sur MailHog	81
3.6.2	Lancement du déploiement	82
3.7	État d'un déploiement	84
3.7.1	Consultation de l'état du déploiement	84
3.7.2	Consultation du gestionnaire de réplicats	86
3.7.3	Consultation de l'état d'un pod	87
3.7.4	Journal d'activité du container	88
3.7.5	Scalabilité	89
3.7.6	Mise à jour de l'application	90
3.7.7	Pour résumer	92
4.	Présentation de l'outil kubectl	92
4.1	Préambule	92
4.2	Consultation des éléments	92
4.3	Liste des pods	93
4.4	Liste des machines d'un cluster	94
4.4.1	Connexion à la machine Minikube	94
4.4.2	Liste des nœuds d'un cluster	95
4.4.3	Affichage des caractéristiques étendues	95
5.	Le moteur Docker de Minikube	96
5.1	Initialisation de l'environnement	96
5.2	Installation de Docker	97
5.3	Vérification de la communication avec le moteur Docker	97
5.4	Les containers associés aux pods	98

Chapitre 4

Automatisation et publication d'une application

1.	Objectifs du chapitre et prérequis	101
2.	Gestion par kubectl d'une application	101
2.1	Suppression d'un déploiement	101
2.2	Création d'un déploiement	102
2.3	État du déploiement	103

- 2.4 Mécanisme des réplicats. 105
 - 2.4.1 Consultation des réplicats 105
 - 2.4.2 Description des réplicats 106
- 2.5 État du pod 107
 - 2.5.1 Liste des pods 107
 - 2.5.2 Détails de l'état d'un pod 108
- 2.6 Accès aux logs des containers 109
- 2.7 Accéder à l'application MailHog 110
- 3. Exposition de services. 111
 - 3.1 Pourquoi utiliser un service ? 111
 - 3.2 Exposition d'un déploiement via un service 112
 - 3.3 Vérification du service mailhog 112
 - 3.4 Que faire en cas d'absence de shell ? 113
 - 3.5 Résilience et scalabilité. 116
 - 3.5.1 Origine du besoin 116
 - 3.5.2 Scalabilité manuelle 116
 - 3.5.3 Nombre de pods associés à un déploiement 116
 - 3.5.4 Arrêter temporairement une application 117
- 4. Automatisation de déploiement par fichier YAML 118
 - 4.1 Mécanisme de création et mise à jour 118
 - 4.2 Structure YAML d'un déploiement 119
 - 4.2.1 Quelques rappels. 119
 - 4.2.2 Récupération d'une structure au format YAML 119
 - 4.2.3 Édition d'un déploiement. 121
 - 4.2.4 Squelette pour un déploiement 121
 - 4.2.5 Création d'un déploiement à l'aide d'un fichier 123
 - 4.2.6 Suppression des éléments d'un fichier 124
 - 4.2.7 Gestion de l'idempotence et de la réentrance 124
 - 4.3 Création du service 126
 - 4.3.1 Définition du service 126
 - 4.3.2 Application de la définition du service 127
 - 4.3.3 Gestion de la réentrance. 127
 - 4.4 Mécanisme de sélecteur et labels 128

6 **Kubernetes**

Plateforme de déploiement de vos applications

4.5	Regroupement de la création des éléments.	129
4.5.1	Création d'un groupe d'objets	129
4.5.2	Consultation de l'état d'un groupe d'objets	130
4.6	Structure des objets	131
4.6.1	Interrogation de Kubernetes avec kubectl	131
4.6.2	Référence de l'API en ligne.	132
5.	Ingress et reverse proxy	132
5.1	Origine du besoin	132
5.2	Activation du contrôleur Ingress dans Minikube.	133
5.3	Déclaration d'une règle Ingress	134
5.4	Consultation des règles Ingress	134
5.5	Désactivation de la redirection HTTP vers HTTPS	136
5.6	Hôte virtuel et nip.io	137
5.6.1	Hôte virtuel par défaut.	137
5.6.2	Présentation du mécanisme de nip.io	138
5.6.3	Création d'un hôte virtuel pour mailhog	139

Chapitre 5

Cycle de vie d'un container dans Kubernetes

1.	Objectifs du chapitre et prérequis	141
2.	Gestion des crashes d'application	142
2.1	Consultation de l'état des pods	142
2.2	Connexion au pod	142
2.3	Container associé à MailHog.	143
2.4	Comportement en cas de crash	145
2.5	État du container après redémarrage du pod	146
2.6	Container vu depuis Docker (Minikube)	146
2.7	Attention au nettoyage	148
3.	État d'un container.	148
3.1	Pourquoi scruter l'état d'un container ?	148
3.2	Readiness vs Liveness	149
3.3	Utilisation et bonne pratique	150

3.4	Structure des champs de surveillance	150
3.5	Vérification de la présence d'un port.	151
3.5.1	Définition de la surveillance	151
3.5.2	Test d'indisponibilité sur un pod non prêt	153
3.5.3	État des pods en cas d'indisponibilité	153
3.5.4	Test d'indisponibilité sur un pod en mauvaise santé.	154
3.5.5	État des pods en cas de problème sur un pod	154
3.5.6	Attention à la consistance des tests	155
3.5.7	Uniformisation des tests	157
3.6	Surveillance HTTP	157
3.6.1	Pourquoi privilégier ce type de surveillance ?	157
3.6.2	Surveillance de l'application MailHog.	158
3.7	Point d'entrée de surveillance HTTP d'une application.	158
3.7.1	Un mot sur les frameworks modernes	158
3.7.2	Présentation de l'application Flask	159
3.7.3	Exemple de déclaration	159
3.7.4	Déploiement de l'application Flask	160
3.7.5	Consultation de l'état de l'application	161
3.8	Lancement d'un shell	162
3.8.1	Principe de fonctionnement.	162
3.8.2	Exemple de surveillance d'une base Postgres	163
3.8.3	Déclaration de la commande	163
4.	Définition de la capacité d'un pod.	164
4.1	Pourquoi définir une capacité ?	164
4.2	Réservation et surallocation.	164
4.3	Allocation de ressources à un container	165
4.4	Allocation de ressources à l'application MailHog.	166
4.5	Comportement en cas de saturation des ressources.	167
4.5.1	Demande trop importante de CPU	167
4.5.2	Dépassement de la mémoire allouée	168
4.6	Priorité d'un pod	169
4.6.1	Présentation du mécanisme	169
4.6.2	Consultation des types par défaut	169

4.6.3	Consultation des priorités des pods	170
4.6.4	Création d'une classe de priorité	171
4.6.5	Affectation d'une classe de priorité personnalisée	172
4.6.6	Remarque sur les classes de priorité par défaut	173

Chapitre 6

Persistance des données

1.	Objectifs du chapitre et prérequis	175
2.	Persistance des données	175
2.1	Origine du besoin	175
2.2	Utilisation d'un volume persistant externe	176
2.3	Volumes persistants	177
2.3.1	Structure du volume persistant	177
2.3.2	Création du volume persistant	178
2.4	Persistance de données avec MailHog	178
2.4.1	Opérations à réaliser	178
2.4.2	Déclaration de l'objet PersistentVolumeClaim	179
2.4.3	État des objets de volume persistant	180
2.4.4	État de la demande de volume persistant	181
2.4.5	Déclaration du point de montage	181
2.4.6	Ajout d'un point de montage sur le container	182
2.4.7	Options de lancement de MailHog	182
2.4.8	Déclaration entière suite aux modifications	183
2.5	Test de la persistance	185
2.5.1	Installation de mhsendmail	185
2.5.2	Ouverture de la communication avec le port SMTP	186
2.5.3	Envoi d'un mail	186
2.5.4	Droits du répertoire de persistance des données	187
2.5.5	Consultation de l'interface MailHog	188
2.5.6	Suppression des pods	189
2.5.7	Vérification du fonctionnement de la persistance	189

- 3. Classes de stockage 190
 - 3.1 Origine du besoin 190
 - 3.2 Liste des classes de stockage 191
 - 3.3 Détail d’une classe de stockage 192
 - 3.4 Classe de stockage par défaut 192
 - 3.5 Les différentes classes de stockage 193
 - 3.5.1 Les différentes familles 193
 - 3.5.2 Origine de ces familles 193
 - 3.6 Caractéristiques des classes de stockage 194
 - 3.6.1 Modes d’accès 194
 - 3.6.2 Caractéristiques de certains pilotes 195
 - 3.6.3 Liste des pilotes chargés 196
 - 3.7 Déclaration d’une classe de stockage 197
 - 3.7.1 Structure de la déclaration 197
 - 3.7.2 Exemple de déclaration 197
 - 3.8 Test de création automatique d’un volume persistant 198

Chapitre 7

Hébergement d’application en cluster

- 1. Objectifs du chapitre et prérequis 201
- 2. Déploiement d’une base de données MariaDB 201
 - 2.1 Origine du besoin 201
 - 2.2 Déploiement 202
 - 2.2.1 Choix de l’image Docker 202
 - 2.2.2 Version initiale du fichier de déploiement 202
 - 2.2.3 Gestion de la réentrance 203
 - 2.3 Volume persistant 204
 - 2.3.1 Demande de volume persistant 204
 - 2.3.2 État de la demande de volume persistant 204
 - 2.3.3 Ajout d’une persistance sur le container de MariaDB . . 205
 - 2.3.4 Consultation de l’état du déploiement 206
 - 2.4 Configuration de la base de données 207

2.5	Consultation de l'état du pod	208
2.5.1	Liste des pods	208
2.5.2	Connexion au container	209
2.6	Surveillance de la base de données	209
2.6.1	Définition des commandes de surveillance	209
2.6.2	Application de la modification	211
2.6.3	Vérification du déploiement	211
2.7	Mécanisme de déploiement	212
3.	Mise en place d'un StatefulSet	214
3.1	Augmentation du nombre de pods associés au déploiement	214
3.2	Présentation du type StatefulSet	216
3.2.1	Caractéristiques	216
3.2.2	Limitations	216
3.3	Déclaration du premier objet StatefulSet	217
3.3.1	Purge de l'ancien déploiement	217
3.3.2	Modifications à réaliser	217
3.3.3	Création du StatefulSet	218
3.3.4	État des volumes persistants	219
3.3.5	Suppression des anciens objets PV/PVC	220
3.4	Scalabilité de l'objet StatefulSet	220
3.5	Pods et volumes persistants d'un objet StatefulSet	220
3.6	Réduction de la taille du StatefulSet	221
4.	Base et compte de test	222
4.1	Variables d'environnement du container	222
4.2	ConfigMap et secret	223
4.2.1	Pourquoi y faire appel ?	223
4.2.2	Structure d'un objet ConfigMap	223
4.2.3	Déclaration d'un objet Secret	224
4.2.4	Rattachement au container	225

Chapitre 8

Mise en place d'une réplication entre pods

- 1. Objectifs du chapitre et prérequis 227
- 2. Synchronisation des pods MariaDB 228
 - 2.1 Exposition de la problématique 228
 - 2.2 Principe de fonctionnement de la synchronisation 228
 - 2.2.1 Opérations à réaliser 228
 - 2.2.2 Nombre de réplicats 229
 - 2.3 Identifiants des serveurs 229
 - 2.3.1 Connexion aux pods 229
 - 2.3.2 Connexion à la base de données 229
 - 2.3.3 Identifiants des serveurs 230
 - 2.3.4 ID du maître 230
 - 2.3.5 Création du compte de réplication sur le maître 231
 - 2.3.6 Configuration de l'esclave 232
 - 2.4 Activation de la synchronisation 232
 - 2.4.1 Activer les journaux pour la réplication 232
 - 2.4.2 Commande docker-entrypoint.sh 233
 - 2.4.3 Consultation de l'état du maître 234
 - 2.4.4 Configuration de l'esclave 234
 - 2.5 Test de la réplication 236
 - 2.5.1 Connexion au maître 236
 - 2.5.2 Création d'une table 236
 - 2.5.3 Connexion à l'esclave 237
- 3. Automatisation de la synchronisation 238
 - 3.1 Scripts de démarrage et synchronisation 238
 - 3.1.1 Script de démarrage 238
 - 3.1.2 Configuration de la synchronisation 239
 - 3.1.3 Scripts SQL additionnels 240
 - 3.1.4 Script d'arrêt de la base 240
 - 3.2 Scripts et objet ConfigMap 241
 - 3.3 Création du ConfigMap 241

3.4	Montage du ConfigMap	244
3.4.1	Référencement du ConfigMap dans la liste des volumes	244
3.4.2	Point de montage du ConfigMap	245
3.5	Démarrage et arrêt du container	245
3.5.1	Commande de démarrage	245
3.5.2	Commande d'arrêt de la base	246
3.6	Résumé des modifications	246
3.7	État du déploiement	248
3.7.1	État des pods	248
3.7.2	Journaux d'activité du pod esclave	248
3.7.3	Test de la synchronisation	248
3.7.4	Vérification du fonctionnement de la synchronisation	249

Chapitre 9

Gestion des briques internes de Kubernetes

1.	Objectifs du chapitre et prérequis	251
2.	Espace de noms kube-system	251
2.1	Pods présents dans l'espace de noms kube-system	251
2.2	CoreDNS	252
2.3	etcd	253
2.4	Le gestionnaire d'extensions de Minikube	253
2.5	Le serveur d'API	253
2.6	Le proxy Kubernetes (kube-proxy)	254
2.7	Le gestionnaire de tâches (scheduler)	254
2.8	Le gestionnaire de contrôle (controller manager)	254
2.9	Kubelet	254
3.	Configuration des serveurs maîtres	255
3.1	Principe de lancement des pods système	255
3.2	Contenu du répertoire /etc/kubernetes/manifests	255
3.3	Contenu des fichiers	256

- 3.4 Désactivation d'un pod système 257
- 3.5 Réactivation du pod système. 258
- 4. Monitoring des containers du cluster avec Glances 259
 - 4.1 Origine du besoin 259
 - 4.2 Consultation des DaemonSets 259
 - 4.3 Présentation de Glances 260
 - 4.4 Définition du DaemonSet 260
 - 4.4.1 Structure de la déclaration 260
 - 4.4.2 Champ volumes 261
 - 4.4.3 Champ containers 261
 - 4.5 Création du DaemonSet. 262
 - 4.5.1 Déclaration complète 262
 - 4.5.2 Création du DaemonSet. 262
 - 4.5.3 Consultation des pods 263
 - 4.6 Annotations de tolérance 264
 - 4.6.1 Présentation du mécanisme 264
 - 4.6.2 Récupération des annotations taints 264
 - 4.6.3 Tolérances de lancement 265
 - 4.6.4 Modification du DaemonSet 266
 - 4.7 Connexion à Glances 267

Chapitre 10

Helm - Gestionnaire de package

- 1. Objectifs du chapitre et prérequis 269
- 2. Présentation de Helm 269
 - 2.1 Pourquoi faire appel à Helm ? 269
 - 2.2 Principe de fonctionnement. 270
- 3. Déploiement de Helm 271
 - 3.1 Installation du client Helm 271
 - 3.2 Consultation de la version de Helm 271
 - 3.3 Configuration du client Helm 272

3.4	Initialisation de la partie serveur (Tiller) avec Helm 2.x	272
3.4.1	Un mot sur la sécurité (Helm 2.x)	272
3.4.2	Le compte de service de Tiller (Helm 2.x)	273
3.4.3	Création du compte de service	273
3.4.4	Attribution des droits d'administrateur au compte de service	275
3.4.5	Initialisation de Tiller	276
3.5	Suppression de Tiller (Helm 2)	278
4.	Déploiement d'une application avec Helm	278
4.1	Déterminer le package à déployer	278
4.2	Installation du package Wordpress	280
4.2.1	Un peu de vocabulaire	280
4.2.2	Installation avec Tiller	280
4.2.3	Installation sans Tiller	282
4.3	Corrections de l'installation	283
4.3.1	Quelques remarques	283
4.3.2	Spécification du nom et espace de noms	283
4.3.3	Lancement de l'installation	283
4.3.4	Mise à jour et réentrance	284
4.4	Éléments déployés avec Helm	285
4.5	Suppression d'un déploiement	286
4.6	Annulation de la suppression	287
4.7	Purge d'un chart Helm	288
5.	Cycle de vie d'une application déployée avec Helm	289
5.1	Ouverture du port vers WordPress	289
5.2	Connexion à WordPress	290
5.3	Configuration d'un chart Helm	291
5.3.1	Consultation des options d'un chart	291
5.3.2	Configuration de la publication (Minikube)	293
5.4	Historique de déploiement	295
5.5	Retour arrière	296
5.6	Portail Helm Hub	297

Chapitre 11
Contextes et outils tiers Kubernetes

- 1. Objectifs du chapitre et prérequis 299
- 2. Gestion des contextes avec kubectl 300
 - 2.1 Origine du besoin 300
 - 2.2 Lister les contextes 300
 - 2.3 Variable d’environnement KUBECONFIG 301
 - 2.3.1 Spécifier l’emplacement du fichier 301
 - 2.3.2 Spécifier plusieurs fichiers 301
 - 2.4 Changement de contexte 302
 - 2.5 Créer un contexte 302
 - 2.6 Supprimer un contexte 304
 - 2.7 Outils de gestion de contexte 304
 - 2.7.1 Présentation de kubectx et kubens 304
 - 2.7.2 Installation de kubectx et kubens 305
 - 2.7.3 Mise en place de l’autocomplétion 305
 - 2.7.4 Test des commandes 306
 - 2.8 Contexte dans le prompt utilisateur 306
 - 2.8.1 Pourquoi afficher le contexte ? 306
 - 2.8.2 Activation à l’aide de oh-my-zsh 307
 - 2.8.3 Activation avec bash 307
 - 2.8.4 Exemple d’affichage 307
 - 2.9 Changement des couleurs du terminal avec Konsole 308
 - 2.9.1 Principe de fonctionnement 308
 - 2.9.2 Création de la fonction 308
 - 2.9.3 Ajout de l’appel dans l’invite de commande 309
- 3. Utilitaires Kubernetes 310
 - 3.1 K9s : interface texte de suivi 310
 - 3.1.1 Contexte 310
 - 3.1.2 Installation de k9s 311
 - 3.1.3 Lancement de k9s 311
 - 3.2 Kubespy : espionnage de l’activité 312
 - 3.2.1 Présentation de Kubespy 312

3.2.2	Installation de Kubespy	312
3.2.3	Observation d'un déploiement avec Kubespy	312
3.3	Krew : gestionnaire d'extensions.	313
3.3.1	Présentation du mécanisme d'extensions	313
3.3.2	Installation de Krew	314
3.3.3	Test de l'extension	314
3.4	Sniff : capture du trafic réseau d'un pod	315
3.4.1	Principe de fonctionnement.	315
3.4.2	Installation de Sniff et Wireshark	316
3.4.3	Lancement d'une séance de capture	316
3.5	Kube Hunter : outil d'analyse du cluster	317
3.5.1	Présentation de Kube Hunter	317
3.5.2	Lancement de l'analyse	317
3.5.3	Résultat de l'analyse.	318
3.5.4	Publication de rapports HTML	319
3.5.5	Lancement à intervalles réguliers	321

Chapitre 12**Services managés Kubernetes**

1.	Objectifs du chapitre et prérequis	325
2.	Service managé de Google : GKE	326
2.1	Présentation du service Google	326
2.2	Administration depuis la console Google	326
2.3	Installation de la commande gcloud en local	328
2.3.1	Installation sur Debian/Ubuntu	328
2.3.2	Mise en place de l'autocomplétion	329
2.4	Configuration de l'environnement	330
2.4.1	Authentification auprès de Google Cloud	330
2.4.2	Projet associé avec le contexte courant	332
2.4.3	Activation de l'API	332
2.5	Gestion du cluster GKE	333
2.5.1	Consultation de la liste des clusters	333

2.5.2	Versions et régions disponibles	334
2.6	Création d'un cluster	334
2.6.1	Options de création	334
2.6.2	Lancement de la création du cluster	335
2.6.3	Récupération du fichier d'accès au cluster	336
2.7	Consultation du cluster	336
2.7.1	Liste des nœuds.	336
2.7.2	Services démarrés	337
2.8	Délégation des droits d'accès	338
2.8.1	Configuration des accès	338
2.8.2	Principe du mécanisme sous-jacent.	339
2.9	Configuration de Helm/Tiller 2.x	341
2.10	Suppression d'un cluster GKE	341
3.	Service managé Microsoft Azure : AKS	342
3.1	Présentation du service Azure	342
3.2	Administration depuis la console Azure	342
3.2.1	Présentation de la console	342
3.2.2	Consultation du tableau de bord Kubernetes.	344
3.3	Installation de la commande az en local.	345
3.3.1	Installation sur Debian/Ubuntu	345
3.3.2	Mise en place de l'autocomplétion	346
3.4	Authentification auprès du service Azure.	347
3.5	Emplacement de déploiement	348
3.5.1	Liste des emplacements	348
3.5.2	Versions disponibles de Kubernetes	349
3.6	Création d'un cluster	350
3.6.1	Création d'un groupe de ressources.	350
3.6.2	Lancement de la création du cluster	350
3.6.3	Récupération du fichier de connexion.	352
3.6.4	Zone DNS par défaut	352
3.7	Consultation de la liste des clusters	353
3.8	Délégation des droits d'accès	353
3.9	Configuration de Helm/Tiller 2.x	354

3.10	Suppression d'un cluster AKS	354
4.	Service managé d'Amazon : EKS	355
4.1	Présentation du service Amazon AWS	355
4.2	Introduction de la commande eksctl	356
4.3	Configuration des accès Amazon	356
4.4	Installation des binaires	360
4.4.1	Installation d'eksctl à l'aide de snap	360
4.4.2	Installation de l'outil aws cli	361
4.4.3	Vérification de la communication avec AWS	362
4.5	Création du cluster EKS	363
4.5.1	Aide en ligne d'eksctl	363
4.5.2	Options intéressantes à la création d'un cluster	363
4.5.3	Lancement de la création du cluster	364
4.6	Configuration des accès kubectl	366
4.7	Installation de aws-iam-authenticator	366
4.8	Délégation des droits d'accès	367
4.8.1	Configuration des accès	367
4.8.2	Principe du mécanisme sous-jacent	368
4.9	Configuration de Helm/Tiller 2.x	369
4.10	Suppression du cluster	370
5.	Accès en lecture-écriture multiple	370
5.1	Origine du besoin	370
5.2	Serveur NFS déployé dans Kubernetes	371
5.2.1	Limitations	371
5.2.2	Déploiement d'un serveur NFS	372
5.2.3	Vérification du déploiement	372
5.2.4	Test de création de volume persistant (PVC)	373
5.3	Service managé Amazon : EFS	373
5.3.1	Consultation des instances EFS	373
5.3.2	Création d'une instance EFS	374
5.3.3	Déploiement de la classe de stockage EFS	374
5.4	Service managé Google : Filestore	376
5.4.1	Consultation des instances Filestore	376

- 5.4.2 Création d'une instance Filestore 376
- 5.5 Classe de stockage NFS 377
 - 5.5.1 Installation du chart 377
 - 5.5.2 Vérification de l'installation du chart 378
 - 5.5.3 Test de création d'une demande
de volume persistant (PVC) 378
- 5.6 Classe de stockage Azure 379

Chapitre 13

Installation de Kubernetes en interne

- 1. Objectifs du chapitre et prérequis 381
- 2. Installation à l'aide de Kubespray 382
 - 2.1 Origine du besoin 382
 - 2.2 Pourquoi Kubespray ? 382
 - 2.3 Principe de Kubespray 382
 - 2.4 Prérequis des machines à administrer avec Ansible 383
 - 2.4.1 Échange de clés SSH 383
 - 2.4.2 Escalade de droits sudo 383
 - 2.5 Structure du cluster 385
 - 2.5.1 Architecture du cluster Kubespray 385
 - 2.5.2 Groupes de machines 385
 - 2.6 Préparation des éléments d'installation 386
 - 2.6.1 Clonage du dépôt Git 386
 - 2.6.2 Installation des prérequis 386
 - 2.7 Préparation de l'inventaire Ansible 387
 - 2.7.1 Qu'est-ce qu'un inventaire ? 387
 - 2.7.2 Fichier d'inventaire d'exemple 387
 - 2.7.3 Test de la communication Ansible/SSH 389
 - 2.8 Installation du cluster 390
 - 2.8.1 Configuration du cluster 390
 - 2.8.2 Lancement de l'installation 391
 - 2.8.3 Mise à jour du cluster 392

2.9	Accès au cluster	392
2.9.1	Configuration de l'accès au cluster	392
2.9.2	Utilisation d'un répartiteur de charge	392
2.9.3	Vérification de la communication avec le cluster	393
3.	Environnement embarqué avec k3s	394
3.1	Présentation et but du projet	394
3.2	Installation de k3s	395
3.3	Lancement de k3s	395
3.4	Communication avec le cluster	395

Chapitre 14

Exposition des applications sur Internet

1.	Objectifs du chapitre et prérequis	397
2.	Gestion des entrées DNS	398
2.1	Principe de fonctionnement	398
2.2	Prérequis	399
2.3	Retour sur le fonctionnement du domaine DNS nip.io	399
2.4	Activation du service DNS	400
2.4.1	Console Cloud DNS Google	400
2.4.2	Service DNS Zones d'Azure	403
2.4.3	Service Route 53 d'AWS	404
2.5	Configuration DNS	405
2.5.1	Opération à réaliser	405
2.5.2	Console DNS OVH	405
2.5.3	Vérification de la délégation	407
2.6	Gestionnaire de DNS	408
2.6.1	Présentation de la brique external-dns	408
2.6.2	Création du compte d'administration DNS de Google Cloud	408
2.6.3	Création du compte d'administration DNS service Azure	410

- 2.6.4 Création du compte d'administration
DNS Amazon (Route 53)..... 411
 - 2.6.5 Création d'un secret (Google et Azure)..... 414
 - 2.6.6 Déploiement d'external-dns..... 415
 - 2.6.7 Vérification du fonctionnement d'external-dns..... 416
 - 3. Exposition de services et répartition de charge..... 417
 - 3.1 Présentation du mécanisme..... 417
 - 3.2 Retour sur la notion de service..... 417
 - 3.2.1 Rôle d'un service..... 417
 - 3.2.2 Structure d'un service..... 418
 - 3.2.3 Type ClusterIP..... 418
 - 3.2.4 Type NodePort et LoadBalancer..... 419
 - 3.2.5 Type ExternalName..... 419
 - 3.3 Service associé au proxy inverse..... 419
 - 4. Le contrôleur Ingress..... 420
 - 4.1 Principe de fonctionnement..... 420
 - 4.2 Le rôle du contrôleur Ingress..... 421
 - 4.3 Structure d'une règle Ingress..... 421
 - 4.4 Droits nécessaires pour un contrôleur..... 422
 - 5. Le contrôleur Ingress Google..... 423
 - 5.1 Prérequis..... 423
 - 5.2 Présentation du contrôleur GLBC..... 423
 - 5.3 Déploiement de MailHog..... 424
 - 5.3.1 Préparation de la règle Ingress..... 424
 - 5.3.2 Déploiement de l'application MailHog..... 425
 - 5.3.3 Consultation de l'état de l'objet Ingress..... 425
 - 5.3.4 Consultation du journal d'activité d'external-dns..... 425
 - 5.3.5 Consultation de MailHog..... 427
 - 6. Le contrôleur Ingress Nginx..... 429
 - 6.1 Pourquoi changer de contrôleur Ingress ?..... 429
 - 6.2 Présentation du logiciel Nginx..... 430
 - 6.3 Installation d'Ingress Nginx sur GKE (Google)..... 430
 - 6.3.1 Détermination du chart Helm à installer..... 430

6.3.2	Espace de noms et configuration du chart	431
6.4	Utilisation du contrôleur	432
6.4.1	Utilisation de l'annotation kubernetes.io/ingress.class	432
6.4.2	Vérification du déploiement	433
6.5	Annotations Ingress Nginx	434
7.	Le contrôleur Ingress Traefik	435
7.1	Présentation de Traefik	435
7.2	Installation du chart Helm	435
7.3	Utilisation du nouveau contrôleur Ingress	437
7.3.1	Sélectionner le contrôleur Ingress Traefik	437
7.3.2	Création de la règle Ingress faisant appel à Traefik	437
7.3.3	État des règles Ingress	438
7.4	Tableau de bord de Traefik	439
7.5	Annotations Ingress Traefik	440

Chapitre 15

Sécurisation : accès aux applications

1.	Objectifs du chapitre et prérequis	441
2.	Mise en place de Let's Encrypt	442
2.1	Présentation de Let's Encrypt	442
2.2	Installation du chart Helm cert-manager	442
2.2.1	Présentation du chart Helm	442
2.2.2	Prérequis avant installation	443
2.3	L'émetteur de certificats (issuer)	444
2.3.1	Principe du protocole ACME	444
2.3.2	Structure de la déclaration d'un émetteur	445
2.3.3	Exemple de déclaration Issuer Google	446
2.3.4	Exemple de déclaration Issuer Azure	447
2.3.5	Exemple de déclaration Issuer Amazon	447
2.3.6	Limitations et certificats avec joker	448
2.4	Exemples de déclarations	449
2.4.1	Serveur Let's Encrypt de test	449

2.4.2	Serveur Let's Encrypt de production	450
2.5	Déclaration des certificats	450
2.5.1	État des émetteurs de certificats	450
2.5.2	Structure d'un certificat	451
2.5.3	Certificat de test	452
2.5.4	État du certificat	452
2.5.5	Journal d'activité de cert-manager	453
2.5.6	Consultation du secret	455
2.5.7	Certificat de production	456
2.5.8	Marche à suivre en cas de problèmes	458
2.6	Rattachement du certificat à la règle Ingress	458
2.7	Automatisation de la gestion des certificats	460
2.7.1	Certificat par défaut du contrôleur Ingress Nginx	460
2.7.2	Mécanisme d'annotations	461
2.7.3	Émetteur de certificats par défaut	462
3.	Protection de l'accès aux applications	463
3.1	Origine du besoin	463
3.2	Mot de passe simple (HTTP basic)	463
3.2.1	Principe de fonctionnement	463
3.2.2	Création du secret à l'aide de htpasswd	463
3.2.3	Import du secret	464
3.2.4	Configuration de l'authentification	464
4.	Authentification basée sur OAuth2	467
4.1	À propos du protocole OAuth2	467
4.2	Principe de la solution	467
4.3	Création d'un identifiant GitHub	468
4.4	Déploiement du proxy	469
4.4.1	À propos du proxy	469
4.4.2	Configuration du chart Helm	470
4.4.3	Déploiement du chart Helm	471
4.4.4	État du déploiement	471
4.5	Déclaration des règles Ingress	471
4.5.1	Description des règles Ingress	471

4.5.2	Annotations Ingress de MailHog	472
4.5.3	Description Ingress du proxy OAuth	472
4.5.4	Déclaration des règles Ingress	472
4.6	Tests de connexion	474
4.7	Restriction des accès	475
4.7.1	Mécanisme d'autorisation	475
4.7.2	Restriction par domaine e-mail	476
4.7.3	Restriction par organisation GitHub	476

Chapitre 16

Polices réseau

1.	Objectifs du chapitre et prérequis	479
2.	Les polices réseau (network policies)	479
2.1	Présentation du mécanisme	479
2.2	Kubernetes Network Plugins	480
2.3	Polices réseau sur services managés et installation maison	480
2.4	Installation de Calico sur Minikube	481
2.4.1	Activation de CNI sur Minikube	481
2.4.2	Installation de Calico	481
2.5	Connexions entrantes	482
2.5.1	Test de la connexion en interne	482
2.5.2	Bloquer les accès internes	482
2.5.3	Test de la règle	484
2.6	Connexions sortantes	484
2.6.1	Test des connexions externes	484
2.6.2	Restriction sur les règles sortantes	485
2.6.3	Test de la règle	485
3.	Protection de l'application WordPress	486
3.1	Contexte	486
3.2	Déploiement de WordPress	486
3.3	Restriction des accès	487
3.3.1	Référencement de l'ensemble des flux	487

- 3.3.2 Restriction de tous les accès 487
- 3.3.3 Autorisation de l'accès du contrôleur Ingress
sur WordPress 488
- 3.3.4 Autorisation de l'accès entre WordPress et MariaDB . . . 489
- 3.3.5 Test des règles réseau 491
- 3.4 Ressources externes 492

Chapitre 17

Montée en charge automatique

- 1. Objectifs du chapitre et prérequis 493
- 2. Le serveur de métriques 494
 - 2.1 Présentation de la brique metrics-server 494
 - 2.2 Activation du serveur de métriques 494
 - 2.2.1 Vérification de la présence du serveur de métriques . . . 494
 - 2.2.2 Activation sous Minikube 495
 - 2.2.3 Activation sur Amazon EKS 495
 - 2.3 Consultation de la consommation des pods et des nœuds . . . 496
 - 2.3.1 État des nœuds d'un cluster 496
 - 2.3.2 État des pods 496
- 3. Activation de la montée en charge automatique 497
 - 3.1 Test avec l'application MailHog 497
 - 3.2 Lancement d'un bench 498
 - 3.2.1 Présentation d'Apache Bench 498
 - 3.2.2 Installation d'Apache Bench 498
 - 3.2.3 Lancement du test initial 499
 - 3.3 Gestion de la montée en charge 499
 - 3.4 Lancement du test 501
 - 3.4.1 Test à froid : montée en charge des pods 501
 - 3.4.2 Test à chaud 501
 - 3.4.3 Diminution du nombre de pods 502
- 4. Scalabilité des nœuds d'un cluster 502
 - 4.1 Contexte 502

4.2	Présentation de l'autoscaler	503
4.3	Activation de l'autoscaler avec Google Cloud	503
4.4	Activation de l'autoscaler avec Azure AKS	504
4.5	Activation de l'autoscaler avec Amazon EKS	506
4.5.1	Présentation du mécanisme de l'ASG	506
4.5.2	Affichage des tags d'un groupe ASG	506
4.5.3	Activation de l'autoscaler.	507
4.5.4	Vérification de l'activation du mécanisme de l'autoscaler	508
4.6	Test de montée en charge	508
4.6.1	Déploiement de WordPress	508
4.6.2	Pods en attente de ressources.	509
4.6.3	État des nœuds	510
4.6.4	Démarrage du pod	510
4.6.5	Nettoyage des déploiements	511

Chapitre 18

Surveillance à l'aide de Prometheus

1.	Objectifs du chapitre et prérequis	513
2.	Mise en place de Prometheus.	514
2.1	À propos de Prometheus	514
2.2	Fonctionnement de Prometheus	515
2.2.1	Architecture de Prometheus	515
2.2.2	Le moteur Prometheus	515
2.2.3	Les exporteurs Prometheus	516
2.3	Installation de Prometheus	517
2.3.1	Choix du chart Prometheus	517
2.3.2	Qu'est-ce qu'un opérateur ?	517
2.3.3	Déploiement de l'opérateur Prometheus.	518
2.3.4	Pods démarrés	519
2.3.5	Objets déploiements.	520
2.3.6	Nouvelles ressources Prometheus	521

2.3.7	DaemonSet : node exporter	522
2.4	Priorisation des briques de surveillance	522
2.4.1	Problème de la surveillance	522
2.4.2	Déclaration des classes de priorité	522
2.4.3	Modification du déploiement de Prometheus	523
3.	Utilisation de Prometheus	524
3.1	Fonctionnement des métriques	524
3.1.1	Consultation des métriques de Prometheus	524
3.1.2	Présentation de l'interface de Prometheus	525
3.1.3	Métriques de Kubernetes	526
3.1.4	Déclaration des points de collecte dans Kubernetes	527
3.1.5	Consultation des points de collecte dans Prometheus	528
3.2	Définition des alertes	529
3.2.1	Consultation de la liste des alertes	529
3.2.2	Structure d'une règle d'alerte	530
3.2.3	Définition d'alertes	531
3.3	Gestionnaire d'alertes	532
3.3.1	Rôle du gestionnaire d'alertes	532
3.3.2	Consultation du gestionnaire d'alertes	533
3.3.3	Configuration des alertes	533
3.3.4	Désactivation des alertes scheduler et manager (clusters managés)	534
3.3.5	Configuration de l'envoi des notifications	536
3.3.6	Adresse de l'API de Slack	537
4.	Tableaux de bord Grafana	540
4.1	Présentation de Grafana	540
4.2	Configuration de Grafana	540
4.2.1	Branchement au moteur Prometheus	540
4.2.2	Définition des tableaux de bord	541
4.3	Interface Grafana	542
4.4	Sécurisation de l'accès à Grafana	543
5.	Suppression du chart de Prometheus	544

Chapitre 19**Centralisation des journaux d'activité**

1. Objectifs du chapitre et prérequis	545
2. Principe de la centralisation des journaux.	546
2.1 Architecture	546
2.2 Caractéristiques de l'agent déployé.	547
2.3 Mécanisme de centralisation des journaux.	547
3. Centralisation dans le cloud.	548
3.1 Centralisation à l'aide d'un service managé	548
3.2 Google Stackdriver	548
3.2.1 Présentation de Stackdriver	548
3.2.2 Pod Fluentd (cluster GKE)	549
3.2.3 Consultation des journaux	549
3.3 Azure Monitor	550
3.3.1 Présentation d'Azure Monitor	550
3.3.2 Consultation des journaux	550
3.4 Amazon Cloudwatch	551
3.4.1 Présentation de Cloudwatch	551
3.4.2 Activation de Cloudwatch sur le centre de contrôle	551
3.4.3 Configuration de Cloudwatch.	553
3.4.4 Création de la police de communication avec Cloudwatch	553
3.4.5 Création d'un compte et rattachement à la police.	555
3.4.6 Création d'une clé d'accès	555
3.4.7 Envoi des journaux dans Cloudwatch	556
3.4.8 État des pods déployés	557
3.4.9 Consultation des journaux dans Cloudwatch	558
4. Centralisation des journaux avec Loki	559
4.1 Présentation de Loki	559
4.1.1 Origine de Loki	559
4.1.2 Loki vs Elasticsearch.	560
4.1.3 Conseil d'utilisation	560

4.2	Installation de Loki	560
4.3	Configuration de la source de données Grafana	561
4.4	Consultation des journaux dans Grafana	563
4.4.1	Vérification des sources de données	563
4.4.2	Consultation des journaux	563
5.	Centralisation des journaux avec Elasticsearch	565
5.1	Avertissements et limitations	565
5.2	À propos d'Elasticsearch.	565
5.3	Déploiement des briques Elasticsearch	565
5.3.1	Installation d'Elasticsearch	565
5.3.2	Installation de l'agent fluent-bit	567
5.3.3	Installation de Kibana	568
5.3.4	Installation de Cerebro.	568
5.4	État des différentes briques	569
5.4.1	État du moteur Elasticsearch	569
5.4.2	Agent fluent-bit	569
6.	Gestion d'Elasticsearch.	570
6.1	Accès à l'interface Cerebro.	570
6.2	Utilisation de Kibana	572
6.2.1	Accéder à l'interface de Kibana	572
6.2.2	Création de l'index	573
6.3	Branchement sur Grafana	575
6.3.1	Source de données Elasticsearch	575
6.3.2	Création d'un objet ConfigMap	575

Chapitre 20

Maillage de services avec Istio

1.	Objectifs du chapitre et prérequis	577
2.	Présentation d'Istio	578
2.1	Micro-services et mise en réseau de services	578
2.2	Présentation d'Istio.	579
2.3	Principe de fonctionnement.	579

3.	Installation d'Istio	581
3.1	Configuration d'external-dns	581
3.2	Dépôt des charts Istio	582
3.3	Installation du chart istio-init	583
3.4	Installation du chart Istio	583
3.4.1	Activation des tableaux de bord	583
3.4.2	Activation du mécanisme SDS	585
3.4.3	Configuration d'Istio	585
3.4.4	Installation d'Istio	587
3.5	Configuration du gateway Istio	588
3.5.1	Présentation du mécanisme	588
3.5.2	Configuration du gateway	588
3.5.3	Génération du certificat	589
4.	Utilisation d'Istio	590
4.1	Injection de pods dans le maillage de services	590
4.1.1	Installation d'istiocli	591
4.1.2	Injection du sidecar à l'aide d'istiocli	591
4.1.3	Injection du sidecar par annotation	592
4.1.4	Désactivation du sidecar par annotation	592
4.2	Déploiement d'une application de test	593
4.2.1	Principe de l'exposition d'application avec Istio	593
4.2.2	Préparation du fichier de déploiement MailHog	593
4.2.3	Déclaration du service MailHog	595
4.2.4	Création du gateway	596
4.2.5	Création du service virtuel (VirtualService)	597
4.3	Sécurisation des flux	599
4.3.1	Activation de Mutual TLS (mTLS)	599
4.3.2	Consultation des polices du service MailHog	600
4.3.3	Forcer l'utilisation de mTLS	600
5.	Tableaux de bord	603
5.1	Présentation des différentes briques	603
5.2	Interface Kiali	604
5.3	Interface Grafana	605

5.4 Interface Jaeger 607

Chapitre 21

Compilation et stockage d'image Docker

1. Objectifs du chapitre et prérequis 609

2. Création d'une image Docker 610

 2.1 Application d'exemple Flask Healthcheck 610

 2.1.1 Présentation de l'application 610

 2.1.2 Présentation des dépendances 610

 2.1.3 Description des dépendances 610

 2.1.4 Installation des dépendances 611

 2.1.5 Initialisation de l'application 611

 2.1.6 Fonction racine 612

 2.1.7 Lancement du programme 612

 2.2 Environnement de compilation 615

 2.3 Le fichier Dockerfile 616

 2.3.1 Présentation du format 616

 2.3.2 Création de l'image 616

 2.3.3 Compilation et tag de l'image 617

 2.3.4 Authentification sur le registre d'images Docker 618

 2.3.5 Pousser l'image sur le registre 619

3. Image Docker multi-étape (multi-stage) 620

 3.1 Origine du besoin 620

 3.2 Exemple de compilation avec Maven 621

4. Analyse d'images 621

 4.1 Historique des commandes 621

 4.2 Analyse de l'image : Dive 622

 4.2.1 Présentation de Dive 622

 4.2.2 Installation de Dive 622

 4.2.3 Consultation du contenu d'une image 622

5. Utilisation de registres Docker privés 623

 5.1 Origine du besoin 623

5.2	Déploiement d'image d'un registre privé	624
5.2.1	Exposition de la problématique	624
5.2.2	Création du secret	624
5.2.3	Utilisation du secret	625
5.2.4	Recopie d'un secret entre espaces de noms	626
5.3	Erreurs de récupération des images	627
5.3.1	Détection des erreurs	627
5.3.2	Erreur sur le nom de l'image	627
5.3.3	Secret absent ou non spécifié	627
5.3.4	Identifiants invalides	628
5.4	Services de registres managés	628
5.4.1	Solutions managées	628
5.4.2	Service Google Container Registry	628
5.4.3	GitLab.com	630

Chapitre 22

Usine logicielle

1.	Objectifs du chapitre et prérequis	633
2.	Compilation à l'aide de GitLab	634
2.1	Application à compiler	634
2.2	Mécanisme de pipeline GitLab	634
2.3	Adresse et contenu du dépôt	634
2.4	Structure du fichier <code>.gitlab-ci.yml</code>	635
2.5	Exemple de fichier <code>.gitlab-ci.yml</code> de compilation d'image	636
2.6	Pour la suite	638
3.	Déploiement continu avec Jenkins	639
3.1	À propos de Jenkins	639
3.2	Installation de Jenkins	639
3.2.1	Configuration du chart	639
3.2.2	Vérification de l'installation	640
3.2.3	Connexion à l'interface de Jenkins	640
3.2.4	Installation d'extensions	641

4. Pipeline de déploiement continu avec Jenkins	642
4.1 Prérequis	642
4.2 Présentation du mécanisme de déploiement continu	642
4.3 Stockage des identifiants Docker.	643
4.4 Création de l'environnement develop	643
4.5 Création du pipeline	643
4.5.1 Création du pod de compilation	643
4.5.2 Squelette du pipeline de déploiement	646
4.5.3 Récupération du code source	647
4.5.4 Vérifications et tests.	647
4.5.5 Compilation de l'image Docker.	648
4.5.6 Connexion au registre et publication	648
4.5.7 Mise à jour du déploiement de test	649
4.5.8 Programme complet	650
4.6 Lancement de la compilation.	652
4.6.1 Création du job	652
4.6.2 Lancement du build	653
4.7 Compte de service.	654
4.7.1 Opérations à réaliser.	654
4.7.2 Création d'un compte de service	654
4.7.3 Création du rôle de mise à jour	654
4.7.4 Affectation du rôle au compte de service	655
4.7.5 Affectation du compte de service	656
4.7.6 Relance de la compilation	656
4.8 Mécanisme de Webhook	657
4.8.1 Présentation du mécanisme	657
4.8.2 Déclenchement du Webhook.	657
4.8.3 Création du Webhook	658
5. Un mot sur Jenkins X.	659

Chapitre 23**Packager son application avec Helm**

1. Objectifs du chapitre et prérequis	661
2. Helm	661
2.1 Origine du besoin	661
2.2 Création d'un chart	662
2.3 Contenu d'un package	662
2.3.1 Structure d'un package	662
2.3.2 Variables .Values, .Chart et .Release	663
2.4 Adaptation à l'application MailHog	663
2.4.1 Résumé des travaux à réaliser	663
2.4.2 Utilisation de l'image de MailHog	664
2.4.3 Correction sur les ports de service	664
2.4.4 Ajout d'un ConfigMap	665
2.4.5 Ajout du volume persistant	666
2.4.6 Montage du volume dans le container	667
2.4.7 Test de déploiement	668
2.5 Dépendances	669
2.5.1 Déclaration des dépendances	669
2.5.2 Récupération des dépendances	670
2.5.3 Déploiement du chart avec les dépendances	671
3. Template Go	671
3.1 Principe de fonctionnement	671
3.2 Substitution du contenu d'une variable	672
3.3 Blocs conditionnels	672
3.4 Gestion des conditions	673
3.4.1 Les valeurs de vrai ou faux	673
3.4.2 Opérateurs de comparaison	673
3.4.3 Conditions multiples et négation	674
3.5 Itération	674
3.5.1 Affichage du contenu d'un tableau	674
3.5.2 Suppression des sauts de lignes surnuméraires	675
3.5.3 Itération sur un groupe fixe d'éléments	676

3.5.4	Itération sur un tableau de hachage	676
3.5.5	Accéder à une variable globale depuis une boucle	677
3.6	Filtres	677
3.7	Valeurs par défaut	678
3.8	Fonction template	678
3.8.1	Exemple de définition de fonction	678
3.8.2	Passage de paramètres et contexte global	679
3.9	Redéploiement sur changement de configuration	680
3.9.1	Exposition de la problématique	680
3.9.2	Principe de la solution	680
3.9.3	Exemple d'implémentation	680

Chapitre 24

Restriction et délégation d'accès

1.	Objectifs du chapitre et prérequis	683
2.	Mise en place de quotas	684
2.1	Origine du besoin	684
2.2	Quotas par défaut sur un espace de noms	684
2.2.1	Création d'un espace de noms	684
2.2.2	Structure d'un objet LimitRange	685
2.2.3	Exemple de définition de limitations	686
2.2.4	Vérification de l'application des limitations	687
2.2.5	Test du mécanisme	688
2.2.6	Analyse du problème	688
2.3	Quotas globaux sur un espace de noms	689
2.3.1	Présentation des quotas de ressources (ResourceQuota)	689
2.3.2	Structure d'un quota de ressources	690
2.3.3	Exemples de restriction de consommation CPU et mémoire	690
2.3.4	Champs pour positionner des limitations (champ hard)	691
2.3.5	Test du mécanisme	692

2.3.6	Nettoyage en fin d'exercice	693
3.	Authentification et autorisation	694
3.1	Origine du besoin	694
3.2	Prérequis	695
3.3	Activation des accès anonymes	695
3.3.1	Activation des accès anonymes sur Minikube	695
3.3.2	Création du fichier d'accès.	696
3.3.3	Affectation des droits en lecture	697
3.3.4	Suppression des droits d'accès anonymes.	698
3.4	Principe de l'authentification par certificat	698
3.5	Problème de la révocation des certificats	699
3.6	Génération du certificat	699
3.6.1	Création de la clé et du certificat client	699
3.6.2	Emplacement de la PKI	700
3.6.3	Signature du certificat	702
3.7	Authentification par certificat.	703
3.7.1	Récupération des informations de connexion au cluster	703
3.7.2	Utilisation du certificat pour l'authentification.	704
3.7.3	Test de la connexion	705
3.8	Quelques exemples d'erreurs de manipulation.	706
3.8.1	Problème d'autorité de certification	706
3.8.2	Problème de couple clé/certificat.	707
3.8.3	Pas d'identifiant renseigné.	707
3.8.4	Utilisation de la demande de certificat à la place du certificat.	707
3.9	Attributions de droits administrateur sur le cluster.	707
3.9.1	Contexte et opérations à réaliser.	707
3.9.2	Affectation des droits administrateur à un utilisateur	708
3.9.3	Test du nouvel administrateur	709
3.9.4	Création de nouveaux utilisateurs	710
3.9.5	Affectation des droits administrateur à un groupe	711
3.9.6	Administrateur d'un espace de noms	711

4. Mécanismes d'authentification externes	713
4.1 Présentation du mécanisme	713
4.2 Communication entre le fournisseur OAuth2 et le cluster	714
4.3 Création des identifiants	715
4.4 Modification des options de démarrage (Minikube)	716
4.5 Configuration des accès clients	718
4.5.1 Présentation de l'outil k8s-oidc-helper	718
4.5.2 Installation du compilateur Go	718
4.5.3 Installation de k8s-oidc-helper	718
4.5.4 Génération des identifiants d'accès	718
4.5.5 Renseignement du cluster et contexte	720
4.5.6 Test de connexion	721
4.6 Attribution des droits d'accès	722

Chapitre 25

Les opérateurs Kubernetes

1. Objectifs du chapitre et prérequis	723
2. Utilisation des opérateurs	724
2.1 Présentation du principe	724
2.2 L'opérateur de Prometheus	724
2.2.1 Retour sur le chart prometheus-operator	724
2.2.2 Structure d'un objet Prometheus	725
2.3 Ressources externes sur les opérateurs existants	727
2.4 Présentation de l'opérateur MySQL	727
2.4.1 Justification du choix de MySQL	727
2.4.2 Choix de l'opérateur	728
2.5 Déploiement de l'opérateur	728
2.6 Création d'une instance MysqlCluster	729
2.6.1 Création d'un secret pour l'instance MysqlCluster	729
2.6.2 Création du cluster	730

2.7	Objets créés au déploiement du cluster	731
2.7.1	Volumes persistants	731
2.7.2	Services MySQL	732
2.8	Tableau de bord de l'opérateur	732
2.9	Test de la réplication	735
2.9.1	Connexion aux instances maître et esclave	735
2.9.2	Création d'une table	736
2.9.3	Alimentation de la table	736
2.9.4	Changement du nombre de réplicats	737
2.10	Pour conclure	738

Annexes

1.	bash vs zsh	739
1.1	Les shells Unix	739
1.2	zsh et oh-my-zsh	740
1.3	Tester zsh et oh-my-zsh	740
1.4	Configurer zsh comme shell pour l'utilisateur courant	741
2.	Déploiement du tableau de bord Kubernetes	742
2.1	Installation du tableau de bord (application dashboard)	742
2.2	Création du compte d'accès	742
2.3	Récupération du jeton de connexion	743
2.4	Décodage du contenu du jeton	744
2.5	Connexion au tableau de bord	746

Index	747
-------	-----

Les éléments à télécharger sont disponibles à l'adresse suivante :
<http://www.editions-eni.fr>
Saisissez la référence de l'ouvrage **EPKUBCLU** dans la zone de recherche et validez. Cliquez sur le titre du livre puis sur le bouton de téléchargement.

Chapitre 1 Fonctionnement de Kubernetes

- 1. Contenu du livre 9
 - 1.1 Objectifs 9
 - 1.2 Prérequis de lecture 10
 - 1.3 Avertissement sur les versions de Kubernetes 10
 - 1.4 Kubernetes et Azure 11
- 2. À quoi sert Kubernetes ? 12
 - 2.1 Les besoins en déploiement applicatif 12
 - 2.1.1 Montée en charge par répartition 12
 - 2.1.2 Mises à jour progressives 13
 - 2.1.3 Composition applicative robuste 13
 - 2.2 Structuration du système d'information 14
 - 2.2.1 Principes 14
 - 2.2.2 Découplage obtenu par Kubernetes 16
- 3. Historique de Kubernetes 18
 - 3.1 Aux origines : Google Borg 18
 - 3.2 Transfert à la Cloud Native Computing Foundation 19
 - 3.3 Et la concurrence ? 19
- 4. Principes d'architecture de Kubernetes 21
 - 4.1 Mise en cluster 21
 - 4.1.1 Principes 21
 - 4.1.2 Les masters et leurs composants 22
 - 4.1.3 Les nodes et leurs composants 25
 - 4.2 Concepts liés à Kubernetes 26
 - 4.2.1 Pods 27
 - 4.2.2 Namespaces 28

2 **Kubernetes**

Mise en œuvre d'un cluster et déploiement de microservices

4.2.3	Services	29
4.2.4	IngressController	32
4.2.5	Volumes	37
4.2.6	PersistentVolume et PersistentVolumeClaim	39
4.2.7	ConfigMap	41
4.2.8	Secrets	45
4.2.9	Deployment	48
4.2.10	ReplicaSet	50
4.2.11	DaemonSet	52
4.2.12	Résumé des concepts	53
4.3	Role Based Access Control	54
4.3.1	Principe	54
4.3.2	Role	56
4.3.3	Cluster Role	56
4.3.4	Role Binding	57
4.3.5	Cluster Role Binding	59
4.4	Notion de réseau	59
4.5	Kubernetes et Docker	60

Chapitre 2

Création et gestion d'un cluster Kubernetes

1.	Méthodes d'installation d'une plateforme Kubernetes	61
1.1	Plusieurs façons de faire	61
1.2	Environnement de développement	62
1.3	Environnement on premise	63
1.4	Environnement cloud	64
2.	Installation d'une plateforme Kubernetes	66
2.1	Généralités et préparation	66
2.1.1	Description du mode stacked	67
2.1.2	Description du mode external	69
2.1.3	Prérequis techniques	71

2.2	Installation avec Kubeadm.	72
2.2.1	Initialisation des machines proxy.	72
2.2.2	Configuration logicielle des machines proxy.	75
2.2.3	Configuration de Keepalived	75
2.2.4	Validation du fonctionnement de Keepalived	79
2.2.5	Configuration de HAProxy	80
2.2.6	Initialisation des machines master	83
2.2.7	Configuration des machines master.	85
2.2.8	Configuration de Docker	86
2.2.9	Installation des paquets pour Kubernetes	89
2.2.10	Préparation de la configuration réseau.	90
2.2.11	Installation du cluster	92
2.2.12	Gestion des certificats	95
2.2.13	Jointure des autres serveurs master	97
2.2.14	Option de récupération automatique des certificats.	98
2.2.15	Vérification de fonctionnement du cluster	99
2.2.16	Déploiement applicatif pour validation	101
2.2.17	Suppression du cluster	110
2.3	Installation avec Kubespray.	110
2.3.1	Description de l'outil	110
2.3.2	Opérations préparatoires.	111
2.3.3	Création éventuelle des machines proxy	112
2.3.4	Création des machines master	114
2.3.5	Paramétrage divers.	116
2.3.6	Installation du cluster	118
2.3.7	Vérification de fonctionnement du cluster	119
3.	Mise en œuvre d'un cluster Kubernetes	123
3.1	Connexion au cluster	123
3.1.1	Les utilisateurs	123
3.1.2	Modes de connexion	124
3.1.3	Le fichier KUBECONFIG	125
3.1.4	Gestion des contextes	125
3.1.5	Ajout du contexte pour les développeurs.	127

3.1.6	Test du contexte pour les développeurs	131
3.1.7	Ajout du contexte pour les administrateurs	132
3.1.8	Test du contexte pour les administrateurs	133
3.1.9	Ajustement des autorisations pour les administrateurs	135
3.1.10	Ajustement des autorisations pour les développeurs . .	137
3.2	Exploitation du cluster	141
3.2.1	Binaire kubectl	141
3.2.2	Commandes de base	142
3.2.3	Commandes de déploiement	145
3.2.4	Commandes de gestion	146
3.2.5	Commandes de débogage	147
3.2.6	Commandes de paramétrage	149
3.3	Exposition des applications	150
3.3.1	Installation de l'Ingress Controller Nginx	150
3.3.2	Paramétrage d'une Ingress Rule	153
3.3.3	Validation du fonctionnement	157
3.3.4	Mise en place de la haute disponibilité	157
3.3.5	Passage à l'échelle de l'application	158
3.3.6	Gestion de versions	160
4.	Maintien en condition opérationnelle d'un cluster Kubernetes . . .	164
4.1	Surveillance de l'écosystème Kubernetes	164
4.1.1	Surveillance du cluster	164
4.1.2	Surveillance des pods	165
4.1.3	Présentation des outils de supervision	166
4.1.4	Architecture de supervision proposée	167
4.1.5	Mise en place de l'architecture	168
4.1.6	Paramétrage d'un tableau de bord	172
4.1.7	Utilisation des outils déployés	173
4.2	Filtrage réseau avec les NetworkPolicies	175
4.2.1	Présentation des NetworkPolicies	175
4.2.2	Exemple de NetworkPolicy	176
4.2.3	NetworkPolicies standards	178
4.2.4	Mise en pratique des NetworkPolicies	179

- 4.3 Quelques bonnes pratiques 182
 - 4.3.1 Recommandations sur les créations
de ressources Kubernetes. 182
 - 4.3.2 Recommandations sur les infrastructures cloud 183
 - 4.3.3 Stratégie de téléchargement des images. 184

Chapitre 3
Déploiement d'applications avec Kubernetes

- 1. Contexte 187
 - 1.1 Objectifs généraux 187
 - 1.1.1 Exploiter la plateforme Kubernetes mise en place. 187
 - 1.1.2 Remarque sur l'approche DevOps 188
 - 1.1.3 Principaux enseignements à attendre. 190
 - 1.2 Outillage 191
 - 1.2.1 Cluster Azure Kubernetes Services. 191
 - 1.2.2 Kubectl 207
 - 1.2.3 Azure CLI 207
 - 1.2.4 Paramétrage de kubectl 209
 - 1.2.5 Docker pour Windows 210
- 2. Premier déploiement par ligne de commande 212
 - 2.1 Contexte 212
 - 2.2 Création du déploiement 213
 - 2.3 Vérification du déploiement 213
 - 2.4 Présence d'un pod 214
 - 2.5 Exposition par un service 215
 - 2.6 Test de l'application 216
 - 2.7 Nettoyage 217
- 3. Second déploiement à l'aide d'un fichier de configuration 218
 - 3.1 Objectifs 218
 - 3.2 Application exemple 219
 - 3.2.1 Présentation de l'application 219
 - 3.2.2 Limites logicielles 220

6 **Kubernetes**

Mise en œuvre d'un cluster et déploiement de microservices

3.2.3	Pourquoi des microservices ?	221
3.2.4	API	222
3.2.5	Scénarios d'utilisation	223
3.3	Développement logiciel	228
3.3.1	Code source et paramétrage	228
3.3.2	Intégration continue	232
3.4	Déploiement de l'application	238
3.4.1	Structure du projet de déploiement	239
3.4.2	Namespace	240
3.4.3	Fichier de paramétrage	241
3.4.4	Fichiers liés aux bases de données	243
3.4.5	Fichiers liés aux services	246
3.4.6	Définition de l'Ingress	249
3.4.7	Lancement d'un seul coup	254
3.5	Vérification du déploiement	255
3.6	Paramétrage de l'application	258
3.6.1	Avec Postman	258
3.6.2	Avec curl	267
3.6.3	Modification des variables d'environnement	267
3.7	Test de l'application	269
3.8	Supervision du cluster	270
3.8.1	Azure Monitor Insights	271
3.8.2	Azure Monitor logs	275
3.8.3	Kubernetes dashboard	276
3.8.4	Azure Monitor	284
3.8.5	Autres méthodes de monitoring	286
3.9	Scalabilité	287
3.9.1	Mode statique	287
3.9.2	Mode dynamique	291
3.10	Fonctionnement du déploiement continu	297
3.11	Mise à jour des versions applicatives	306
3.11.1	Commande de mise à jour	306
3.11.2	Fonctionnement en rolling update	308

- 3.11.3 Amélioration du fonctionnement par les sondes 312
- 3.11.4 Historique de mises à jour. 318
- 3.12 Gestion de la persistance 321
 - 3.12.1 Kubernetes et les volumes. 321
 - 3.12.2 Utilisation d'un disque Azure 322
 - 3.12.3 Remarque sur les bases de données en Kubernetes 336
 - 3.12.4 Sauvegarde de bases de données 337
- 3.13 Utilisation de Secrets 349
- 3.14 Suppression de l'application. 356
 - 3.14.1 Suppression dans Kubernetes 356
 - 3.14.2 Cas particulier sur l'ingress 358
 - 3.14.3 Suppression des ressources Azure. 360

- Index 361