

**RBER**

# Architecture RKE2 - Universités du Bénin

Infrastructure Cloud-Native pour  
l'Enseignement à Distance

PREPARE POUR  
**SBIN**

PREPARE PAR  
**IDADU TECH**



Connecting business need with IT



## Table des matières

Table des illustrations .....	8
1. Contexte du projet .....	10
1.1 Genèse.....	10
1.2 Universités concernées .....	10
1.3 Limites de l'infrastructure existante .....	10
2. Mission et objectifs.....	10
2.1 Mission .....	10
2.2 Objectifs quantitatifs.....	11
3. Périmètre du projet.....	11
3.1 Applications hébergées .....	11
3.2 Sites géographiques .....	11
4. Principes architecturaux.....	11
4.1 Cloud-Native .....	11
4.2 Haute disponibilité.....	12
4.3 Sécurité intégrée .....	12
5. Stack technologique .....	12
6. Organisation du document .....	13
7. Points clés du chapitre .....	13
1. Approche par plans .....	14
1.1 Pourquoi raisonner en plans .....	14
1.2 Principe directeur .....	14
1.3 Vue d'ensemble des plans .....	14
2. Plan de contrôle développeur .....	15
2.1 Objectif .....	15
2.2 Outils disponibles .....	15
2.3 Flux de travail développeur .....	15
3. Plan d'intégration et de livraison .....	15
3.1 Objectif .....	15
3.2 Composants CI/CD.....	15
3.3 Pipeline type .....	16
4. Plan de surveillance et logging .....	16
4.1 Objectif .....	16
4.2 Stack d'observabilité.....	16
4.3 Dashboards clés.....	16
5. Plan de sécurité .....	16
5.1 Objectif .....	16

5.2 Composants de sécurité .....	17
6. Plan de ressources .....	17
6.1 Objectif .....	17
6.2 Ressources par site .....	17
6.3 Services de données.....	17
7. Points clés du chapitre .....	17
1. Vue d'ensemble.....	19
1.1 Sites de déploiement .....	19
1.2 Architecture type d'un cluster.....	19
2. Spécifications serveurs .....	20
2.1 Configuration serveur .....	20
2.2 Capacité totale par site .....	21
2.3 Capacité totale infrastructure (2 sites) .....	21
3. Architecture réseau .....	21
3.1 Topologie réseau.....	21
3.2 Segmentation VLAN.....	21
3.3 Plan d'adressage par site .....	22
4. Connectivité inter-sites .....	22
4.1 Backbone RBER .....	22
4.2 Contraintes et impact architecture.....	22
5. Architecture actif-actif régional.....	22
5.1 Mode de fonctionnement.....	22
5.2 Justification de l'architecture .....	23
5.3 Avantages de l'architecture .....	23
6. Équipements complémentaires .....	23
6.1 Stockage NAS.....	23
6.2 Alimentation électrique .....	23
7. Points clés du chapitre .....	23
Infrastructure Hyperconvergée Harvester .....	24
1. Vue d'ensemble de l'hyperconvergence .....	24
1.1 Définition et principes .....	24
1.2 Pourquoi Harvester pour RBER .....	24
2. Architecture technique Harvester .....	25
2.1 Composants du stack Harvester .....	25
2.2 Architecture physique par site.....	25
3. Stockage distribué Longhorn.....	25
3.1 Fonctionnement de Longhorn .....	25

3.2 Configuration stockage RBER .....	25
3.3 Capacité effective .....	26
4. Réseau virtualisé .....	26
4.1 Architecture réseau Harvester .....	26
4.2 Plan d'adressage Harvester .....	26
4.3 Configuration réseau des nœuds .....	26
5. Gestion des machines virtuelles .....	26
5.1 Création de VM .....	26
5.2 Migration live .....	27
5.3 Snapshots et backups .....	27
1. Présentation de RKE2 .....	28
1.1 Qu'est-ce que RKE2 .....	28
1.2 Pourquoi RKE2 pour RBER .....	28
2. Architecture du cluster .....	29
2.1 Topologie haute disponibilité .....	29
2.2 Justification de la séparation etcd .....	29
2.3 Plan d'adressage cluster Fidjrossè .....	30
3. Configuration RKE2 .....	30
3.1 Fichier de configuration serveur .....	30
3.2 Configuration etcd externe .....	30
3.3 Conformité CIS Benchmark .....	30
4. Networking Kubernetes .....	31
4.1 Plugin CNI Canal .....	31
4.2 Architecture réseau du cluster .....	31
4.3 Network Policies par défaut .....	31
5. Ingress et Load Balancing .....	31
5.1 Architecture HAProxy .....	31
5.2 Services exposés .....	33
6. Gestion des certificats .....	33
6.1 cert-manager .....	33
6.2 Rotation des certificats Kubernetes .....	33
7. Stockage Kubernetes .....	33
7.1 StorageClasses .....	33
7.2 CloudNativePG pour PostgreSQL .....	33
8. Exploitation et maintenance .....	34
8.1 Commandes de diagnostic .....	34
8.2 Mise à jour du cluster .....	34

8.3 Backup et restauration .....	34
9. Points clés du chapitre .....	34
6. Exploitation et maintenance .....	36
6.1 Interface de gestion .....	36
6.2 Procédures de maintenance .....	36
6.3 Mise à jour Harvester .....	36
7. Points clés du chapitre .....	36
1. Stratégie de sécurité .....	37
1.1 Principes directeurs .....	37
1.2 Périmètre de sécurité .....	37
2. Conformité aux standards .....	37
2.1 FIPS 140-2.....	37
2.2 CIS Kubernetes Benchmark .....	38
3. Gestion des identités et accès (IAM) .....	38
3.1 Architecture IAM RBER .....	38
3.2 RBAC Kubernetes .....	39
3.3 Authentification multi-facteur (MFA) .....	39
4. Chiffrement .....	39
4.1 Chiffrement en transit (TLS) .....	39
4.2 Chiffrement au repos .....	40
4.3 Gestion des secrets .....	40
5. Sécurité réseau .....	40
5.1 Segmentation réseau .....	40
5.2 Network Policies Kubernetes .....	40
5.3 Firewall et règles .....	40
6. Audit et logging .....	41
6.1 Architecture de logging .....	41
6.2 Audit Kubernetes.....	41
6.3 Alerting sécurité.....	41
7. Sécurité des conteneurs.....	42
7.1 Pod Security Admission (PSA) .....	42
7.2 Scan d'images .....	42
7.3 Bonnes pratiques images .....	42
8. Procédures de sécurité .....	42
8.1 Réponse aux incidents.....	42
8.2 Rotation des credentials.....	43
9. Points clés du chapitre .....	43



## Table des illustrations

Figure 1 - Infrastructure physique .....	19
Figure 2 - architecture hyperconvergée .....	24
Figure 3 - Architecture cluster haute disponibilité.....	29
Figure 4 - Architecture HAPROXY .....	32



Tableau 1 – Sites géographiques .....	11
Tableau 2 – Stack technologique RBER .....	12
Tableau 3 – Structure du document .....	13
Tableau 4– Vue d'ensemble des plans .....	14
Tableau 5– Outils du plan développeur .....	15
Tableau 6 – Composants du plan CI/CD .....	15
Tableau 7 – Stack d'observabilité.....	16
Tableau 8 – Composants de sécurité .....	17
Tableau 9 – Ressources par site .....	17
Tableau 10 – Services de données.....	17
Tableau 11 – Sites de déploiement .....	19
Tableau 12 – Spécifications serveur HPE DL380.....	20
Tableau 13 – Capacité totale par site .....	21
Tableau 14 – Capacité totale infrastructure .....	21
Tableau 15 – Segmentation VLAN (site Fidjrossè) .....	21
Tableau 16 – Plan d'adressage par site .....	22
Tableau 17 – Caractéristiques backbone inter-sites.....	22
Tableau 18 – Justification architecture 2 sites .....	23
Tableau 19 – Stockage NAS par site .....	23
Tableau 20 – Comparaison Harvester vs VMware vSphere .....	24
Tableau 21 – Composants du stack Harvester.....	25
Tableau 22 – Spécifications serveurs Harvester par site .....	25
Tableau 23 – Configuration Longhorn RBER .....	26
Tableau 24 – Capacité stockage effective par site .....	26
Tableau 25 – Plan d'adressage réseau Harvester .....	26
Tableau 26 – Types de protection des données Harvester .....	27
Tableau 27 – Comparaison distributions Kubernetes.....	28
Tableau 28 – Composants du cluster RKE2.....	29
Tableau 29 – Plan d'adressage cluster Fidjrossè.....	30
Tableau 30 – Paramètres de configuration RKE2.....	30
Tableau 31 – Contrôles CIS Benchmark activés.....	31
Tableau 32 – Réseaux Kubernetes .....	31
Tableau 33 – Services exposés via HAProxy .....	33
Tableau 34 – Politique de certificats .....	33
Tableau 35 – Classes de stockage Kubernetes.....	33
Tableau 36 – Commandes de diagnostic .....	34
Tableau 37 – Stratégie de backup.....	34
Tableau 38 – URLs d'accès Harvester.....	36
Tableau 39 – Périmètre de sécurité par couche .....	37
Tableau 40 – Composants FIPS 140-2.....	37
Tableau 41 – Conformité CIS Benchmark (extrait) .....	38
Tableau 42 – Architecture IAM RBER .....	38
Tableau 43 – Rôles RBAC Kubernetes .....	39
Tableau 44 – Chiffrement en transit .....	39
Tableau 45 – Chiffrement au repos.....	40
Tableau 46 – Zones de sécurité réseau.....	40
Tableau 47 – Règles firewall (extrait) .....	41
Tableau 48 – Sources de logs et rétention.....	41
Tableau 49 – Alertes sécurité .....	41
Tableau 50 – Niveaux Pod Security Admission .....	42
Tableau 51 – Politique de rotation des credentials.....	43

# Introduction

## 1. Contexte du projet

### 1.1 Genèse

En 2020, face aux défis imposés par la pandémie de COVID-19, le gouvernement du Bénin a lancé une plateforme d'enseignement à distance. Cette solution d'urgence s'est transformée en pilier stratégique de la transformation numérique de l'éducation béninoise.

Avec le projet Africa Digital Campus (ADC) et la stratégie nationale de télé-enseignement pour la rentrée 2025-2026, l'infrastructure doit désormais supporter **200 000 utilisateurs simultanés** et garantir une disponibilité de service exemplaire.

### 1.2 Universités concernées

Code	Université	Localisation	Étudiants
UAC	Université d'Abomey-Calavi	Abomey-Calavi	~120 000
UP	Université de Parakou	Parakou	~35 000
UNA	Université Nationale d'Agriculture	Kétou	~15 000
UNSTIM	Université Nationale des Sciences, Technologies, Ingénierie et Mathématiques	Abomey	~30 000

**Tableau 1.1** – Universités publiques du Bénin

### 1.3 Limites de l'infrastructure existante

L'infrastructure précédente présentait plusieurs limitations :

- **Disponibilité insuffisante** : architecture non conçue pour la haute disponibilité
- **Scalabilité limitée** : incapacité à gérer les pics de charge (examens, rentrées)
- **Maintenance complexe** : architecture monolithique difficile à mettre à jour
- **Sécurité insuffisante** : absence de conformité aux standards (FIPS, CIS)
- **Silos techniques** : chaque université gère sa propre infrastructure

## 2. Mission et objectifs

### 2.1 Mission

Construire une infrastructure cloud-native résiliente, évolutive et hautement disponible pour héberger les plateformes éducatives (Moodle, PeerTube, BigBlueButton) au service de toutes les universités béninoises.

## 2.2 Objectifs quantitatifs

Objectif	Cible	Mesure
Disponibilité	99.9%	SLA annuel (max 8h45 d'indisponibilité)
Utilisateurs simultanés	200 000	Sessions actives concurrentes
Temps de réponse	< 2 secondes	P95 des requêtes HTTP
RTO (Recovery Time Objective)	< 4 heures	Temps de restauration après sinistre
RPO (Recovery Point Objective)	< 1 heure	Perte de données maximale
Déploiement	< 15 minutes	Temps de déploiement d'une mise à jour

Tableau 1.2 – Objectifs de niveau de service

## 3. Périmètre du projet

### 3.1 Applications hébergées

Application	Description	Utilisateurs cibles
Moodle	Plateforme e-learning (cours, devoirs, quiz)	Étudiants, enseignants
BigBlueButton	Visioconférence et classes virtuelles	Enseignants, étudiants
PeerTube	Plateforme vidéo (cours enregistrés)	Tous
Gitea	Hébergement de code source	Développeurs, étudiants IT
Harbor	Registre d'images Docker	DevOps, développeurs
ArgoCD	GitOps et déploiement continu	DevOps

Tableau 1.3 – Applications hébergées sur la plateforme

### 3.2 Sites géographiques

L'infrastructure est déployée sur deux sites principaux avec possibilité d'extension :

Site	Rôle	Localisation	Statut
Fidjrossè	Site principal	Cotonou (Sud)	Production
Parakou	Site secondaire	Parakou (Nord)	Production
IMSP	Extension future	Dangbo	Planifié
UNSTIM	Extension future	Abomey	Planifié

Tableau 1 – Sites géographiques

## 4. Principes architecturaux

### 4.1 Cloud-Native

L'architecture adopte les principes cloud-native définis par la Cloud Native Computing Foundation (CNCF) :

- **Conteneurisation** : toutes les applications sont packagées en containers Docker
- **Orchestration** : Kubernetes (RKE2) gère le cycle de vie des containers

- **Microservices** : découplage des composants pour une évolution indépendante
- **Infrastructure as Code** : toute la configuration est versionnée (GitOps)
- **Observabilité** : métriques, logs et traces pour le monitoring

## 4.2 Haute disponibilité

La haute disponibilité est assurée par :

1. **Redondance N+1** : chaque composant critique a au moins un backup
2. **Réplication des données** : stockage Longhorn avec 3 réplicas
3. **Load balancing** : HAProxy avec failover automatique
4. **Clusters PostgreSQL** : CloudNativePG avec réplication synchrone
5. **Multi-sites** : deux datacenters géographiquement séparés

## 4.3 Sécurité intégrée

La sécurité est intégrée dès la conception (Security by Design) :

- **Conformité FIPS 140-2** : chiffrement validé par le NIST
- **CIS Benchmark** : configuration Kubernetes sécurisée
- **Zero Trust** : authentification et autorisation systématiques
- **Chiffrement bout-en-bout** : TLS pour toutes les communications

## 5. Stack technologique

Catégorie	Technologie	Rôle
<b>Virtualisation</b>	Harvester 1.3+	Infrastructure hyperconvergée (compute, storage, network)
<b>Kubernetes</b>	RKE2 1.32+	Orchestration de containers (conformité FIPS/CIS)
<b>Stockage</b>	Longhorn	Stockage distribué avec réplication
<b>Base de données</b>	PostgreSQL 16 (CNPG)	Bases de données applicatives (HA)
<b>Objet storage</b>	MinIO	Stockage S3 pour backups et assets
<b>Load balancer</b>	HAProxy + Keepalived	Ingress et load balancing L7
<b>Identité</b>	Keycloak (RBER Connect)	SSO, fédération LDAP, OIDC/SAML
<b>CI/CD</b>	Gitea + Drone + ArgoCD	Pipeline de build et déploiement GitOps
<b>Monitoring</b>	Prometheus + Grafana	Métriques, alerting, dashboards
<b>Logging</b>	Loki + Promtail	Agrégation et recherche de logs
<b>Certificats</b>	cert-manager	Gestion automatique des certificats TLS

Tableau 2 – Stack technologique RBER

## 6. Organisation du document

Ce document d'architecture est organisé en chapitres thématiques :

Ch.	Titre	Contenu
1	<b>Introduction</b>	Contexte, objectifs, périmètre, principes
2	<b>Infrastructure logique</b>	Approche par plans, architecture en couches
3	<b>Infrastructure matérielle</b>	Serveurs, réseau, capacité, sites
4	<b>Infrastructure hyperconvergée</b>	Harvester, Longhorn, virtualisation
5	<b>Architecture Kubernetes</b>	RKE2, etcd, networking, ingress
6	<b>Sécurité et conformité</b>	FIPS, CIS, IAM, chiffrement, audit

Tableau 3 – Structure du document

## 7. Points clés du chapitre

- L'infrastructure RBER supporte 4 universités et 200 000 utilisateurs simultanés
- L'objectif de disponibilité est de 99.9% (SLA)
- L'architecture cloud-native repose sur Kubernetes (RKE2) et Harvester
- Deux sites géographiques assurent la haute disponibilité (Fidjrossè et Parakou)
- La sécurité est intégrée dès la conception (FIPS, CIS, Zero Trust)

# Infrastructure logique

Ce chapitre décrit l'approche architecturale adoptée pour la plateforme RBER. Plutôt qu'une vision purement technique verticale, nous utilisons une approche par plans horizontaux qui facilite la compréhension et la collaboration entre équipes.

## 1. Approche par plans

### 1.1 Pourquoi raisonner en plans

Lorsqu'on conçoit une plateforme Kubernetes, il est facile de se perdre dans la multitude de couches : infrastructure, services, sécurité. L'approche par plans offre plusieurs avantages :

- **Langage commun** : développeurs, architectes et gestionnaires parlent le même langage
- **Vision transversale** : comprendre les liens entre équipes et technologies
- **Évolutivité** : chaque plan peut être enrichi indépendamment
- **Clarté** : évite d'accumuler les couches sans structure

### 1.2 Principe directeur

« Centraliser ce qui nécessite un contrôle, décentraliser ce qui nécessite une mise à l'échelle. »

Il n'est pas nécessaire de déployer une instance Grafana par cluster, ni un service Keycloak par équipe. Les services de contrôle et d'observabilité sont centralisés, tandis que les workloads applicatifs sont distribués.

### 1.3 Vue d'ensemble des plans

Plan	Fonction	Mode
Développeur	Point d'entrée utilisateurs	Centralisé
Intégration/Livraison	Build, packaging, déploiement	Centralisé
Surveillance	Monitoring, logs, alertes	Centralisé
Sécurité	IAM, politiques, conformité	Centralisé
Ressources	Clusters, bases de données, stockage	Distribué

Tableau 4– Vue d'ensemble des plans

## 2. Plan de contrôle développeur

### 2.1 Objectif

Le plan développeur est le point d'entrée des utilisateurs. L'objectif est d'offrir aux développeurs un accès simple et sécurisé, sans les enfermer. C'est une « voie royale », pas une « prison dorée ».

### 2.2 Outils disponibles

Outil	Usage	URL
<b>Gitea</b>	Gestion du code source (Git)	git.rber.bj
<b>Harbor</b>	Registry d'images Docker	registre.rber.bj
<b>Rancher</b>	Console de gestion Kubernetes	rancher.rber.bj
<b>ArgoCD</b>	Dashboard GitOps	argocd.rber.bj
<b>Grafana</b>	Dashboards et visualisation	grafana.rber.bj
<b>Backstage (futur)</b>	Portail développeur unifié	backstage.rber.bj

Tableau 5– Outils du plan développeur

### 2.3 Flux de travail développeur

Le workflow type d'un développeur sur la plateforme RBER :

1. **Authentification** : connexion via RBER Connect (SSO)
2. **Code** : clone/push sur Gitea
3. **Build** : pipeline Drone CI déclenché automatiquement
4. **Image** : push automatique vers Harbor
5. **Déploiement** : ArgoCD synchronise avec le cluster
6. **Monitoring** : visualisation dans Grafana

## 3. Plan d'intégration et de livraison

### 3.1 Objectif

Ce plan gère tout ce qui concerne la création, le packaging et le déploiement de logiciels. L'objectif est de connecter tous les éléments de manière cohérente et automatisée pour offrir une véritable autonomie aux équipes.

### 3.2 Composants CI/CD

Composant	Technologie	Fonction
<b>Source Control</b>	Gitea	Hébergement Git, Pull Requests, Issues
<b>CI Pipeline</b>	Drone CI	Build, tests, scan sécurité, push images
<b>Registry</b>	Harbor	Stockage images Docker, scan vulnérabilités
<b>Artifacts</b>	Harbor (OCI)	Stockage Helm charts, modules Terraform
<b>GitOps</b>	ArgoCD	Déploiement déclaratif, synchronisation Git → K8s
<b>Secrets</b>	Sealed Secrets	Secrets chiffrés dans Git

Tableau 6 – Composants du plan CI/CD

### 3.3 Pipeline type

Un pipeline CI/CD typique sur la plateforme RBER :

1. **Trigger** : push sur branche main ou Pull Request
2. **Clone** : récupération du code source
3. **Build** : compilation, bundling
4. **Test** : tests unitaires et d'intégration
5. **Scan** : analyse de sécurité (SAST, dépendances)
6. **Docker build** : création de l'image container
7. **Push** : envoi vers Harbor
8. **Update manifests** : mise à jour du tag dans le repo GitOps
9. **Sync** : ArgoCD déploie automatiquement

## 4. Plan de surveillance et logging

### 4.1 Objectif

Ce plan informe sur l'état des clusters et applications, idéalement avant même que les utilisateurs ne remarquent un problème. La surveillance est **centralisée** pour éviter de jongler entre plusieurs tableaux de bord.

### 4.2 Stack d'observabilité

Composant	Technologie	Fonction
<b>Métriques</b>	Prometheus	Collecte et stockage des métriques (TSDB)
<b>Visualisation</b>	Grafana	Dashboards, exploration, requêtes ad-hoc
<b>Alerting</b>	AlertManager	Gestion des alertes, routing, silencing
<b>Logs</b>	Loki	Agrégation de logs (label-based)
<b>Collection logs</b>	Promtail	Agent de collecte sur chaque nœud
<b>Uptime</b>	Blackbox Exporter	Probes HTTP/TCP/ICMP externes

Tableau 7 – Stack d'observabilité

### 4.3 Dashboards clés

Dashboards Grafana pré-configurés :

- **Cluster Overview** : santé globale des clusters K8s
- **Node Exporter** : métriques système (CPU, RAM, disque, réseau)
- **Longhorn** : état du stockage distribué
- **PostgreSQL** : métriques des clusters CloudNativePG
- **Moodle** : performances applicatives, connexions, erreurs
- **HAProxy** : trafic, latence, codes HTTP

## 5. Plan de sécurité

### 5.1 Objectif



Le plan de sécurité englobe la gestion des identités, les politiques réseau et les règles de conformité. L'objectif est de **protéger sans étouffer**, d'intégrer la sécurité au flux de travail plutôt que de la considérer comme un obstacle.

## 5.2 Composants de sécurité

Domaine	Technologie	Fonction
<b>Identité (IAM)</b>	RBER Connect (Keycloak)	SSO, fédération LDAP, OIDC/SAML
<b>Autorisation K8s</b>	RBAC	Contrôle d'accès basé sur les rôles
<b>Politiques pods</b>	Pod Security Admission	Restriction des configurations pods
<b>Politiques réseau</b>	Calico (via Canal)	Network Policies Kubernetes
<b>Scan images</b>	Trivy (Harbor)	Détection vulnérabilités CVE
<b>Secrets</b>	Sealed Secrets / Vault (futur)	Gestion sécurisée des secrets
<b>Conformité</b>	RKE2 (FIPS/CIS)	Chiffrement validé, hardening

Tableau 8 – Composants de sécurité

## 6. Plan de ressources

### 6.1 Objectif

Le plan de ressources définit où s'exécutent les charges de travail. Il inclut les clusters de calcul, les bases de données, les systèmes de stockage et les services externes.

### 6.2 Ressources par site

Site	Ressources	Workloads
<b>Fidjrossè</b>	4× HPE DL380, 1 TB RAM, 52 TB	UAC, UNA, services centraux
<b>Parakou</b>	4× HPE DL380, 1 TB RAM, 52 TB	UP, UNSTIM, backup DR

Tableau 9 – Ressources par site

### 6.3 Services de données

Service	Technologie	Usage
<b>PostgreSQL</b>	CloudNativePG	Bases applicatives (Moodle, Gitea, etc.)
<b>Object Storage</b>	MinIO	Backups, assets vidéo, moodledata
<b>Block Storage</b>	Longhorn	Volumes persistants Kubernetes
<b>Cache</b>	Redis (futur)	Sessions, cache applicatif

Tableau 10 – Services de données

## 7. Points clés du chapitre

- L'approche par plans offre un langage commun entre équipes techniques et métier
- Les services de contrôle sont centralisés (monitoring, sécurité, CI/CD)
- Les ressources de calcul sont distribuées par site géographique
- Le plan développeur fournit un point d'entrée unique et autonome

- Le pipeline CI/CD est entièrement automatisé (GitOps)
- La sécurité est intégrée à chaque plan, pas ajoutée en couche

# Infrastructure matérielle

Ce chapitre décrit l'infrastructure physique de la plateforme RBER : serveurs, réseau, stockage et connectivité inter-sites. L'infrastructure est déployée sur deux datacenters géographiquement séparés pour assurer la haute disponibilité.

## 1. Vue d'ensemble

### 1.1 Sites de déploiement

L'infrastructure RBER est déployée sur deux sites principaux avec des clusters identiques :

Site	Localisation	Rôle	Serveurs
Fidjrossè	Datacenter SBIN, Cotonou	Principal	4× HPE
Parakou	Université de Parakou	Secondaire	4× HPE

Tableau 11 – Sites de déploiement

### 1.2 Architecture type d'un cluster

Infrastructure physique zone bj-south-n-1

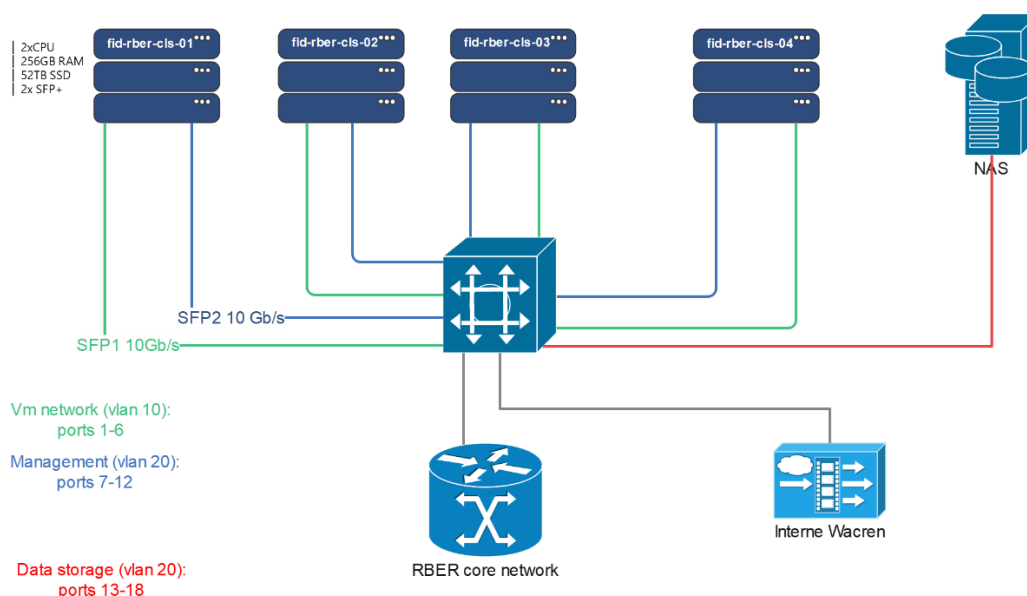


Figure 1 - Infrastructure physique

Chaque site héberge une infrastructure identique composée de :

- 4 serveurs physiques : HPE ProLiant DL380 Gen10+
- Cluster Harvester : infrastructure hyperconvergée
- Cluster RKE2 : 9 VMs (3 etcd + 3 control plane + 3 worker)

- NAS 100 TB : stockage backup local

## 2. Spécifications serveurs

### 2.1 Configuration serveur

Composant	Spécification
Modèle	HPE ProLiant DL380 Gen10+
Processeur	2× Intel Xeon Gold 6348 (28 cores @ 2.6 GHz, 56 cores total)
Mémoire	256 GB DDR4-3200 ECC RDIMM (16× 16 GB)
Stockage SSD	2× 960 GB SSD NVMe (RAID 1) - OS + cache
Stockage HDD	12× 4 TB SAS 7200 RPM (JBOD pour Longhorn) = 48 TB brut
Réseau	4× 25 GbE SFP28 (2× bond LACP)
Alimentation	2× 800W Platinum (redondant)
Management	HPE iLO 5 Advanced (IPMI)

Tableau 12 – Spécifications serveur HPE DL380

## 2.2 Capacité totale par site

Ressource	Par serveur	Total site (×4)
CPU (cores physiques)	56 cores	224 cores
CPU (vCores HT)	112 vCores	448 vCores
RAM	256 GB	1 024 GB (1 TB)
Stockage SSD	1.9 TB (RAID 1)	7.6 TB
Stockage HDD (brut)	48 TB	192 TB
Réseau	100 Gbps (2×50G)	400 Gbps agrégé

Tableau 13 – Capacité totale par site

## 2.3 Capacité totale infrastructure (2 sites)

Ressource	Total (2 sites)
Serveurs physiques	8× HPE DL380 Gen10+
CPU (cores physiques)	448 cores
CPU (vCores avec HT)	896 vCores
RAM totale	2 048 GB (2 TB)
Stockage brut HDD	384 TB
Stockage effectif (Longhorn 3×)	~128 TB
Stockage effectif (Longhorn 2×)	~192 TB
NAS backup	200 TB (100 TB × 2 sites)

Tableau 14 – Capacité totale infrastructure

# 3. Architecture réseau

## 3.1 Topologie réseau

Chaque site dispose d'une architecture réseau redondante basée sur des switchs HPE:

Core switches : 2× HPE SN2410 (25GbE, 48 ports) en stack

Bonding LACP : 2 bonds de 50 Gbps par serveur

Uplink inter-sites : backbone RBER (fibre optique)

## 3.2 Segmentation VLAN

VLAN	Nom	Usage	Réseau
1120	Management	Harvester, iLO, switches	10.29.112.0/24
1130	Production VMs	VMs RKE2, applications	10.29.113.0/24
1140	DMZ	Services exposés Internet	10.29.114.0/24
1150	Storage	Trafic Longhorn, réplication	10.29.115.0/24
1160	Backup	NAS, MinIO backup	10.29.116.0/24

Tableau 15 – Segmentation VLAN (site Fidjrossè)

### 3.3 Plan d'adressage par site

Site	Réseau	Gateway	Plage DHCP
Fidjrossè	10.29.0.0/16	10.29.112.1	.200-.254
Parakou	10.25.0.0/16	10.25.112.1	.200-.254
IMSP (futur)	10.16.0.0/16	10.16.112.1	.200-.254
UNSTIM (futur)	10.21.0.0/16	10.21.112.1	.200-.254

Tableau 16 – Plan d'adressage par site

## 4. Connectivité inter-sites

### 4.1 Backbone RBER

Paramètre	Valeur
Bande passante	300 Mbps (upgrade récent depuis 100 Mbps)
Latence	< 30 ms Fidjrossè ↔ Parakou
Technologie	Fibre optique (dark fiber RBER)
Redondance	Lien backup via Internet (VPN IPsec)
SLA opérateur	99.5% disponibilité

Tableau 17 – Caractéristiques backbone inter-sites

### 4.2 Contraintes et impact architecture

Le lien 300 Mbps entre sites impose des contraintes architecturales :

Pas de réplication synchrone : la latence (>30ms) rend la réplication PostgreSQL synchrone impraticable

Pas d'etcd stretch : un cluster etcd multi-sites nécessite <10ms de latence

Backup asynchrone : réplication MinIO et WAL shipping PostgreSQL

Clusters indépendants : chaque site opère de manière autonome

## 5. Architecture actif-actif régional

### 5.1 Mode de fonctionnement

Les deux clusters fonctionnent en mode actif-actif régional :

Fidjrossè : héberge les universités du Sud (UAC, UNA) et les services centraux

Parakou : héberge les universités du Nord (UP, UNSTIM)

Backup croisé : chaque site sauvegarde les données de l'autre

Failover manuel : en cas de sinistre, basculement vers l'autre site

## 5.2 Justification de l'architecture

Critère	Justification
Contrainte réseau	300 Mbps insuffisant pour réplication sync (>200ms latence DB)
Répartition géographique	Sud (Fidjrossè) + Nord (Parakou) couvre la majorité démographique
Complexité	Évite un mesh complet (6 liens de réplication DB)
Évolutivité	IMSP/UNSTIM peuvent passer en actif si >50k connexions
Coût	Optimise l'investissement matériel actuel

Tableau 18 – Justification architecture 2 sites

## 5.3 Avantages de l'architecture

Haute disponibilité locale : 3 nœuds minimum par cluster

Disaster Recovery : backup asynchrone entre sites

Proximité utilisateurs : latence réduite pour chaque région

Simplicité opérationnelle : pas de complexité inutile (mesh 4 sites)

Évolutivité future : IMSP et UNSTIM peuvent être ajoutés

# 6. Équipements complémentaires

## 6.1 Stockage NAS

Caractéristique	Site Fidjrossè	Site Parakou
Capacité	100 TB	100 TB
Usage	Backup local + réplication	Backup local + réplication
Protocole	NFS, S3 (MinIO)	NFS, S3 (MinIO)
RAID	RAID 6	RAID 6

Tableau 19 – Stockage NAS par site

## 6.2 Alimentation électrique

Chaque site dispose d'une alimentation redondante :

UPS : onduleur 20 kVA avec 30 minutes d'autonomie

Groupe électrogène : démarrage automatique en 30 secondes

Double alimentation : chaque serveur sur 2 circuits séparés

# 7. Points clés du chapitre

8 serveurs HPE DL380 Gen10+ répartis sur 2 sites

Capacité totale : 896 vCores, 2 TB RAM, ~128-192 TB stockage effectif

Réseau 25 GbE avec bonding LACP et segmentation VLAN

Backbone inter-sites 300 Mbps avec <30ms de latence

Architecture actif-actif régional (Sud + Nord)

200 TB de NAS backup répartis sur les 2 sites

## Infrastructure Hyperconvergée Harvester

Ce chapitre décrit l'architecture hyperconvergée déployée pour RBER, basée sur Harvester HCI. Cette solution consolide compute, stockage et réseau en une plateforme unifiée optimisée pour Kubernetes.

### 1. Vue d'ensemble de l'hyperconvergence

#### Architecture Hyperconvergée

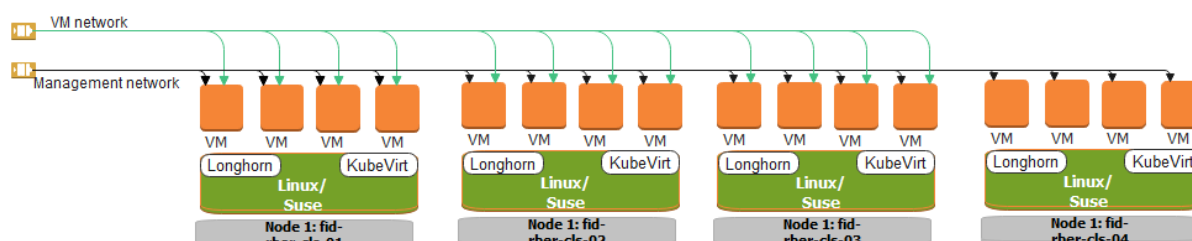


Figure 2 - architecture hyperconvergée

#### 1.1 Définition et principes

L'infrastructure hyperconvergée (HCI) est une architecture logicielle qui combine trois composants dans un cluster de nœuds identiques :

- **Compute** : virtualisation via KubeVirt (machines virtuelles sur Kubernetes)
- **Storage** : stockage distribué via Longhorn (réplication automatique)
- **Network** : réseau virtualisé via Multus et VLAN (isolation des flux)

Contrairement à une architecture traditionnelle avec SAN externe et hyperviseur séparé, l'HCI permet un scale-out horizontal : ajouter un nœud augmente simultanément CPU, RAM et stockage.

#### 1.2 Pourquoi Harvester pour RBER

Harvester est une solution HCI open-source développée par SUSE/Rancher. Elle a été retenue pour les raisons suivantes :

Critère	Harvester HCI	VMware vSphere
<b>Licence</b>	Open-source (Apache 2.0)	Commerciale (~\$6000/CPU)
<b>Stockage</b>	Longhorn intégré	vSAN ou SAN externe
<b>Intégration K8s</b>	Native (Rancher)	Tanzu requis
<b>Migration live</b>	Oui	Oui (vMotion)
<b>Snapshots VM</b>	Oui (Longhorn)	Oui
<b>GPU Passthrough</b>	Oui (PCI)	Oui
<b>Support RBER</b>	Cohérent avec RKE2	Stack séparé

Tableau 20 – Comparaison Harvester vs VMware vSphere



## 2. Architecture technique Harvester

### 2.1 Composants du stack Harvester

Harvester repose sur plusieurs composants Kubernetes natifs :

Composant	Fonction	Description
<b>KubeVirt</b>	Virtualisation	Exécute les VMs comme pods Kubernetes via QEMU/KVM
<b>Longhorn</b>	Stockage distribué	Stockage bloc répliqué avec snapshots et backups
<b>Multus CNI</b>	Réseau multi-homed	Attache plusieurs interfaces réseau aux VMs
<b>Elemental</b>	OS immutable	Système d'exploitation basé sur SLE Micro, mis à jour atomiquement
<b>Rancher</b>	Gestion clusters	Interface de gestion des clusters RKE2 downstream

Tableau 21 – Composants du stack Harvester

### 2.2 Architecture physique par site

Chaque site RBER (Fidjrossè et Parakou) dispose d'un cluster Harvester identique :

Serveur	CPU	RAM	Stockage
HPE DL380 Gen10+ #1	2× Xeon Gold 6348 (56c)	256 GB DDR4	13 TB NVMe/SSD
HPE DL380 Gen10+ #2	2× Xeon Gold 6348 (56c)	256 GB DDR4	13 TB NVMe/SSD
HPE DL380 Gen10+ #3	2× Xeon Gold 6348 (56c)	256 GB DDR4	13 TB NVMe/SSD
HPE DL380 Gen10+ #4	2× Xeon Gold 6348 (56c)	256 GB DDR4	13 TB NVMe/SSD

Tableau 22 – Spécifications serveurs Harvester par site

**Capacité totale par site :** 224 vCores physiques (448 avec HyperThreading), 1 TB RAM, 52 TB stockage brut.

## 3. Stockage distribué Longhorn

### 3.1 Fonctionnement de Longhorn

Longhorn est le moteur de stockage intégré à Harvester. Il fournit un stockage bloc distribué et répliqué sans dépendance à un SAN externe.

**Caractéristiques principales :**

- **Réplication synchrone** : chaque volume est répliqué sur 2 ou 3 nœuds
- **Snapshots instantanés** : création de points de restauration sans interruption
- **Backup S3** : sauvegarde vers MinIO ou stockage compatible S3
- **Rebuild automatique** : reconstruction des réplicas en cas de panne nœud
- **Thin provisioning** : allocation dynamique de l'espace disque

### 3.2 Configuration stockage RBER

La configuration Longhorn a été optimisée pour les workloads éducatifs (Moodle, BigBlueButton) :

Paramètre	Valeur	Justification
Replica count (défaut)	3	Tolérance 1 panne nœud
Replica count (workloads non critiques)	2	Économie stockage
Data locality	best-effort	Performance I/O locale
Backup target	s3://rber-backups	MinIO local
Recurring snapshot	Quotidien 02:00	RPO 24h
Snapshot retention	7 jours	Rétention courte

Tableau 23 – Configuration Longhorn RBER

### 3.3 Capacité effective

Le stockage disponible dépend du facteur de réplication choisi :

Configuration	Stockage brut (par site)	Stockage effectif
Réplication 3×	52 TB	~17 TB
Réplication 2×	52 TB	~26 TB
Mixte (recommandé)	52 TB	~20 TB

Tableau 24 – Capacité stockage effective par site

## 4. Réseau virtualisé

### 4.1 Architecture réseau Harvester

Harvester utilise une architecture réseau à deux niveaux :

1. **Réseau de management (VLAN 1120)** : communication inter-nœuds  
Harvester, API, interface web
2. **Réseau VM (VLAN 1130)** : trafic des machines virtuelles, accès services

### 4.2 Plan d'adressage Harvester

Site	Réseau MGMT	Réseau VM	VLAN
Fidjrossè (SBIN)	10.29.112.128/25	10.29.113.0/24	1120 / 1130
Parakou (UP)	10.25.112.128/25	10.25.113.0/24	1120 / 1130
IMSP (futur)	10.16.112.128/25	10.16.113.0/24	1120 / 1130
UNSTIM (futur)	10.21.112.128/25	10.21.113.0/24	1120 / 1130

Tableau 25 – Plan d'adressage réseau Harvester

### 4.3 Configuration réseau des nœuds

Chaque serveur Harvester dispose de 4 interfaces réseau 25 GbE configurées en bonding :

- **bond0** : interfaces eno1 + eno2 (LACP 802.3ad) – réseau management
- **bond1** : interfaces eno3 + eno4 (LACP 802.3ad) – réseau VM et stockage

Le switch HPE SN2410 gère le trunk VLAN vers les serveurs avec QoS prioritaire pour le trafic stockage.

## 5. Gestion des machines virtuelles

### 5.1 Création de VM

Harvester permet de créer des VMs de trois façons :

1. **Interface web Harvester** : création manuelle via formulaire
2. **Templates VM** : modèles préconfigurés (Ubuntu, Rocky Linux)
3. **Terraform provider** : Infrastructure as Code pour automatisation

## 5.2 Migration live

La migration live permet de déplacer une VM d'un nœud à un autre sans interruption de service. Cette fonctionnalité est utilisée pour :

- Maintenance planifiée des nœuds (mise à jour firmware, BIOS)
- Rééquilibrage de charge entre nœuds
- Évacuation avant panne prévisible (disque SMART warning)

**Prérequis** : stockage partagé Longhorn, CPU compatible entre nœuds, réseau VM accessible sur les deux nœuds.

## 5.3 Snapshots et backups

Harvester offre deux niveaux de protection des données :

Type	Caractéristiques	Usage RBER
<b>Snapshot VM</b>	Instantané local, restauration rapide	Avant mise à jour applicative
<b>Backup VM</b>	Export vers S3 (MinIO), hors-site	DR et archivage long terme
<b>Snapshot volume</b>	Snapshot Longhorn du disque	Backup bases de données

Tableau 26 – Types de protection des données Harvester

# Architecture Kubernetes RKE2

Ce chapitre décrit l'architecture Kubernetes déployée pour RBER, basée sur RKE2 (Rancher Kubernetes Engine 2). Cette distribution est conçue pour les environnements de production avec des exigences de sécurité élevées.

## 1. Présentation de RKE2

### 1.1 Qu'est-ce que RKE2

RKE2 est une distribution Kubernetes développée par SUSE/Rancher. Elle combine la simplicité de K3s avec la conformité aux standards de sécurité gouvernementaux américains (FIPS 140-2, CIS Benchmark).

#### Caractéristiques principales :

- **Conformité FIPS 140-2** : chiffrement validé pour les environnements gouvernementaux
- **CIS Hardening** : configuration sécurisée par défaut selon le CIS Kubernetes Benchmark
- **Embedded etcd** : etcd intégré ou externe selon les besoins
- **Containerd** : runtime container sans Docker (plus léger et sécurisé)
- **Canal CNI** : plugin réseau par défaut (Flannel + Calico network policies)

### 1.2 Pourquoi RKE2 pour RBER

Critère	RKE2	K3s	Vanilla K8s
<b>FIPS 140-2</b>	Oui (natif)	Non	Manuel
<b>CIS Benchmark</b>	Oui (profile)	Partiel	Manuel
<b>etcd externe</b>	Oui	SQLite/etcd	Oui
<b>Production ready</b>	Oui	Edge/IoT	Oui
<b>Installation</b>	Simple (binaire)	Très simple	Complexe
<b>Support Rancher</b>	Natif	Natif	Import

Tableau 27 – Comparaison distributions Kubernetes

## 2. Architecture du cluster

### 2.1 Topologie haute disponibilité

L'architecture RBER sépare physiquement etcd du control plane pour optimiser les performances et la résilience. Cette topologie est recommandée pour les clusters de production supportant plus de 100 nœuds ou 5000 pods.

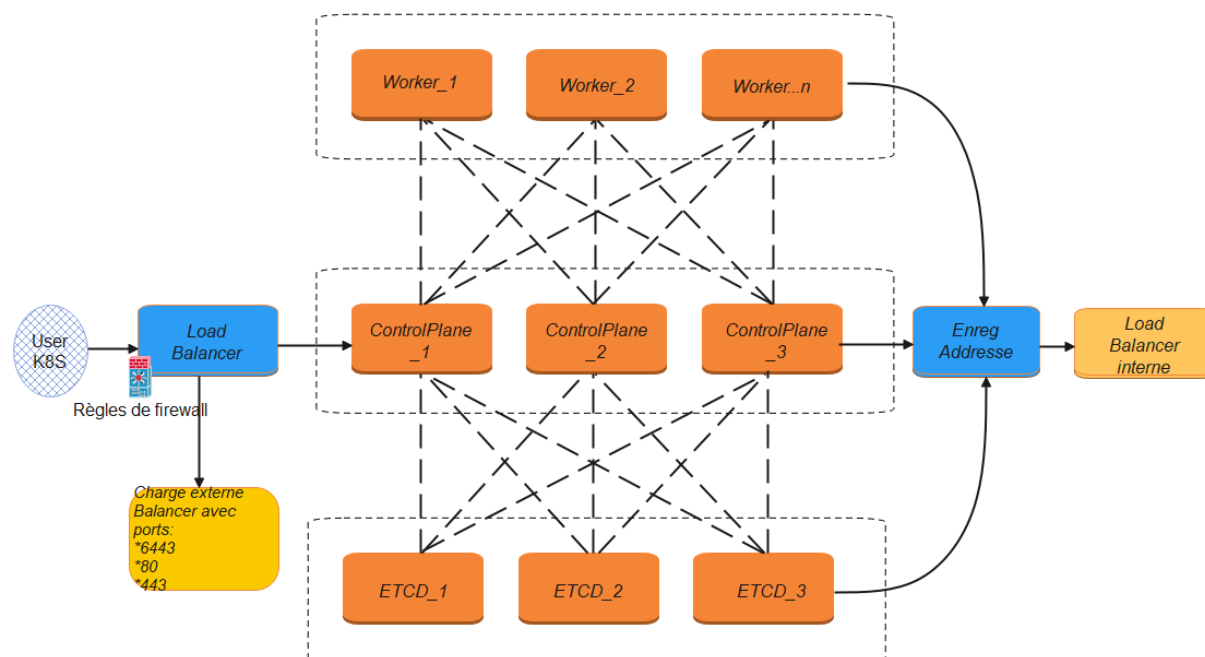


Figure 3 - Architecture cluster haute disponibilité

#### Composants du cluster :

Rôle	Nombre	Ressources	Fonction
<b>etcd</b>	3	8 vCPU, 16 GB	Base de données distribuée du cluster
<b>Control Plane</b>	3	16 vCPU, 32 GB	API Server, Scheduler, Controller Manager
<b>Worker</b>	3-5	32 vCPU, 128 GB	Exécution des workloads applicatifs
<b>Load Balancer</b>	2	2 vCPU, 4 GB	HAProxy + Keepalived (VIP)

Tableau 28 – Composants du cluster RKE2

### 2.2 Justification de la séparation etcd

La séparation d'etcd du control plane apporte plusieurs avantages :

1. **Performance** : etcd dispose de ressources dédiées pour les I/O disque (latence <10ms)
2. **Isolation** : une charge CPU sur l'API Server n'impacte pas les écritures etcd
3. **Sécurité** : surface d'attaque réduite (etcd non exposé sur les control planes)
4. **Maintenance** : mise à jour etcd indépendante du control plane

## 2.3 Plan d'adressage cluster Fidjrossè

Hostname	Rôle	IP	Ressources
fid-rke2-etcd-01	Etcd	10.29.113.130	8 vCPU, 16 GB, 240 GB
fid-rke2-etcd-02	Etcd	10.29.113.131	8 vCPU, 16 GB, 240 GB
fid-rke2-etcd-03	Etcd	10.29.113.132	8 vCPU, 16 GB, 240 GB
fid-rke2-srv-01	Control Plane	10.29.113.134	16 vCPU, 32 GB, 240 GB
fid-rke2-srv-02	Control Plane	10.29.113.135	16 vCPU, 32 GB, 240 GB
fid-rke2-srv-03	Control Plane	10.29.113.136	16 vCPU, 32 GB, 240 GB
fid-rke2-wrk-01	Worker	10.29.113.160	32 vCPU, 128 GB, 500 GB
fid-rke2-wrk-02	Worker	10.29.113.161	32 vCPU, 128 GB, 500 GB
fid-rke2-wrk-03	Worker	10.29.113.162	32 vCPU, 128 GB, 500 GB

Tableau 29 – Plan d'adressage cluster Fidjrossè

## 3. Configuration RKE2

### 3.1 Fichier de configuration serveur

La configuration RKE2 est définie dans le fichier **/etc/rancher/rke2/config.yaml**. Voici les paramètres clés :

Paramètre	Valeur	Description
<b>profile</b>	cis-1.23	Active le hardening CIS Benchmark
<b>selinux</b>	true	Active SELinux pour les conteneurs
<b>secrets-encryption</b>	true	Chiffre les secrets dans etcd
<b>write-kubeconfig-mode</b>	0600	Permissions restrictives kubeconfig
<b>tls-san</b>	[VIP, DNS]	SANs pour le certificat API Server
<b>disable</b>	rke2-ingress-nginx	Désactive l'ingress par défaut (HAProxy)
<b>cni</b>	canal	Plugin réseau (Flannel + Calico policies)
<b>cluster-cidr</b>	10.42.0.0/16	Réseau des pods
<b>service-cidr</b>	10.43.0.0/16	Réseau des services

Tableau 30 – Paramètres de configuration RKE2

### 3.2 Configuration etcd externe

Les nœuds etcd sont configurés en cluster séparé avec les paramètres suivants :

- **Endpoints** :  
https://10.29.113.130:2379,https://10.29.113.131:2379,https://10.29.113.132:2379
- **Authentification** : certificats TLS mutuels (mTLS)
- **Snapshot automatique** : toutes les 12 heures, rétention 5 snapshots
- **Quota DB** : 8 GB (défragmentation automatique à 80%)

### 3.3 Conformité CIS Benchmark

Le profil CIS active automatiquement les contrôles de sécurité suivants :

ID CIS	Contrôle	Statut RKE2
1.2.1	Anonymous auth disabled on API server	Activé par défaut
1.2.6	Kubelet certificate authority set	Activé par défaut
1.2.16	Admission controller PodSecurity enabled	Activé par défaut
1.2.22	Audit logging enabled	Activé par défaut
4.2.1	Kubelet authentication/authorization	Activé par défaut
5.1.1	RBAC enabled	Activé par défaut

Tableau 31 – Contrôles CIS Benchmark activés

## 4. Networking Kubernetes

### 4.1 Plugin CNI Canal

RKE2 utilise Canal comme plugin CNI par défaut. Canal combine Flannel (overlay network) et Calico (network policies) :

- **Flannel VXLAN** : encapsulation des paquets pods dans des tunnels UDP
- **Calico Felix** : enforcement des NetworkPolicies Kubernetes
- **MTU** : 1450 (VXLAN overhead de 50 bytes)

### 4.2 Architecture réseau du cluster

Type	CIDR	Usage
<b>Pod Network</b>	10.42.0.0/16	IP des pods (65k adresses)
<b>Service Network</b>	10.43.0.0/16	ClusterIP des services
<b>Node Network</b>	10.29.113.0/24	IP des nœuds (Fidjrossè)

Tableau 32 – Réseaux Kubernetes

### 4.3 Network Policies par défaut

Les NetworkPolicies suivantes sont appliquées par défaut pour isoler les namespaces:

1. **Deny all ingress** : bloque tout trafic entrant par défaut
2. **Allow DNS** : autorise la résolution DNS (kube-system)
3. **Allow monitoring** : autorise le scraping Prometheus
4. **Allow ingress** : autorise le trafic depuis l'ingress controller

## 5. Ingress et Load Balancing

### 5.1 Architecture HAProxy

RBER utilise HAProxy comme ingress controller et load balancer externe. Cette architecture offre :

- **VIP (Virtual IP)** : 10.29.112.100 (Keepalived VRRP)
- **Load balancing L7** : routage HTTP/HTTPS basé sur le hostname
- **TLS termination** : déchiffrement SSL sur HAProxy
- **Health checks** : vérification état des backends

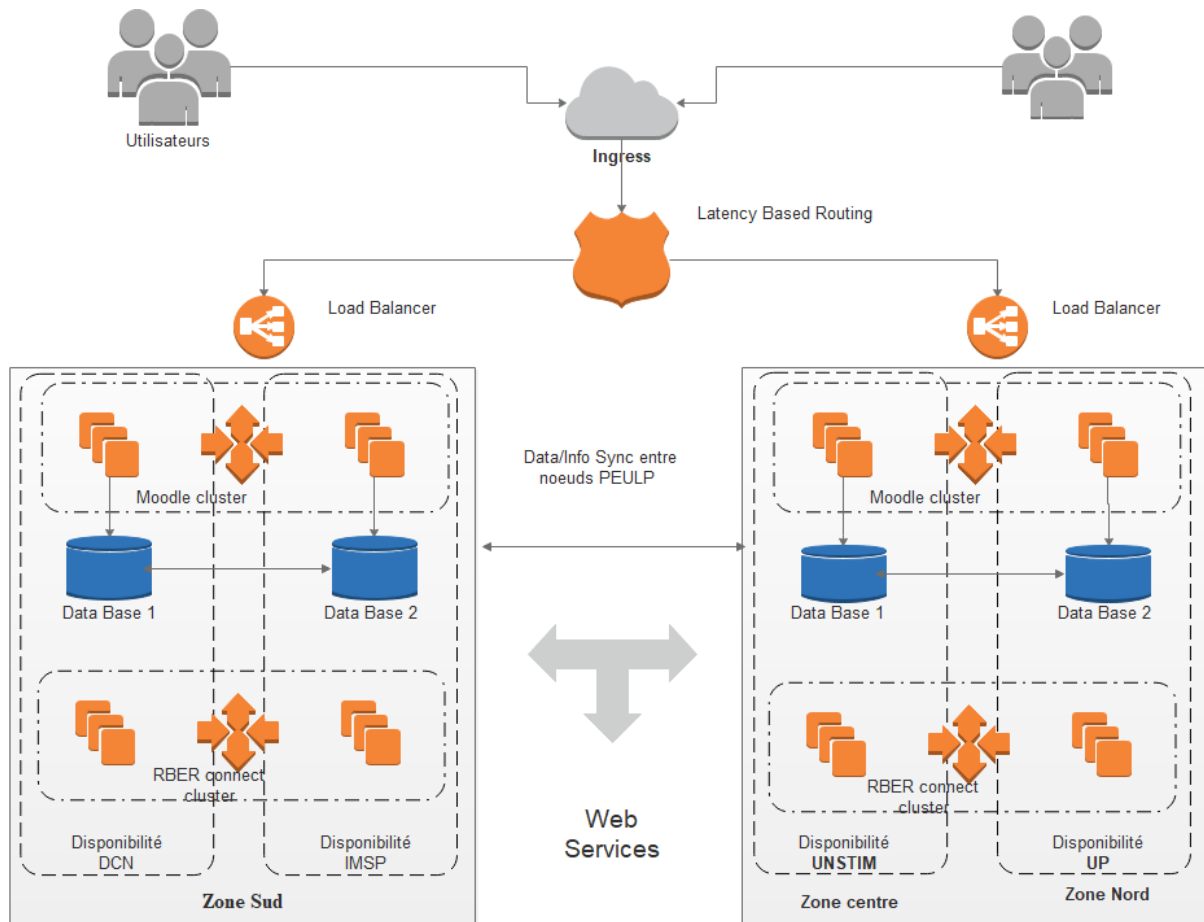


Figure 4 - Architecture HAPROXY



## 5.2 Services exposés

Service	URL	Backend	Port
Kubernetes API	api.rber.bj	Control Planes	6443
Rancher	cloud.rber.bj	Workers	443
Gitea	git.rber.bj	Workers	80/443
Harbor	registre.rber.bj	Workers	80/443
Drone CI	registre.rber.bj	Workers	80/443
ArgoCD	argocd.rber.bj	Workers	443
Grafana	grafana.rber.bj	Workers	80/443

Tableau 33 – Services exposés via HAProxy

## 6. Gestion des certificats

### 6.1 cert-manager

cert-manager automatise la gestion des certificats TLS dans le cluster. Il gère le renouvellement automatique et l'intégration avec Let's Encrypt ou une CA interne.

Type	Émetteur	Durée	Renouvellement
<b>Services publics</b>	Let's Encrypt	90 jours	Auto (30j avant)
<b>Services internes</b>	CA RBER	1 an	Auto (60j avant)
<b>Kubernetes API</b>	RKE2 CA	1 an	Rotation manuelle
<b>etcd mTLS</b>	RKE2 CA	1 an	Rotation manuelle

Tableau 34 – Politique de certificats

### 6.2 Rotation des certificats Kubernetes

RKE2 génère automatiquement les certificats du cluster lors de l'installation. La rotation doit être planifiée annuellement :

1. Sauvegarder etcd avant la rotation
2. Exécuter rke2 certificate rotate sur chaque control plane
3. Redémarrer les services rke2-server
4. Vérifier la connectivité API et les workloads

## 7. Stockage Kubernetes

### 7.1 StorageClasses

Le cluster expose plusieurs classes de stockage via Longhorn :

StorageClass	Réplication	Reclaim	Usage
<b>longhorn (default)</b>	3	Delete	Workloads standards
<b>longhorn-ha</b>	3	Retain	Bases de données critiques
<b>longhorn-fast</b>	2	Delete	Workloads non critiques

Tableau 35 – Classes de stockage Kubernetes

### 7.2 CloudNativePG pour PostgreSQL

CloudNativePG est l'opérateur PostgreSQL déployé pour gérer les bases de données des applications (Moodle, Gitea, Harbor). Il offre :

- **Haute disponibilité** : clusters PostgreSQL avec réplication synchrone
- **Failover automatique** : promotion du standby en cas de panne du primary
- **Backup continu** : WAL archiving vers MinIO (S3)
- **Point-in-time recovery** : restauration à un instant précis

## 8. Exploitation et maintenance

### 8.1 Commandes de diagnostic

Principales commandes pour le diagnostic du cluster :

Commande	Description
kubectl get nodes -o wide	État des nœuds avec IP et version
kubectl get pods -A   grep -v Running	Pods en erreur
kubectl top nodes	Utilisation CPU/RAM des nœuds
etcdctl endpoint status --cluster	État du cluster etcd
kubectl get events --sort-by=.lastTimestamp	Événements récents du cluster

Tableau 36 – Commandes de diagnostic

### 8.2 Mise à jour du cluster

La mise à jour RKE2 suit une procédure rolling upgrade :

1. Sauvegarder etcd (snapshot)
2. Mettre à jour les nœuds etcd un par un
3. Mettre à jour les control planes un par un
4. Cordon, drain et mise à jour des workers
5. Valider les workloads et uncordon

**Temps estimé** : 2-3 heures pour un cluster complet (9 nœuds).

### 8.3 Backup et restauration

Stratégie de backup pour le cluster :

Composant	Méthode	Fréquence	Rétention
Etcd	etcdctl snapshot	Toutes les 12h	5 snapshots
PVCs critiques	Longhorn backup	Quotidien	30 jours
PostgreSQL	CloudNativePG WAL	Continu	7 jours
Manifests K8s	GitOps (ArgoCD)	À chaque commit	Historique Git

Tableau 37 – Stratégie de backup

## 9. Points clés du chapitre

- RKE2 offre une conformité FIPS et CIS Benchmark native
- La séparation etcd/control plane améliore performances et résilience
- Canal CNI assure le réseau pods et les network policies
- HAProxy fournit l'ingress et le load balancing L7

- cert-manager automatise la gestion des certificats TLS
- CloudNativePG gère les clusters PostgreSQL haute disponibilité

## 6. Exploitation et maintenance

### 6.1 Interface de gestion

Harvester expose une interface web accessible via :

Site	URL Harvester	Accès
Fidjrossè	<a href="https://harvester.fid.rber.bj">https://harvester.fid.rber.bj</a>	VPN ou réseau interne
Parakou	<a href="https://harvester.up.rber.bj">https://harvester.up.rber.bj</a>	VPN ou réseau interne

Tableau 38 – URLs d'accès Harvester

### 6.2 Procédures de maintenance

#### Maintenance préventive mensuelle :

1. Vérification état de santé des volumes Longhorn
2. Contrôle répartition des réplicas sur les nœuds
3. Vérification des alertes SMART disques
4. Test de restauration snapshot (environnement de test)
5. Mise à jour firmware si nécessaire (fenêtre planifiée)

### 6.3 Mise à jour Harvester

Les mises à jour Harvester suivent un processus rolling upgrade :

- Migration des VMs du nœud à mettre à jour
- Mise en maintenance du nœud (cordon + drain)
- Application de la mise à jour (reboot automatique)
- Remise en service et passage au nœud suivant

**Temps estimé :** 15-20 minutes par nœud, soit environ 1h pour un cluster de 4 nœuds.

## 7. Points clés du chapitre

- Harvester consolide compute, stockage et réseau dans une architecture unifiée
- Longhorn assure la réplication des données avec rebuild automatique
- La migration live permet une maintenance sans interruption
- L'architecture 4 nœuds par site garantit la tolérance aux pannes
- Le coût total de possession est réduit grâce à l'open-source

# Sécurité et Conformité

Ce chapitre décrit la stratégie de sécurité de l'infrastructure RBER. Il couvre la conformité aux standards, la gestion des identités, le chiffrement, l'audit et les bonnes pratiques de sécurité pour un environnement Kubernetes de production.

## 1. Stratégie de sécurité

### 1.1 Principes directeurs

La sécurité de l'infrastructure RBER repose sur le modèle Defense in Depth (défense en profondeur) avec plusieurs couches de protection :

- **Zero Trust** : ne jamais faire confiance, toujours vérifier
- **Least Privilege** : accorder uniquement les permissions nécessaires
- **Separation of Duties** : séparer les responsabilités
- **Defense in Depth** : multiplier les couches de sécurité
- **Secure by Default** : configuration sécurisée dès l'installation

### 1.2 Périmètre de sécurité

Couche	Composants	Mesures de sécurité
Réseau	Firewall, VLAN, VPN	Segmentation, filtrage, chiffrement
Infrastructure	Harvester, serveurs, stockage	Hardening OS, accès SSH restreint
Kubernetes	RKE2, etcd, API Server	RBAC, PSA, Network Policies
Applications	Moodle, Gitea, Harbor	Auth centralisée, WAF, scans
Données	PostgreSQL, MinIO, volumes	Chiffrement at-rest, backups

Tableau 39 – Périmètre de sécurité par couche

## 2. Conformité aux standards

### 2.1 FIPS 140-2

RKE2 est certifié FIPS 140-2 (Federal Information Processing Standard). Cette certification garantit que les modules cryptographiques utilisés sont validés par le NIST.

**Composants FIPS activés :**

Composant	Module cryptographique	Usage
RKE2 Server	BoringCrypto	TLS API Server, etcd
containerd	BoringCrypto	Registry pulls (HTTPS)
etcd	BoringCrypto	Communication inter-nœuds
kubectl	BoringCrypto	Connexion API Server

Tableau 40 – Composants FIPS 140-2

**Activation FIPS** : Le mode FIPS est activé via l'option **INSTALL\_RKE2\_METHOD=fips** lors de l'installation.

## 2.2 CIS Kubernetes Benchmark

Le CIS Benchmark définit les bonnes pratiques de sécurité pour Kubernetes. RKE2 applique automatiquement le profil CIS 1.23 (ou supérieur).

ID	Contrôle	Niveau	Statut RBER
1.1.1	API Server pod spec file permissions	Level 1	Conforme
1.2.1	Anonymous authentication disabled	Level 1	Conforme
1.2.6	Kubelet certificate authority set	Level 1	Conforme
1.2.16	PodSecurity admission enabled	Level 1	Conforme
1.2.22	Audit logging enabled	Level 1	Conforme
1.3.2	Controller Manager profiling disabled	Level 1	Conforme
2.1	etcd TLS encryption enabled	Level 1	Conforme
4.2.1	Kubelet authentication enabled	Level 1	Conforme
5.1.1	RBAC enabled	Level 1	Conforme
5.2.1	Privileged containers restricted	Level 1	Conforme

Tableau 41 – Conformité CIS Benchmark (extrait)

## 3. Gestion des identités et accès (IAM)

### 3.1 Architecture IAM RBER

RBER utilise une architecture IAM centralisée basée sur Keycloak (RBER Connect) avec intégration LDAP des universités.

**Composants IAM :**

Composant	Fonction	Intégration
<b>Keycloak</b>	Identity Provider (IdP)	OIDC, SAML 2.0
<b>LDAP universités</b>	Annuaire utilisateurs	Federation read-only
<b>Rancher</b>	Gestion clusters K8s	OIDC via Keycloak
<b>Gitea</b>	Gestion code source	OAuth2 via Keycloak
<b>Harbor</b>	Registry Docker	OIDC via Keycloak
<b>Moodle</b>	LMS	OAuth2/LDAP direct

Tableau 42 – Architecture IAM RBER

## 3.2 RBAC Kubernetes

Le contrôle d'accès basé sur les rôles (RBAC) définit qui peut faire quoi dans le cluster Kubernetes.

Rôle	Scope	Permissions
<b>cluster-admin</b>	Cluster	Toutes les ressources, tous les verbes
<b>admin</b>	Namespace	CRUD sur toutes les ressources du namespace
<b>Edit</b>	Namespace	CRUD sauf RBAC et quotas
<b>view</b>	Namespace	Lecture seule sur les ressources
<b>dev-team-uac</b>	moodle-uac	CRUD pods, deployments, services, configmaps

Tableau 43 – Rôles RBAC Kubernetes

## 3.3 Authentification multi-facteur (MFA)

Le MFA est requis pour tous les accès administrateurs :

- **Rancher UI** : TOTP (Google Authenticator, Authy)
- **SSH serveurs** : clé SSH + passphrase (ou certificat)
- **Harvester UI** : TOTP via Keycloak
- **Harbor/Gitea** : TOTP optionnel, recommandé pour maintenir

# 4. Chiffrement

## 4.1 Chiffrement en transit (TLS)

Toutes les communications sont chiffrées avec TLS 1.2 minimum (TLS 1.3 recommandé) :

Communication	Protocole	Cipher	Certificat
Client → API Server	TLS 1.3	AES-256-GCM	RKE2 CA
API Server → etcd	mTLS 1.3	AES-256-GCM	RKE2 CA
etcd inter-nœuds	mTLS 1.3	AES-256-GCM	etcd CA
Kubelet → API Server	mTLS 1.2+	AES-256-GCM	RKE2 CA
Ingress → Backend	TLS 1.2+	AES-256-GCM	Cluster CA
User → Ingress	TLS 1.2+	AES-256-GCM	Let's Encrypt

Tableau 44 – Chiffrement en transit

## 4.2 Chiffrement au repos

Les données sensibles sont chiffrées au repos :

Données	Méthode	Clé
<b>Secrets Kubernetes</b>	AES-CBC ou AES-GCM	EncryptionConfig RKE2
<b>etcd</b>	AES-256	Secrets encryption key
<b>Volumes Longhorn</b>	LUKS2 (dm-crypt)	Per-volume key (optionnel)
<b>Backups MinIO</b>	SSE-S3	MinIO KMS
<b>PostgreSQL</b>	pgcrypto (colonnes)	Application-level

Tableau 45 – Chiffrement au repos

## 4.3 Gestion des secrets

Les secrets sont gérés via plusieurs mécanismes :

1. **Kubernetes Secrets** : stockage natif (chiffré dans etcd)
2. **External Secrets Operator** : synchronisation depuis sources externes
3. **Sealed Secrets** : secrets chiffrés dans Git (GitOps)
4. **HashiCorp Vault** : gestion centralisée (futur)

## 5. Sécurité réseau

### 5.1 Segmentation réseau

L'infrastructure est segmentée en plusieurs zones de sécurité :

Zone	VLAN	Accès autorisé	Niveau
<b>Management</b>	1120	Admins RBER (VPN)	Critique
<b>Production VMs</b>	1130	Services internes	Élevé
<b>DMZ</b>	1140	Internet (filtré)	Moyen
<b>Backup</b>	1150	Serveurs backup	Critique

Tableau 46 – Zones de sécurité réseau

### 5.2 Network Policies Kubernetes

Les Network Policies isolent les workloads au niveau pod. Politique par défaut : deny all ingress.

**Policies appliquées par namespace :**

- **default-deny-ingress** : bloque tout trafic entrant non explicitement autorisé
- **allow-dns** : autorise la résolution DNS vers kube-system
- **allow-ingress-controller** : autorise le trafic depuis HAProxy
- **allow-monitoring** : autorise le scraping Prometheus
- **allow-database** : autorise les connexions vers PostgreSQL

### 5.3 Firewall et règles

Les règles firewall sont appliquées au niveau du switch et des serveurs :



Source	Destination	Port	Protocole	Action
Internet	HAProxy VIP	80, 443	TCP	Allow
VPN Admins	Management	22, 443	TCP	Allow
Nodes	etcd	2379-2380	TCP	Allow
Any	Any	*	*	Deny

Tableau 47 – Règles firewall (extrait)

## 6. Audit et logging

### 6.1 Architecture de logging

Les logs sont collectés et centralisés via la stack Loki/Promtail :

Source	Type de logs	Rétention
<b>API Server</b>	Audit logs (who did what)	90 jours
<b>Pods applicatifs</b>	stdout/stderr	30 jours
<b>Nodes (syslog)</b>	System events	30 jours
<b>HAProxy</b>	Access logs	90 jours
<b>Authentification</b>	Login/logout, échecs	1 an

Tableau 48 – Sources de logs et rétention

### 6.2 Audit Kubernetes

L'audit Kubernetes enregistre toutes les actions sur l'API Server. Le niveau d'audit est configuré par ressource :

- **Secrets** : niveau Metadata (pas le contenu)
- **ConfigMaps** : niveau Metadata
- **Pods, Deployments** : niveau Request (inclut le body)
- **RBAC** : niveau RequestResponse (requête + réponse)

### 6.3 Alerting sécurité

Des alertes sont configurées pour les événements de sécurité critiques :

Événement	Sévérité	Notification
Échec auth API Server (>5/min)	Critical	Email + SMS
Pod privileged créé	Warning	Email
Secret accédé (namespace sensible)	Info	Log only
Modification RBAC	Warning	Email
Node NotReady	Critical	Email + SMS

Tableau 49 – Alertes sécurité

## 7. Sécurité des conteneurs

### 7.1 Pod Security Admission (PSA)

Kubernetes 1.25+ utilise Pod Security Admission pour contrôler les configurations de pods. RBER applique le niveau **restricted** par défaut.

Niveau PSA	Restrictions	Namespaces RBER
<b>privileged</b>	Aucune restriction	kube-system, longhorn-system
<b>baseline</b>	Bloque hostNetwork, privileged	monitoring, logging
<b>restricted</b>	+ runAsNonRoot, drop ALL caps	Tous les autres (défaut)

Tableau 50 – Niveaux Pod Security Admission

### 7.2 Scan d'images

Harbor effectue un scan automatique des images avec Trivy :

1. **Scan au push** : chaque image poussée est scannée
2. **Scan périodique** : re-scan quotidien pour nouvelles CVE
3. **Politique de blocage** : images avec CVE Critical bloquées
4. **Rapport** : dashboard vulnérabilités dans Harbor UI

### 7.3 Bonnes pratiques images

Standards pour les images Docker déployées sur RBER :

- **Images de base** : utiliser des images minimales (Alpine, Distroless)
- **User non-root** : exécuter en tant qu'utilisateur non-root
- **Read-only FS** : filesystem racine en lecture seule si possible
- **Pas de secrets** : aucun secret dans les layers de l'image
- **Versions fixées** : utiliser des tags spécifiques, pas :latest

## 8. Procédures de sécurité

### 8.1 Réponse aux incidents

Procédure en cas d'incident de sécurité :

1. **Détection** : alerte reçue ou comportement anormal observé
2. **Qualification** : évaluer la sévérité et l'impact
3. **Containment** : isoler le composant compromis
4. **Investigation** : analyser les logs et traces
5. **Remediation** : corriger la vulnérabilité
6. **Recovery** : restaurer le service
7. **Post-mortem** : documenter et améliorer

## 8.2 Rotation des credentials

Fréquence de rotation des différents credentials :

Credential	Fréquence	Méthode
Certificats Kubernetes	Annuel	rke2 certificate rotate
Tokens service accounts	90 jours	Bound SA tokens (auto)
Mots de passe DB	Trimestriel	Script rotation
Clés SSH	Annuel	Régénération manuelle
Secrets encryption key	Annuel	Procédure documentée

Tableau 51 – Politique de rotation des credentials

## 9. Points clés du chapitre

- RKE2 assure la conformité FIPS 140-2 et CIS Benchmark nativement
- Keycloak centralise l'authentification avec fédération LDAP des universités
- Toutes les communications sont chiffrées TLS 1.2+ (mTLS pour etcd)
- Les secrets Kubernetes sont chiffrés au repos dans etcd
- Network Policies isolent les workloads avec deny-all par défaut
- Pod Security Admission enforce le niveau restricted sur les namespaces applicatifs
- Harbor scanne automatiquement les images pour les vulnérabilités
- Les logs d'audit sont conservés 90 jours minimum