LifeKeeper for Linux Quorum/Witness Kit Storage mode on Azure Step by step Guide

第1版



サイオステクノロジー株式会社

目次

1.	始めに
2.	本ドキュメント6
2.1	対象読者について6
2.2	必要なソフトウェアイメージ6
2.3	製品に関する情報6
3.	QWK によるスプリットブレイン対策7
3.1	スプリットブレインとは7
3.2	QWK の概要7
3.3	QWK の storage モードの概要8
4.	Azure の環境構築9
4.1	ネットワーク構成図9
4.2	リソースグループの作成9
4.3	仮想ネットワークの作成10
4.4	可用性セットの作成12
4.5	仮想マシンの作成13
4.6	NICの設定21
4.7	インターナルロードバランサー(ILB)の作成25
5.	OS の設定
5.1	仮想マシンのログイン30
5.2	ssh の設定31
5.3	NIC アドレスの固定化31
5.4	SELinux の無効化33
5.5	ファイアウォールの無効化33
5.6	ホスト名の名前解決34
6.	ローカルリポジトリの設定35
6.1	/home の拡張35
6.2	OS イメージのマウント35
6.3	ローカルリポジトリの作成36
6.4	rh-cloud-base.repo の無効化36
6.5	ローカルリポジトリの確認
6.6	GUI 設定
7.	LifeKeeper のインストール
7.1	LifeKeeper インストールイメージの転送
7.2	root パスワードの変更
7.3	LifeKeeper のインストール
7.4	GUI ログイン
8.	コミュニケーションパス・リソースの作成41
8.1	コミュニケーションバスの作成
8.2	DataKeeper リソースの作成
8.3	スイッチオーハーのテスト45
8.4	PostgreSQL リソースの作成
8.5	LB Health Check リソースの作成48

8.6	IP リソースの作成	
8.7	リソース依存関係の作成	51
9.	QWK の導入と設定	53
9.1	- NFS ファイル共有の設定	53
9.2	QWK storage モードの導入	54
9.3	フェイルオーバーシナリオ	56
9.4	留意事項	58
10.	お問い合わせ	
11.	免責事項	

改訂履歴

日付	班	変更情報
2023/12/15	第1版	第1版 新規作成

1. 始めに

本ドキュメントに含まれる情報は、公表の日付におけるサイオステクノロジー株式会社の ポリシーに基づいています。サイオステクノロジー株式会社は記載されている内容をお約 束しているわけではありません。また、それらの内容を保証するものでもありません。本ド キュメントは情報提供のみを目的としています。また、記載内容は予告無く変更する場合が あります。予めご了承ください。

2. 本ドキュメント

本ドキュメントは、Microsoft Azure(以下 Azure)環境上の Linux 仮想マシンに LifeKeeper for Linux(以下 LifeKeeper)及び Quorum/Witness Kit(以下 QWK)の storage モードを用いる構成例・構築手順例・検証手順例を記したものです。LifeKeeper の運用における詳細な情報は含まれていません

2.1 対象読者について

本ドキュメントは Azure 上で QWK の storage モードの構築を計画している方が対象 となります。そのため、LifeKeeper の操作や Azure に関する基本的な知識を持っている 技術者を対象としています。

2.2 必要なソフトウェアイメージ

- LifeKeeper for Linux インストールメディア
- Red Hat Enterprise Linux OS イメージ

2.3 製品に関する情報

LifeKeeper for Linux / Single Server Protection 製品に関する詳細は下記リンクより ご参照ください。製品のリリースノートや Recovery Kit の詳細がご確認いただけます。 https://support.us.sios.com/aspx/jpdocs_us_sios_com_home/

また、Recovery Kit の処理概要、脆弱性情報、ライセンスの取得方法等、製品のサポート に関する情報は下記リンクのユーザーポータルでご確認いただけます。 <u>https://lkdkuserportal.sios.jp/hc/ja</u>

3. QWK によるスプリットブレイン対策

本章では HA クラスターで発生するスプリットブレインと呼ばれる現象と、LifeKeeper における対策である QWK の一つ、 storage モードの概要を紹介します。

3.1 スプリットブレインとは

スプリットブレインとは、HA クラスターシステムにおいてハードウェアやノード間通信の 障害により、システムが分断され、1 つのサービスが複数のノードで同時に起動する現象を 指します。具体的には、複数のシステム上で同時に LifeKeeper の階層が In Service に なる状況です。

スプリットブレインが発生すると、各ノードがアプリケーションを制御できると認識して いるため、両ノードでデータへのアクセス、共有ストレージデバイスにデータを書き込もう として、リソースの整合性やデータの一貫性を損なう可能性があります。

3.2 QWK の概要

LifeKeeper に QWK を組み合わせることにより、「スプリットブレイン」の発生リスクを 大幅に軽減しながら、より高い信頼度でフェイルオーバーを実行することができるように なります。

QWK は以下の主要なチェックで構成されます。

- Quorum チェック:通信障害が発生した時、各ノード間で最も多数の合意を得られたノードを Quorum を保持しているとし、リソースを起動することが許可されます。
 Quorum を持っている状態を Quorum チェック成功とします。
 主にスプリットブレインを抑止します。
- Witness チェック: Quorum チェック成功すると実施されます。クラスターには組 み込まれていない第三者ノードを Witness ノードと呼びます。クラスターのノード A と、Witness ノードが障害のあったノードのステータスの確認を行い、同じ意見であ る場合のみ、ノード A のリソースの起動が許可されます。 主にノード間で発生する単純な通信障害から発生するフェイルオーバーを回避します。

Quorum チェックのモードとして、以下の4つが用意されています。

- majority
- tcp_remote
- storage
- none または off

本ドキュメントでは storage モードを使用します。

※他のモードに関しましてはテクニカルドキュメントをご参照ください。 Quorum/Witness

3.3 QWK の storage モードの概要



図 storage モードの標準構成

storage モードはクラスター内のすべてのノードからアクセスできる共有ストレージを Witness デバイスとして用いる Quorum のモードです。共有ストレージを介してノード の情報交換を行いますが、この共有ストレージに書き込まれたノードの情報を QWK オブ ジェクトと呼びます。

Quorum チェックでは共有ストレージにアクセスできることで Quorum チェック成功 と判断します。Witness チェックでは、障害が疑われるノードの QWK オブジェクトを確 認します。一定期間内に更新されていれば障害無し、更新が止まっていれば障害が発生して いると考えます。

両方のチェックが成功すると、クラスターは正常に動作していると判断されます。

4. Azure の環境構築

本ドキュメントにおいて Azure を使用して Virtual Machine を構築する際に必要な設 定について説明します。

4.1 ネットワーク構成図



本ドキュメントで構成する仮想ネットワークは図の通りとなります。

図 4.1-1 ネットワーク構成図

4.2 リソースグループの作成

クラスターを構成するために、リソースグループを作成します。リソースグループは Azure の各種リソースを一括で管理することができます。

リソースグループのページを開き、「作成」をクリックします。
 以降、Azure でリソースを作成する場合は上記の手順を行います。



図 4.2-1 リソースグループ 初期画面

サブスクリプションを選択し、リソースグループに名前を書き込みます。
 ここでは「LKL-QWK-storage」とします。

リソース グループを作成し	ます	
基本 タグ 確認および作成 リソース グループ - Azure ソリューションの関連 ることも、グループとして管理したいリソースのみ 割り当てる方法を決めてください。詳細情報 c	車リソースを保持するコンテナー。リソース グループには、ソリュ- を含めることもできます。 組織にとって最も有用なことに基づい ♂	-ションのすべてのリソースを含め て、リソース グループにリソースを
プロジェクトの詳細		
サブスクリプション* ①		\checkmark
リソース グループ * ①	LKL-QWK-storage	✓
リソースの詳細		
リージョン * ①	(Asia Pacific) Japan East	\checkmark

図 4.2-2 リソースグループの設定画面

3. 作成者がわかるようにタグをつけます。

同様に、以降のリソースの全てにタグをつけます。

ホーム > リソース グルーフ リソース グループ	ったい ひんします しんします しんしょう しんしょ しんしょ		
基本 タグ 確認お Azure リソースをカテゴリに 小文字が区別されず、タグ	よび作成 分けて論理的に整理するため、タグを適 値は大文字と小文字が区別されます。	I用します。タグは、キー (名前)と値で構成されます。タグ 詳細情報 ♂	名は大文字と
名前	值①	リソース	
creator		リソース グループ	
	:	リソース グループ	

図 4.2-3 タグの設定画面

4. 「確認及び作成」タブに移動し、「作成」を押すことでリソースグループの作成を終了 します。



図 4.2-4 リソースグループの作成

4.3 仮想ネットワークの作成

- 1. Azure 画面上部の検索窓から、仮想ネットワークを検索し、作成ボタンを押します。
- 2. 仮想ネットワークの「基本」タブでは以下の表の通り設定を行います。

項目	設定値
リソースグループ	LKL-QWK-storage
仮想ネットワーク名	LKL-QWK-storage-Vnet
地域	(Asia Pacific) Japan East

プロジェクトの詳細		
デプロイされたリソースとコストを管理するためのサブスクリプションを選択します。フォルダーなどのリソース グループ を使用すると、すべてのリソースを整理して管理することができます。		
サブスクリプション *	×	
リソース グループ *	LKL-QWK-storage	
	3/12/21F10x	
インスタンスの詳細		
仮想ネットワーク名 *	LKL-QWK-storage- <u>Vnet</u>	
地域 ① *	(Asia Pacific) Japan East	
	エッジ ゾーンにデプロイ	

図 4.3-1 仮想ネットワーク 基本の設定画面

- 3. 「セキュリティ」タブではデフォルトの設定のまま進めます。
- 4. IPv4 アドレス空間の設定を行います。

設定は以下の表の通りに行います。

項目	設定値
アドレス空間の種類	IPv4
開始アドレス	10.5.0.0
アドレス空間のサイズ	/16 (65536 個のアドレス)

まず、デフォルトのアドレス空間の削除を行い、改めて IPv4 アドレス空間の追加を 行い、表の設定値と同じ値を入れます。

10.5.0.0/16			・ アドレス空間 ・
10.5.0.0	/16 (65,536 個のアドレス	$\gamma \sim 0$	
+ サブネットの追	加		

図 4.3-2 IP アドレス空間の設定画面

5. サブネットを二つ追加します。

サブネットの設定値は表の通りです。

プライベートサブネットとセキュリティに関しましてはデフォルトのまま進めます。

項目	サブネット1	サブネット 2
サブネットの目的	Default	Default
名前	LKL-QWK-storage- subnet1	LKL-QWK-storage- subnet2
開始アドレス	10.5.1.0	10.5.2.0
サイズ	/24 (24 個のアドレス)	/24 (24 個のアドレス)

サブネットの追加 ×		
アドレス空間を選択し、サブネットを構成します。選択したサービスを後で追加する予定の場合は、既定のサブネットをカスタマイズす るか、サブネット テンプレートから選択できます。 詳細情報 ♂		
サブネットの目的 ①	Default	
名前 * ①	LKL-QWK-storage-subnet1	
IPv4		
IPv4 アドレス空間を含める		
IPv4 アドレスの範囲 ①	10.5.0.0/16 くうちょう (65536 アドレス)	
開始アドレス ①	10.5.1.0	
サイズ ①	/24 (256 個のアドレス)	
サブネット アドレスの範囲 ①	10.5.1.0 - 10.5.1.255 (256 アドレス)	

図 4.3-3 サブネットの追加画面

6. 仮想ネットワークを作成します。

4.4 可用性セットの作成

Azure の可用性セットを利用し、インフラストラクチャの冗長設定を行います。

- 1. Azure 画面上部の検索窓から、仮想ネットワークを検索し、作成ボタンを押します。
- 以下の通りに可用性セットの設定を行います。記載していない項目につきましてはデフォルトの値で進めます。

項目	設定値
リソースグループ	LKL-QWK-storage
名前	LKL-QWK-storage-availability
地域	(Asia Pacific) Japan East
マネージドディスクを使用	はい (配置)

プロジェクトの詳細 デプロイされているリソースとコストを管理するサプ し、管理します。	ブスクリプションを選択します。フォルダーのよう	はリソース グループを使用して、すべてのリソースを整理
サブスクリプション* ①		\sim
リソース グループ* ①	LKL-QWK-storage	~
	第17天兄们午内以	
インスタンスの詳細		
名前* ①	LKL-QWK-storage-availability	
地域* ①	(Asia Pacific) Japan East	~
障害ドメイン ①		2
更新ドメイン ①		5
マネージド ディスクを使用 ①	(いいえ (クラシック) (はい (配置))	

図 4.4-1 可用性セットの設定画面

4.5 仮想マシンの作成

本ドキュメントでは仮想マシンを 3 台作成します。Witness/クライアントノード(以下 Witness ノード)とクラスターノード 2 台 (以下ノード A、ノード B) です。

1) Witness ノードの作成

Witness サーバとして使用する仮想マシンを作成します。 また、クラスターノードへ接続するための踏み台サーバと、PostgreSQL のクライア ントとしても使用します。

1. Azure 画面上部の検索窓から、仮想ネットワークを検索し、作成ボタンを押しま

す。そして Azure 仮想マシンを選択します。



図 4.5-1 仮想マシンの作成

2. 以下の通りに「基本」タブの設定を行います。

記載していない項目はデフォルトのまま進めます。

項目	設定値
リソースグループ	LKL-QWK-storage
仮想マシン名	Witness-node
地域	(Asia Pacific) Japan East
可用性オプション	インフラストラクチャ冗長は必要ありません
セキュリティの種類	Standard
イメージ	Red Hat Enterprise Linux バージョンは 9.2(LVM) - x64 Gen 2
サイズ	Standard_B2s -2 vcpu, 4GiBのメモリ
認証の種類	パスワード
ユーザ名	Ikadmin
パスワード	*******(自分で設定した値)
パブリック受信ポート	選択したポートを許可する
受信ポートを選択	SSH (22)

LK for Linux QWK Storage mode on Azure Step by step Guide

ブロジェクトの詳細 テプロイされているリソースとコストを管理するサブ し、管理します。	スクリプションを選択します。フォルダーのようなリソース グループを使用して、	すべてのリソースを整理
サブスクリプション* ①		\sim
リソース グループ* ①	LKL-QWK-storage	~
インスタンスの詳細		
仮想マシン名* ①	Witness-node	~
地域* ①	(Asia Pacific) Japan East	\sim
可用性オプション ①	インフラストラクチャ冗長は必要ありません	\sim
セキュリティの種類 ①	Standard	\sim
イメージ* ①	Red Hat Enterprise Linux 9.2 (LVM) - x64 Gen2 オペアのイメージを完テ LVM の世代の構成	~
	 このイメージは、追加のセキュリティ機能と互換性があります。トラステッド 類に切り替えるには、ここをクリックしてください。 	記動のセキュリティの種
VM アーキテクチャ ①	 ARM64 x64 	
	● Arm64 は、選択したイメージではサポートされていません。	
Azure Spot 割引で実行する ①		
サイズ* ①	Standard_B2s - 2 vcpu 数、4 GiB のメモリ (\$39.71/月) 9へてのジイズを衣示	~

図 4.5-2 基本設定 1

休止状態を有効にする (プレビュー) 🕕		
	1 To enable Hibernation,	you must register your subscription. 詳細情報 ♂
管理者アカウント		
認証の種類 ①	○ SSH 公開≠-	
	● パスワード	
┐━サテ━匁* ①	Ikadmin	
- 7 - 0		
パスワード* ①	•••••	
パスワードの確認 * ①	•••••	×
受信ポートの規則		
パブリック インターネットからアクセスできる仮想 アクセスを指定できます。	マシン ネットワークのポートを選択し	ます。[ネットワーク] タブで、より限定的または細かくネットワーク
パブリック受信ポート* ①	○ なし	
	 選択したポートを許可する 	5
受信ポートを選択 *	SSH (22)	~
 インターネットからのすべてのトラフィックは、既定でブロックされます。受信ポートのルールは、 [VM] > [ネットワーク]ページから変更できます。 		

図 4.5-3 基本設定 2

3. 「ディスク」タブの設定をします。

OS ディスクの種類を Premium SSD から Standard SSD に変更します。



図 4.5-4 ディスクの設定

4. 「ネットワーク」の設定を行います。

以下の表の通りに設定を行います。

他の設定項目はデフォルトの値を使用します。

項目	設定値
仮想ネットワーク	LKL-QWK-storage-Vnet
サブネット	LKL-QWK-storage-subnet1 (10.5.1.0/24)
パブリック IP	新規作成 (デフォルトの値を使用)
VM が削除されたときにパブ	チェックをしわる
リック IP と NIC を削除する	フェックを入れる

ネットワーク インターフェイス		
仮想マシンの作成中に、ユーザー用にネットワーク インターフェイスが作成されます。		
仮想ネットワーク* ①	LKL-QWK-storage-Vnet // 新規作成	
サブネット * ①	LKL-QWK-storage-subnet1 (10.5.1.0/24) > サブネット構成の管理	
バブリック IP 🕕	(新規)Witness-node-ip	
NIC ネットワーク セキュリティ グループ ①	 なし Basic 詳細 	
パブリック受信ポート * ①	 ○ なし ● 選択したポートを許可する 	
受信术——卜を選択 *	SSH (22) V	
	▲ これにより、すべての IP アドレスが仮想マシンにアクセスできるようになります。 これはテスト にのみ推奨されます。 [ネットワーク] タブの詳細設定コントロールを使用して、受信トラフィ ックを既知の IP アドレスに制限するための規則を作成します。	
VM が削除されたときにパブリック IP と NIC を 削除する ①		
高速ネットワークを有効にする ①	選択した VM のサイズは、高速ネットワークをサポートしていません。	
負荷分散		
既存の Azure 負荷分散ソリューションのバックエンド プールにこの仮想マシンを配置できます。詳細情報 🖻		
負荷分散のオブション ①	 なし Azure Load Balancer すべての TCP または UDP ネットワーク トラフィック、ポート フォワーディング、送信フローをサ ポートしています。 アプリケーション ゲートウェイ URL ベースのルーティング、SSL 終了、セッション永続化、Web アプリケーション ファイアウ 	

図 4.5-5 ネットワークの設定

- 「管理・監視・詳細」タブの設定をします。
 LifeKeeper のクラスター構築に必要な設定はないため、設定はスキップします。
 必要に応じて適宜設定してください。
- 6. 仮想マシンを作成します。
- 2) クラスターノードの作成
 - 1. 仮想マシンを作成します。Witness ノードの作成手順と同様に行います。
 - 2. 「基本」タブの設定です。Witness ノードと変更になる設定は赤字で示していま す。記載していない項目につきましてはデフォルトの値で進めます。

項目	設定値
リソースグループ	LKL-QWK-storage
仮想マシン名	Clstr-nodeA
地域	(Asia Pacific) Japan East
可用性オプション	可用性セット
可用性セット	LKL-QWK-storage-availability
セキュリティの種類	Standard
イメージ	Red Hat Enterprise Linux バージョンは 9.2(LVM) - x64 Gen 2
サイズ	Standard_B2s -2 vcpu, 4GiBのメモリ
認証の種類	パスワード
ユーザ名	Ikadmin
パスワード	*******(自分で設定した値)
パブリック受信ポート	なし

3. 「ディスク」タブの設定をします。

OS ディスクの種類を Premium SSD から Standard SSD に変更します。 新しいディスクを作成します。

OS ディスク	
OS ディスク サイズ ①	イメージの既定値 (64 GiB) V
OS ディスクの種類 * ①	Standard SSD (ローカル冗長ストレージ) 違択した VM のサイスは Premium テイスクをサポートします。 IOPS の高い作業負荷には Premium SSD がお勧めです。 Premium SSD ディスクを搭載した仮想マシンは、 99.9% の 接続 SLA に適合します。
VM と共に削除 ①	
キーの管理 ①	ブラットフォーム マネージド キー ~
Ultra Disk の互換性を有効にする ①	Ultra Disk は、選択された VM サイズ Standard_B2s の可用性ソーン 1,2,3 でサポートさ れています。
Witness-node の データ ディスク	
仮想マシンに別のデータ ディスクを追加およます。	び構成したり、既存のディスクを接続したりすることができます。この VM には、一時ディスクも付属してい
L··· 名前	サイズ (… ディスクの種類 ホスト キャッ… VM と共に削除 ①
新しいディスクを作成し接続する 既存	のディスクの接続

図 4.5-6 ディスクの設定

新しいディスクではサイズを変更します。

また、「VM とともにディスクを削除」にチェックを入れます。

18 ©2023 SIOS Technology, Inc.

名前 *	Witness-node_DataDisk_0	
ソースの種類 * ①	なし (空のディスク)	~
サイズ* ①	1024 GiB Dramium SSD L PS サイズを変更します	
キーの管理 ①	プラットフォーム マネージド キー	~
共有ディスクを有効にする	🔘 lili 💽 lilik	
VM と共にディスクを削除		

図 4.5-7 新しいディスクの作成

ストレージの種類は Standard SSD (ローカル冗長ストレージ)、サイズは 16GiB です。

利用可能なディスク サイズとその機能を参照します。 ストレージの種類 ①			
サイズ	パフォーマンス レベル	ン プロビジョニングされ…	プロビジョ
4 GiB	E1	500	60
8 GiB	E2	500	60
16 GiB	E3	500	60
32 GiB	E4	500	60
64 GiB	E6	500	60
128 GiB	E10	500	60
256 GiB	E15	500	60
512 GiB	E20	500	60
1024 GiB	E30	500	60
2048 GiB	E40	500	60
4096 GiB	E50	500	60
8192 GiB	E60	2000	400
16384 GiB	E70	4000	600
32767 GiB	E80	6000	750
 ▲ カスタム ディスク サイズ (GiB) * ① 16 			

図 4.5-8 新しいディスクの設定

「ネットワーク」の設定を行います。
 以下の表の通りに設定を行います。
 他の設定項目はデフォルトの値を使用します。

LK for Linux QWK Storage mode on Azure Step by step Guide

項目	設定値
仮想ネットワーク	LKL-QWK-storage-Vnet
サブネット	LKL-QWK-storage-subnet1 (10.5.1.0/24)
パブリック IP	なし
VM が削除されたときにパブ	チェックをしわる
リック IP と NIC を削除する	テエックを入れる

- 5. 残りの操作は Witness ノードと同様の操作を行い、仮想マシンを作成します。
- 3) クラスターノード B の作成
 - 1. 仮想マシンを作成します。ノード A の作成手順と同様に行います。以下の通りに 設定します。

「基本」タブの設定

項目	設定値
リソースグループ	LKL-QWK-storage
仮想マシン名	Clstr-nodeB
地域	(Asia Pacific) Japan East
可用性オプション	可用性セット
可用性セット	LKL-QWK-storage-availability
セキュリティの種類	Standard
イメージ	Red Hat Enterprise Linux
אר ע	Standard_B2s - 2 vcpu, 4GBのメモリ
認証の種類	パスワード
ユーザ名	lkadmin
パスワード	*******(自分で設定した値)
パブリック受信ポート	なし

「ディスク」タブの設定

項目	設定値
OS ディスクの種類	Standard SSD
ストレージの種類	Standard SSD

Step by step Guide

ディスクのサイズ	16GiB

「ネットワーク」タブの設定

項目	設定値
仮想ネットワーク	LKL-QWK-storage-Vnet
サブネット	LKL-QWK-storage-subnet1 (10.5.1.0/24)
パブリック IP	なし
VM が削除されたときにパブ	エーックをしわる
リック IP と NIC を削除する	フェッフで入れる

4.6 NIC の設定

1. 各 NIC に振られる IP アドレスを静的 IP アドレスに変更します。

まず、対象の仮想マシンの概要画面から、「ネットワーク設定」 -> 「ネットワークインター フェイス」へ進みます。



図 4.6-1 仮想マシンの概要画面

ネットワークインターフェイスの概要画面に移動後、「IP 構成」 -> 「ipconfig1」へ進み ます。

▶ 検索	 《 ()最新の情報に更新 			
	<u> </u>			
アクティビティ ログ	IP の設定			
多 アクセス制御 (IAM)	IP 転送を有効にする ①			
୭୦	仮想ネットワーク	LKL-QWK-storage-	Vnet	
設定 目 IP 構成	サブネット ①	LKL-QWK-storage	e-subnet1 (10.5.1	.0/24) 248 件の空き IP アド 248 件の空き IP ア
🔲 DNS サーバー				
💎 ネットワーク セキュリティ グループ	プライベート IP アドレスとパブリ	リック IP アドレスは、仮想マ	シンのネットワ	ークインターフェイスコン
プロパティ	ーフーに割り当(ることかできま フェイスに必要な数のプライベー	9。Azure の制限の記事に記 トおよびパブリックの IPv4	「軟されている制 アドレスを追加」	限内で、ネットリークイン できます。 詳細情報 d
監視	十 追加 際 フライマリにする			
♀ 分析情報	名前	IP バージョ	種類	プライベート IP アド
■ 警告		10.4		
(前) メトリック	[] ipconfig1	IPv4	ノライマリ	10.5.1.5 (副化)
- 50HK87	4			

図 4.6-2 IP 構成編集画面

続いて、NICに割り当てられている IP を静的にし、割り振るアドレスを入力します。

名前	ipconfig1
IP バージョン	IPv4
	O IPv6
種類	③ プライマリ
) セカンダリ
プライベート IP アドレスの設定	
割り当て	〇 動的
	● 静的
プライベート IP アドレス	10.5.1.11
パブリック IP アドレスの設定	
パブリック IP アドレスを関連付ける	

図 4.6-3 IP アドレス設定画面

上記の操作を全てのノードで行います。静的 IP は以下のように設定します。

ノード	IP アドレス
Clstr-nodeA	10.5.1.11
Clstr-nodeB	10.5.1.12
Witness-node	10.5.1.20

2. 各ノードにデータレプリケーション用の NIC を追加します。

再度対象の仮想マシンの概要画面から、「ネットワーク設定」 -> 「ネットワークインター

フェイス」へ進みます。ここではノードAの場合の設定を行います。

NIC を追加するため、「ネットワークインターフェイスのアタッチ」を押します。

🧟 Clstr-nodeA ネット _{仮想マシン}	フーク設定 ☆ …
▶ 検索 《	① これはプレビューエクスペリエンスです。 <u>フィードバックをお寄せく</u>
📮 概要 🔺	ダ ネットワーク インターフェイスのアタッチ 🕺 ネットワーク インタ-
アクティビティ ログ	
⁸ 2、アクセス制御 (IAM)	ネットワークインターフェイス / IP 構成
ダ タヴ	- clstr-nodea704 (フライマリ) / ipconfig1 (フラ イマリ)
★ 問題の診断と解決	1(3)

図 4.6-4 NIC の追加

追加する NIC を新規作成します。

Sector Clstr-nodeA え 仮想マシン	ペットワーク設定 ☆ …
▶ 検索	« ① これはプレビューエクスペリエンスです。フィードバックをお寄せく
♀ 概要	ネットワーク インターフェイスのアタッチ
アクティビティ ログ	按準可能だえ、トワークノンターフェイフがおりません
𝒫 アクセス制御 (IAM)	ネットワークインターフェイスの作成と接続
90	
★ 問題の診断と解決	

図 4.6-5 新しい NIC の作成

以下の表の通りに設定します。

他の値はデフォルトの値のまま進めます。

項目	設定値
リソースグループ	LKL-QWK-storage
ネットワークインターフェイスの名前	Clstr-nodeA-Vnet
サブネット	LKL-QWK-storage-subnet2 (10.5.2.0/24)
プライベート IP アドレスの割り当て	静的
プライベート IP アドレス	10.5.2.11

プロジェクトの詳細
リソース グループ * ①
LKL-QWK-storage V
<u>新规律成</u>
場所 ①
(Asia Pacific) Japan East \lor
ネットワーク インターフェイス
名前*
Clstr-nodeA-vnic 🗸
「唐朝今」」に、A 〇
LKL-QWK-Storage-vnet
サブネット* 〇
LKL-QWK-storage-subnet2 (10.5.2.0/24)
Bacic
バブリック受信ボート* ①
○ 選択したポートを許可する
受信ボートを選択
1つ以上のボートを選択してください く
インターネットからのすべてのトラフィックは、既定でブロックされます。受信ボートのルールは、
[VM] > [ネットワーク] ページから変更できます。
ゴライベート ID アドレフの利的光イ
CONTRACT CONTRACT
プライベート IP アドレス *
10.5.2.11 🗸
高速ネットワーク ①
無効有効

図 4.6-6 ネットワークインターフェイスの設定

ノードBも上記の手順で NIC を追加します。

追加時の設定値は以下の表の通りになります。

ノードB

項目	設定値
リソースグループ	LKL-QWK-storage
ネットワークインターフェイスの名前	Clstr-nodeB-Vnet
サブネット	LKL-QWK-storage-subnet2 (10.5.2.0/24)
プライベート IP アドレスの割り当て	静的
プライベート IP アドレス	10.5.2.12

4.7 インターナルロードバランサー(ILB)の作成

Azure では IP リソースが保護する仮想 IP アドレスを Azure の仮想ネットワークで認識することができません。この影響で、通常 LifeKeeper for Linux が想定している仮想 IP アドレスによるネットワーク通信を行うことができません。そのため LifeKeeper では ILB の導入を行い、ILB が設定する仮想 IP アドレスをネットワーク通信経路として設定 します。

詳細につきましてはテクニカルドキュメントからご確認いただけます。 <u>https://docs.us.sios.com/spslinux/9.8.0/ja/topic/lifekeeper-specific-configurations-in-azure</u>

- 1. ロードバランサーの作成を作成します。Azure 画面上部の検索窓から、仮想ネットワー クを検索し、作成ボタンを押します。
- 以下の表の通りに「基本」タブの設定をします。記載していない項目につきましてはデ フォルトの値で進めます。

項目	設定値
リソースグループ	LKL-QWK-storage
ロードバランサーの名前	Clstr-nodeB-Vnet

プロジェクトの詳細	
サブスクリプション *	
リソース グループ *	LKL-QWK-storage 新規作成
インスタンスの詳細	
名前 *	LKL-QWK-storage-ILB
地域 *	Japan East
ѕки* ⊙	 Standard ゲートウェイ Basic
種類 * ①	パブリック() 内部
レベル*	 地域 グローバル

図 4.7-1 ロードバランサーの設定

3. 続いて、「フロントエンド IP 構成」タブの設定をします。 フロントエンド IP 構成の追加ボタンを押します。

次の表に従い、フロントエンド IP の設定を行います。

項目	設定値
名前	LKL-QWK-storage-FEIP
仮想ネットワーク	LKL-QWK-storage-Vnet
サブネット	LKL-QWK-storage-subnet1 (10.5.1.0/24)
割り当て	静的
IP アドレス	10.5.1.100
可用性ゾーン	ゾーン冗長

フロントエンド IP 構成の追加	×
名前 *	
LKL-QWK-storage-FEIP 仮想ネットワーク*	_~
LKL-QWK-storage-Vnet (LKL-QWK-storage)	\checkmark
LKL-QWK-storage-subnet1 (10.5.1.0/24)	\sim
割り当て <u>動的</u> 静的	
IP アドレス * 10.5.1.100	~
可用性ゾーン* ① ゾーン冗長	\vee

図 4.7-2 フロントエンド IP 構成の設定

設定を終えたら追加ボタンを押し、フロントエンド IP 構成を追加します。

続いて、バックエンドプールを追加します。
 バックエンドプールの名前を付け、IP 構成の NIC 追加を行います。

バックエンド プールの追加	
名前* 仮想ネットワーク ① バックエンド プールの構成	LKL-QWK-storage-BEP LKL-QWK-storage-Vnet
IP 構成 仮想マシンと仮想マシン スケール セット 内にある必要があります。	○ IP アドレス に関連付けられている IP 構成は、ロード バラン
+ 追加 × 削除 リソース名	リソース グループ 種類

図 4.7-3 バックエンドプールの追加

追加する NIC はノード A とノード B の 10.5.1.0/24 内にある NIC を追加しま

す。

\sim		リソース名	リソース	種類	IP 構成	IP アドレス
~	仮想	見マシン (6)				
		Clstr-nodeA	LKL-QWK	仮想マシン	ipconfig1	10.5.1.11
		Clstr-nodeA	LKL-QWK	仮想マシン	ipconfig1	10.5.2.11
		Clstr-nodeB	LKL-QWK	仮想マシン	ipconfig1	10.5.2.12
		Clstr-nodeB	LKL-QWK	仮想マシン	ipconfig1	10.5.1.12
		Witness-node	LKL-QWK	仮想マシン	ipconfig1	10.5.1.20
		Witness-node	LKL-QWK	仮想マシン	ipconfig1	10.5.2.20

図 4.7-4 NIC の選択

5. 負荷分散規則を追加します。「インバウンド規則」タブから「負荷分散規則の追加」へ 進みます。

「基本」 タブでは、以下の表の通りに設定します。 記載していない項目はデフォルトの 値のまま進めます。

項目	設定値
名前	LKL-QWK-storage-rule
フロントエンド IP アドレス	LKL-QWK-storage-FEIP (10.5.1.100)
バックエンドプール	LKL-QWK-storage-BEP

©2023 SIOS Technology, Inc.

ポート	5432
バックエンドポート	5432
フローティング IP を有効にする	チェック

名前 *	LKL-QWK-storage-rule
IP バージョン*	IPv4
	O IPv6
フロントエンド IP アドレス * ①	LKL-QWK-storage-FEIP (10.5.1.100)
バックエンド プール * ①	LKL-QWK-storage-BEP
高可用性ポート ③	
プロトコル	• тср
	O UDP
ポート*	5432
バックエンド ポート * ①	5432
正常性プローブ * 🗊	既存のプローブはありません
	新規作成
セッション永続化 ①	なし
アイドル タイムアウト (分) * ①	4
TCP リセットを有効にする	
フローティング IP を有効にする 🛈	

図 4.7-5 負荷分散規則の作成

正常性プローブは新規作成をします。以下の表の通りに設定します。

項目	設定値
名前	LKL-QWK-storage-probe
ポート	12345 ¹

¹ 本ドキュメントではポート番号を 12345 とします。お客様の環境に合わせて設定して ください。ここで設定したポート番号は後述する LB Health Check リソース作成に使用 します。

名前 *	LKL-QWK-storage-probe
プロトコル *	ТСР
ポート * 🗊	12345
間隔(秒)* ()	5
使用者 * ()	未使用

図 4.7-6 正常性プローブの作成

以上で Azure 上の操作は終了です。

5.OSの設定

5.1 仮想マシンのログイン

使用する ssh クライアントについて

LifeKeeper の GUI を使用するために、X Window システムを使います。そのため、X Window システムを利用できる ssh クライアントをお選びください。

1. パブリック IP アドレスの確認

Witness ノードのパブリック IP アドレスを確認します。

witness-node _{仮想マシン}	☆ ☆ …	
▶ 検索	《 🔗 接続 ~ ▷ 開始 🤇 再起動 🗌 停止	L ① 休止状態 (九ビュー) 🔀 キャプチャ 💼 削除 ·
₩ 概要	A	1001 He
📄 アクティビティ ログ		JSON CI-
🍫 アクセス制御 (IAM)	リソース クルーフ (移動) LKL-QWK-storage	オペレーティング システム Linux (rhel 9.2)
4 40	状態	サイズ Standard P2c (2 vcpu)教 4 CiP メモリ)
★ 問題の診断と解決	場所	パブリック IP アドレス
接続	Japan East	
∅ 接続	リノスクリノンヨン (参加)	1仮想不ットワーク/リフネット LKL-QWK-storage-Vnet/LKL-QWK-sto····
S Bastion	サブスクリプション ID 340f215d-7ad3-49ae-a2fd-0ba1cb3c…	DNS 名 <u>未構成</u>
ネットワーク		正常性の状態

図 5.1-1 パブリック IP アドレスの確認

2. Witness ノードヘログイン

ssh クライアントを利用して Witness ノードへ ssh を用いてログインします。 ssh ログインにはパブリック IP、ユーザ名、パスワードが要求されるので入力しログイン します。

3. クラスターノードヘログイン

Witness ノードヘログインができたら、クラスターノードへ ssh ログインができること を確認します。

クラスターノードへ ssh 接続

\$ ssh XXX.XXX.XXX.XXX

パスワードが要求されるので入力します。

接続が確認出来たらログアウトします。

\$ exit

両方のノードでログインできることを確認してください。

5.2 ssh の設定

1. 公開鍵認証の設定

ノード A とノード B へ公開鍵認証が可能となるよう設定を行います。

Witness ノードでキーペアを作成します。

\$ ssh -keygen -t rsa

上記コマンド実行後、/home/lkadmin/.ssh/ 配下に秘密鍵 id_rsa と id_rsa.pub が生成されます。

続いて、クラスターノードへ公開鍵を転送します。

\$ ssh-copy-id lkadmin@XXX.XXX.XXX.XXX

2. sshd_config の編集

パスワード認証の無効化と X11 Forwarding の有効化を行います。

/etc/ssh/sshd_config の一部を以下のように変更します。

公開鍵認証の有効化 (すべてのノードで実施します。)

変更前	変更後
PasswordAuthentication yes	PasswordAuthentication no

X11 Forwarding の有効化 (すべてのノードで実施します。)

変更前	変更後
X11Forwarding no	X11Forwarding yes

最後にすべてのノードで sshd サービスを再起動します。

systemctl restart sshd

X Server を有効化以降は クラスターノードへ ssh 接続する際に、-X オプションをつけ

ることで、クラスターノード上の GUI 画面をローカルの画面で表示することができます。

\$ ssh -X lkadmin@XXX.XXX.XXX.XXX

5.3 NIC アドレスの固定化

Azure インフラストラクチャ更新に伴って NIC アドレスが更新されることを防ぐため、 NIC アドレスを固定します。

クラスターノードで実施します。

1. NIC の情報の確認

仮想マシンの NIC 情報を表示し、各インターフェース名と MAC アドレスを控えます。

[lkadmin@Clstr-nodeA ~]\$ ifconfig eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500 inet 10.5.1.11 netmask 255.255.255.0 broadcast 10.5.1.255 inet6 fe80::20d:3aff:fecc:fbda prefixlen 64 scopeid 0x20<link> ether 00:0d:3a:cc:fb:da txqueuelen 1000 (Ethernet) RX packets 67109 bytes 14595447 (13.9 MiB) RX errors 0 dropped 0 overruns 0 frame 0 TX packets 82566 bytes 19457398 (18.5 MiB) TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0 eth1: flags=4163<UP, BROADCAST, RUNNING, MULTICAST> mtu 1500 inet 10.5.2.11 netmask 255.255.255.0 broadcast 10.5.2.255 inet6 fe80::a2f9:e168:ac0a:bcf8 prefixlen 64 scopeid 0x20<link> ether 60:45:bd:64:e0:4c txqueuelen 1000 (Ethernet) RX packets 2 bytes 762 (762.0 B) RX errors 0 dropped 0 overruns 0 frame 0 TX packets 30 bytes 2584 (2.5 KiB) TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0 lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536 inet 127.0.0.1 netmask 255.0.0.0 inet6 ::1 prefixlen 128 scopeid 0x10<host> loop txqueuelen 1000 (Local Loopback) RX packets 0 bytes 0 (0.0 B) RX errors 0 dropped 0 overruns 0 frame 0 TX packets 0 bytes 0 (0.0 B) TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

2. 設定ファイルの編集

デフォルトでは eth0 のネットワーク設定ファイルしかないため、 eth1 のネットワーク 設定ファイルを作成します。

cd /etc/sysconfig/network-scripts/
cp ifcfg-eth0 ifcfg-eth1

eth0、 eth1 それぞれの設定ファイルを編集します。 HWADDR には、前の手順で控えた 値を使用します。

eth0 の例

Created by cloud-init on instance boot automatically, do not edit.
#
AUTOCONNECT_PRIORITY=999
BOOTPROTO=dhcp
DEVICE=eth0

HWADDR=00:0d:3a:cc:fb:da ONBOOT=yes TYPE=Ethernet USERCTL=no

eth1 の例

```
# Created by cloud-init on instance boot automatically, do not edit.
#
AUTOCONNECT_PRIORITY=999
BOOTPROTO=dhcp
DEVICE=eth1
HWADDR=00:0d:3a:cc:fb:da
ONBOOT=yes
TYPE=Ethernet
USERCTL=no
```

5.4 SELinux の無効化

SELinux が enforcing モードの場合、LifeKeeper は動作しません。

また、必要な場合を除いて、permissive モードは推奨されません。

そのため、以下のコマンドを実行し、SELinux を無効化させます。

クラスターノードで実施します。

sed -i 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/selinux/config 変更は再起動後に適用されます。

LifeKeeper上で SELinux を Permissive モードで動作させる事を検討している場合は以下のページをご参照ください。

LifeKeeper for Linux v9.8.0 テクニカルドキュメンテーション - インストール前の要件 https://docs.us.sios.com/spslinux/9.8.0/ja/topic/installing-the-software

5.5 ファイアウォールの無効化

このセクションでは、SIOS Protection Suite for Linux をインストールするために、 firewalld サービスを無効にします。ファイアウォールは有効なままにしておくことができ ますが、SIOS Protection Suite と保護するアプリケーションに必要なポートを設定する 必要があります(ここでは示していません)。

以下のコマンドを実行し、ファイアウォールを無効化させます。

すべてのノードで実施します。

systemctl disable firewalld.service --now

5.6 ホスト名の名前解決

ノード間の名前解決のために /etc/hosts ファイルにホスト名と IP アドレスを追記しま

す。/etc/hosts に以下を追加します。

すべてのノードで実施します。

10.5.1.11 Clstr-nodeA 10.5.1.12 Clstr-nodeB 10.5.1.20 Witness-node

6. ローカルリポジトリの設定

LifeKeeper のインストール時に 、各 OS のリポジトリから依存関係にあるソフトウェア のインストールも同時に行います。Azure 上の Red Hat 仮想マシンは Azure の RHUI を リポジトリとして利用することができますが、オフラインノード上からは Azure の RHUI にアクセスできません。

そのため、OS のインストールイメージを使用しローカルリポジトリを設定することで、オ フラインノード上でも LifeKeeper のインストールを行うことができます。

6.1 /home の拡張

Azure では RHEL 仮想マシンを作成後に明示的に /home の拡張を行う必要があります。 以下のコマンドを実行してください。**すべてのノードで実施します。**

```
$ cd /
$ sudo su -
# cd /
# umount /home
# lvextend -L +16384 /dev/mapper/rootvg-homelv
# mount /home
# xfs growfs /dev/mapper/rootvg-homelv
```

6.2 OS イメージのマウント

次に、RHELのインストールイメージをローカル PC からクラスターノードに転送します。 今回、仮想マシンのイメージに使用した OS は Red Hat Enterprise Linux 9.2 のため、 Red Hat 公式から rhel-9.2-x86_64-dvd.iso を事前にダウンロードしておき、 始めに踏み台となる Witness サーバへ転送、その後クラスターノードに転送します。 転送には主に scp を用います。

OS イメージはマウントする必要があります。 fstab を編集することで、OS の起動時に自動的に OS イメージをマウントされ、ローカ ルリポジトリを楽に使用することができます。以下の手順はすべてのノードで実施します。 マウントポイントを作成します。

mkdir /media/cdrom

/etc/fstab を編集し、以下を追加します。

/home/lkadmin/rhel-9.2-x86_64-dvd.iso /media/cdrom iso9660 ro,loop 0 0

マウントを行います。

mount /media/cdrom

正常に実行されたら、fstab の編集は終了になります。

6.3 ローカルリポジトリの作成

ローカルリポジトリを以下の設定で /etc/yum.repos.d/rhel-dvd.repo に作成します。

すべてのノードで実施します。

```
[rhel-dvd-BaseOS]
name=Red Hat Enterprise Linux 9.2-x86_64-DVD
baseurl=file:///media/cdrom/BaseOS/
enabled=1
gpgcheck=1
gpgkey=file:///media/cdrom/RPM-GPG-KEY-redhat-release
[rhel-dvd-AppStream]
name=Red Hat Enterprise Linux 9.2-x86_64-DVD
baseurl=file:///media/cdrom/AppStream/
enabled=1
gpgcheck=1
gpgkey=file:///media/cdrom/RPM-GPG-KEY-redhat-release
```

6.4 rh-cloud-base.repo の無効化

ローカルリポジトリの設定が終了したため、Azureの RHUIを以下のコマンドを使用し、

無効にします。

すべてのノードで実施します。

sed -i 's/enabled=1/enabled=0/g' /etc/yum.repos.d/rh-cloud-base.repo

6.5 ローカルリポジトリの確認

ローカルリポジトリが正しく設定されているか確認します。

以下のコマンドでパッケージマネージャーが使用しているリポジトリのリストを確認できます。

正常にローカルリポジトリの設定が完了していると、以下のように表示されます。

<pre>[lkadmin@Clstr-nodeA ~]\$ yum</pre>	repolist
repo id	repo name
rhel-dvd-AppStream	Red Hat Enterprise Linux 9.2-x86_64-DVD
rhel-dvd-BaseOS	Red Hat Enterprise Linux 9.2-x86_64-DVD

6.6 GUI 設定

全ノード上で GUI の ssh 転送に必要なソフトウェアのインストールを行います。 以下のコマンドを実行し、X11 関連パッケージをインストールします。

すべてのノードで実施します。

dnf install xhost xauth

また、LifeKeeperの管理を root ユーザで行うため、/root/.Xauthority に GUI が使用

する DISPLAY 変数を格納する必要があります。

そのため、以下のコマンドを実行します。

すべてのノードで実施します。

\$ echo "sudo xauth add \\$(xauth -f /home/lkadmin/.Xauthority list|tail -1)"\
>> .bashrc

ここまで終了したら、LifeKeeper のインストールの前にすべてのノードを再起動します。

7. LifeKeeper のインストール

ここでは LifeKeeper のインストール方法を説明します。

7.1 LifeKeeper インストールイメージの転送

6.2 と同様に、クラスターノードに LifeKeeper のインストールイメージをローカル PC から転送します。

7.2 root パスワードの変更

LifeKeeper の GUI 表示には root パスワードが必要となるため、ルートパスワードの設 定を行います。 パスワードをクラスター全体で共通にすることで、LifeKeeper の GUI ログインを1回で 済ませることができます。 #su -

```
# passwd
Changing password for user root.
New password:<新しいパスワード>
Retype new password:<新しいパスワード(確認)>
passwd: all authentication tokens updated successfully.
```

7.3 LifeKeeper のインストール

LifeKeeper のインストールを行います。 詳しくは LifeKeeper for Linux v9.8.0 スタートアップガイドを参照してください。 スタートアップガイド

以下の操作はクラスターノード両方で行います。

インストールメディアに格納されている setup スクリプトを実行します。
 実行時、ファイアウォールが無効になっているにも関わらす、ファイアウォールの警告
 文が表示されることがありますが、無視してそのまま進めてください。

2. インストール画面では以下の項目のチェック欄 [] にチェックを入れます。

[] Use Quorum / Witness Functions
Recovery Kit Selection Menu
- Networking> [] LifeKeeper LB Health Check Kit
- Database> [] LifeKeeper PostgreSQL RDBMS Recovery Kit
- Storage> [] DataKeeper for Linux

3. License キーのインストールを行います。

```
# /opt/LifeKeeper/bin/lkkeyins <ライセンスキーのパス>
```

4. ブロードキャスト Ping を無効化します。

Azure では IP リソースが保護する仮想 IP アドレスを Azure の仮想ネットワーク で認識することができません。そのため /etc/default/LifeKeeper で以下のパラメー ターを変更します。

変更前	変更後
NOBCASTPING=0	NOBCASTPING=1

5. 以下のコマンドで LifeKeeper を起動します。

/opt/LifeKeeper/bin/lkstart

LifeKeeperの管理に使用するコマンドの実行ファイルは /opt/LifeKeeper/bin に LifeKeeperの管理コマンドのレファレンスマニュアルは /opt/LifeKeeper/man に それぞれ保存されています。

そのため、以下のコマンドを実行することで、次回以降の root ログイン以降、絶対パスで コマンドを実行する必要がなくなります。

echo "PATH=\$PATH:/opt/LifeKeeper/bin" >> /root/.bash_profile
echo "MANPATH=\$MANPATH:/opt/LifeKeeper/man" >> /root/.bash_profile
echo "export PATH MANPATH" >> /root/.bash_profile

7.4 GUI ログイン

IkGUIapp コマンドを使用し、LifeKeeper の GUI ヘログインします。

/opt/LifeKeeper/bin/lkGUIapp &

ユーザ名とパスワードが要求されます。ユーザ名は root、パスワードは 7.2 で設定した ものを入力します。

🛃 LifeKeeper GUI@Clstr-no	odeB	_		×	
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>H</u> elp					
			. 2	8	
•					
Hierarchies					
None Defined	Clstr-nodeB				
< Clstr-nodeB: Updating	g server state to: alive		< Clstr-nodeB: Updating server state to: alive		

ログインに成功すると以下の画面が表示されます。

図 7.4-1 LifeKeeper GUI ログイン画面

8. コミュニケーションパス・リソースの作成

各ノードを LifeKeeper のコミュニケーションパスでつなぎ、クラスターを構築し、リソー スを保護します。本ドキュメントでは PostgreSQL データベースのデータを DataKeeper で保護します。稼働系ディスクへ書き込みが行われた際は待機系のディスク と同期し、同様のデータになります。PostgreSQL へのアクセスは仮想 IP アドレス 10.5.1.100 を通して行われます。

システム構成図は以下の通りになります。



図 8-1 システム構成図

8.1 コミュニケーションパスの作成

1. GUI 画面の赤枠をクリックし、コミュニケーションパスの作成画面に移ります。



下記の表の通りに設定します。

項目	設定値
Local Server	Clstr-nodeA
Remote Server(s)	Clstr-nodeB
Device Type	ТСР
Local IP Address(es)	10.5.1.11
Remote IP Address	10.5.1.12
Priority	1

同様に、もう一つコミュニケーションパスを作成します。

項目	設定値
Local Server	Clstr-nodeA
Remote Server(s)	Clstr-nodeB
Device Type	ТСР
Local IP Address(es)	10.5.2.11
Remote IP Address	10.5.2.12
Priority	2

コミュニケーションパスを二本作成すると、以下のような状態になります。



図 8.1-2 コミュニケーションパス作成後

8.2 DataKeeper リソースの作成

仮想マシンを作成した際に追加したディスクを DataKeeper で保護します。 データディスクが共有でなくとも各ノードのデータを同期することで、フェイルオーバー が発生した場合でも整合性のあるデータにアクセスすることができます。

 リソース作成ウィザードを起動します。
 以下の赤枠のボタンもしくは Edit -> Server -> Create Resource Hierarchy をク リックします。



図 8.2-1 リソース作成

2. 以下の通りに設定を進めます。

Create Resource Wizard

項目	設定値
Please Select Recovery Kit	Data Replication
Switchback Type	intelligent
Server	Clstr-nodeA
Hierarchy Type	Replicate New Filesystem
Source Disk	/dev/sda (16.0 GB) ²
New Mount Point	/data
New Filesystem Type	xfs
Data Replication Resource Tag	datarep-data
File System Resource Tag	/data
Bitmap File	/opt/LifeKeeper/bitmapdata ³
Enable Asynchronous Replication	no

確認画面に移ります。「Create」をクリックしてください。

作成が完了すると、「Next」のボタンが表示されるので、クリックして次に進みます。

3. Pre Extend ウィザードの設定を行います。

クラスターノードヘリソースの拡張を行います。

以下の通りに設定を行います。

Pre-Extend Wizard

項目	設定値
Target Server	Clstr-nodeB
Switchback Type	intelligent
Template Priority	1
Target Priority	10

Pre Extend checks were successful と表示されれば成功です。

² お客様環境により表示名が変わります。

³ デフォルトの値となっています。

「Next」 から次へ進みます。

4. Extend ウィザードの設定を行います。

以下の通りに設定を行います。

項目	設定値
Target Disk	/dev/sda (16.0 GB)
Data Replication Resource Tag	datarep-data
Bitmap File	/opt/LifeKeeper/bitmapdata
Replication Path	10.5.2.11/10.5.2.12
Mount Point	/data
Root Tag	/data

Hierarchy successfully extended と出力されたら、リソースの拡張も正常に終了です。 Finish を押して DataReplication リソースの作成は以上になります。

DataReplication リソースを作成した後すぐは、データレプリケーションリソースが以下 のように Wait to Resync (再同期中)という表示になります。Wait to Resync から Target の表示に変わるまで少しお待ちください。



図 8.2-2 DataReplication リソースの再同期と GUI 表示

作成後は以下のような画面になります。

LifeKeeper GUI@Clstr-noo	leA		- 0	×
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>H</u> elp				
٩ 🔇 🛃 🛃	0 🔊 🔺 🗼	0		æ
Hierarchies	Clstr-nodeA	cl	str-nodeB	
r- 📀 /data	Active	1	StandBy	10
🗌 🗌 🧭 datarep-dati	Source	1	Target	10

図 8.2-3 DataKeeper リソース作成後の GUI 画面

8.3 スイッチオーバーのテスト

スイッチオーバーのテストを行います。

スイッチオーバーのテストは各リソースの作成時に必ず実施できることを確認してくださ い。

待機系の /data リソースを左クリックし、「In Service」ボタンを押し、正常にスイッチ オーバーができるか確認します。



図 8.2-4 スイッチオーバーのテスト

8.4 PostgreSQL リソースの作成

1. PostgreSQL サーバおよびクライアントのインストールをします。 以下のコマンドで PostgreSQL をインストールします。

クラスターノードで実施します。

dnf install postgresql-server

また、接続の確認を行う Witness ノードでは、以下のコマンドで PostgreSQL クラ

イアントをインストールしてください。

dnf install postgresql

PostgreSQLのデータディレクトリを DataKeeper が保護しているディスク上に作成 します。

また、ディレクトリの所有者を postgres ユーザに変更します。

稼働系ノードがノード A であることを確認し、以下のコマンドを実行します。

稼働系のみで実施します。

mkdir -m 755 -p /data/pgsql/data # chown -R postgres:postgres /data/pgsql

以上でデータディレクトリの作成は終了です。

2. データベースの初期化を行います。

postgres ユーザになり、データベースの初期化を行います。

稼働系のみで実施します。

su - postgres
\$ initdb -D /data/pgsql/data

3. PostgreSQL の設定変更を行います。

稼働系のみで実施します。

以下の通りに変更します。

/data/pgsql/data/pg_hba.conf の設定

変更前	変更後
<pre># IPv4 local connections:</pre>	<pre># IPv4 local connections:</pre>
host all all 127.0.0.1/32 trust	host all all 10.5.0.0/16 trust
<pre># replication privilege.</pre>	<pre># replication privilege.</pre>
local replication all trust	local replication all trust
host replication all 127.0.0.1/32 trust	host replication all 10.5.0.0/16 trust

/data/pgsql/data/postgresql.conf の設定

変更前	変更後
# - Connection Settings -	# - Connection Settings -
<pre>#listen_addresses = 'localhost'</pre>	listen_addresses = '*'
# port = 5432	port = 5432

4. PostgreSQL デーモンの起動と確認をします。

稼働系と Witness ノードで実施します。

ノード A にて、以下のコマンドで PostgreSQL を起動します。

\$ pg_ctl start -D /data/pgsql/data -p "-p 5432" -w

稼働系では以下のコマンドで PostgreSQL の起動を確認できます。

(postgres ユーザで実施してください)

\$ psql -l

また、Witness ノード (PostgreSQL クライアント) からは以下のコマンドで正常に 接続できることを確認します。

\$ psql -h 10.5.1.11 -U postgres -d postgres

正常に接続ができた場合、以下のプロンプトが返ってきます。

psql (13.10) Type "help" for help.

postgres=# q

5. PostgreSQL リソースの作成を行います。

DataReplication リソースを作成時と同様に、Create Resource Hierarchy を選択 します。

以下の表に従ってリソースの作成を進めます。

Create Resource Wizard

項目	設定値	
Please Select Recovery Kit	PostgreSQL Database	
Swithchback Type	intelligent	
Server	Clstr-nodeA	
PostgreSQL Executable Location	/usr/bin	
PostgreSQL Client Executable Location	/usr/bin/psql	
PostgreSQL Administration Executable	/usr/bin/pg_ctl	
Location		
PostgreSQL Data Directory	/data/pgsql/data	
PostgreSQL Port	5432	
PostgreSQL Socket Path	/var/run/postgresql/.s.PGSQL.5432	
Enter Database Administrator User	postgres	
PostgreSQL Logfile	/tmp/pgsql-5432.lk.log	
PostgreSQL Database Tag	pgsql-5432	

Pre-Extend Wizard

項目	設定値
Target Server	Clstr-nodeB
Switchback Type	intelligent
Template Priority	1
Target Priority	10

Extend Wizard

項目	設定値
PostgreSQL Executable Location	/usr/bin
PostgreSQL Database Tag	pgsql-5432

正常に PostgreSQL リソースが追加されると、GUI 上で PostgreSQL が追加されます。

6. スイッチオーバーのテストを行います。

正常に LifeKeeper で保護できていることを確認します。

8.5 LB Health Check リソースの作成

LB Health Check Kit は、ロードバランサーの負荷分散対象インスタンスへのヘルス チェックプローブを待ち受けて応答する仕組みを提供します。

導入することで IP リソースによる仮想 IP アドレスの切り替えが使用可能となります。

1. リソースの作成を行います。

これまでのリソース作成時と同様に、Create Resource Hierarchy を選択します。 以下の表に従ってリソースの作成を進めます。

項目	設定値
Please Select Recovery Kit	LB Health Check
Swithchback Type	intelligent
Server	Clstr-nodeA
Reply daemon Port	12345 ⁴
Reply daemon message	(何も入力しません)
LB Health Check Resource Tag	lbhc-12345

Create Resource Wizard

Pre-Extend Wizard

項目	設定値
Target Server	Clstr-nodeB
Switchback Type	intelligent
Template Priority	1
Target Priority	10

^{4 4.7} で正常性プローブのポート番号で 12345 以外の値を設定した方は、正常性プロー

ブで設定した値を入力してください。

Extend Wizard

項目	設定値
LB Health Check Resource Tag	lbhc-12345

- スイッチオーバーのテストを行います。
 正常に LifeKeeper で保護できていることを確認します。
- 3. Azure ポータル上から LB Health Check リソースの動作を確認することができま す。

ロードバランサー「LKL-QWWK-storage-ILB」の概要ページから、「分析情報」-> 「Show Metrics Pane」へ進みます。

LKL-QWK-storag u-۴ (رجی)-	je-	ILB
▶ 検索	«	$\rightarrow \overline{*}$
	*	へ基
📄 アクティビティ ログ		リソース
Pg アクセス制御 (IAM)		場所
タグ		サブス
★ 問題の診断と解決		サブス
		SKU
設定		タグ (1
□ フロントエンド IP 構成		もっと見
🧕 バックエンド プール		
涎 負荷分散規則		
🃓 インバウンド NAT 規則		
プロパティ		
<u> </u> しック		
監視		
♥ 分析情報		
185-1 ロードバランH		 सम्बद्धाः

LKL-QWK-storage-ILB 分析情報 ☆ … □-F /(5>サ-	
🔎 検索 🛛 New Support Ticket 🏄 Show Metrics Pane 🕀 View Detail	ed Metrics

図 8.5-2 ロードバランサー分析情報画面

バックエンド IP アドレスごとの正常性プローブの状態を確認します。 スイッチオーバーが正常に動作していると、図のように切り替わりが確認できます。



図 8.5-3 正常性プローブ

8.6 IP リソースの作成

PostgreSQL への接続に使用できる仮想 IP アドレスを作成、保護します。フェイル オーバーが発生しても、PostgreSQL への接続先 IP アドレスを変えることなくアク セスすることが可能になります。

1. リソースの作成を行います。

これまでのリソース作成時と同様に、Create Resource Hierarchy を選択します。 以下の表に従ってリソースの作成を進めます。

項目	設定値
Please Select Recovery Kit	IP
Swithchback Type	intelligent
Server	Clstr-nodeA
IP Resource	10.5.1.100 ⁵
Netmask	255.255.255.255
Network Interface	eth0
IP Resource Tag	ip-10.5.1.100

Create Resource Wizard

Pre-Extend Wizard

項目	設定値
Target Server	Clstr-nodeB
Switchback Type	intelligent
Template Priority	1
Target Priority	10

Extend Wizard

項目 設定値

⁵ 4.7 フロントエンド IP で設定した IP を入力します。

IP Resouorce	10.5.1.100
Netmask	255.255.255.255
Network Interface	eth0
IP Resource Tag	ip-10.5.1.100

2. スイッチオーバーのテストを行います。

正常に LifeKeeper で保護できていることを確認します

8.7 リソース依存関係の作成

各リソースの依存関係を定義することで、リソース階層を作成します。

IP リソース作成直後は以下の図のようなリソースの関係になっています。

Hierarchies		Ristr-nodeA			Clstr-nodeB	
┌ 📀 ip-10.5.1.100		Active	1	V	StandBy	10
— 📀 lbhc-12345	0	Active	1	V	StandBy	10
🕂 📀 pgsql-5432		Active	1	V	StandBy	10
- 📀 /data		Active	1	V	StandBy	10
🗆 🌍 datarep-data		Source	1	V	Target	10

図 8.7-1 IP リソース作成直後のリソース関係

これは、IP リソース、LB Health Check リソース、PostgreSQL リソースに依存関係が なく、それぞれが別のノードで動作し、サービスの提供ができない恐れがあります。そのた め、リソース間に適切な依存関係を作成する必要があります。

本ドキュメントでは PostgreSQL リソースの配下に IP リソースと LB Health Check リ ソースを配置します。

1. IP リソースに依存関係を定義します。

GUI 上で 「Edit」->「Resource」->「Create Dependency」を選択します。 以下の表に従い、設定します。

項目	設定値
Server	Clstr-nodeA
Parent Resource Tag	PostgreSQL リソースのタグ名
Child Resource Tag	IP リソースのタグ名

- LB Health Check リソースに依存関係を定義します。
 上記と同様の手順で行います。子リソースの定義の際に、LB Health Check リソースの夕グ名を指定してください。
- 3. 1と2の手順を完了させると、以下の図のようにリソースの依存関係が作成できます。



図 8.7-1 完成したリソース依存関係

- PostgreSQL にすべてのリソースとの依存関係を作成したので、スイッチオーバーの テストを行います。PostgreSQL を待機系で「In Service」にします。最上位のリソー スを起動する際は、配下のリソースも立ち上がるので、全リソースが待機系で起動する ことになります。
- クライアントから PostgreSQL への接続確認をします。
 以下のコマンドをクライアントノードから使用し、正常に PostgreSQL に接続できる か検証します。

\$ psql -h 10.5.1.100 -U postgres -d postgres

9. QWK の導入と設定

スプリントブレイン対策として、QWK を導入します。

本ドキュメントでは Quorum モードに storage モードを使用します。

storage モードはクラスター内の全てのノードからアクセスできる共有ストレージを用います。

Witness ノードを NFS サーバとすることで、共有ストレージとみなします。

9.1 NFS ファイル共有の設定

 Witness ノードを NFS サーバとして構築します。 NFS 構築するには必要なパッケージをインストールする必要があります。 パッケージは NFS サーバと NFS クライアントの機能を含んでいるため、すべての ノードでインストールします。 以下のコマンドを入力し、インストールします。
 # dnf install nfs-utils

2. NFS サーバ側の設定を行います。

Witness ノードで実施してください。

NFS マウントするディレクトリを作成します。

mkdir /root/qwkshare

/etc/expots ファイルを作成します。

このファイルはリモートホストにどのファイルシステムをエクスポートするかを制御し、オプションを指定します。

/etc/expots を新規で作成し、以下を追加します。

/root/qwkshare Clstr-nodeA(rw,no_root_squash,sync,no_wdelay)
/root/qwkshare Clstr-nodeB(rw,no_root_squash,sync,no_wdelay)

このファイルを反映します。

以下のコマンドを入力し、反映されたことを確認します。

exportfs -v

以下のコマンドを入力します。

exportfs -a

以下のコマンドを入力し、反映されたことを確認します。

exportfs -v

確認ができたら、NFS サーバのサービスを起動します。

また、OS 起動時に自動でサービスを開始するよう設定します。

systemctl start nfs-server
systemctl enable nfs-server

3. NFS クライアント側の設定を行います。

クラスターノードで以下の操作を実施します。

マウントポイントの作成を行います。

mkdir /mnt/qwkmnt

作成したマウントポイントに、NFS サーバのディレクトリをマウントします。

OS 起動時に自動マウントするために /etc/fstab を編集し、以下を追加します。

都合上改行していますが、改行は無視して入力してください。

Witness-node:/root/qwkshare /mnt/qwkmnt nfs
soft,timeo=20,retrans=1,noac 0 0

マウントします。

mount /mnt/qwkmnt

マウントが成功しているか確認します。

mount | grep qwkmnt

ノード A の場合、成功していると以下の情報が表示されます。

Witness-node:/root/qwkshare on /mnt/qwkmnt type nfs4
 (rw,relatime,sync,vers=4.2,rsize=524288,wsize=524288,namlen=255,acreg
 min=0,acregmax=0,acdirmin=0,acdirmax=0,soft,noac,proto=tcp,timeo=20,r
 etrans=1,sec=sys,clientaddr=10.5.1.11,local_lock=none,addr=10.5.1.20)

両方のノードでマウントしたことを確認します。

9.2 QWK storage モードの導入

1. クラスターノードの /etc/default/LifeKeeper の内容を編集します。

クラスター両方のノードで以下の操作を実施します。

以下の内容を変更します。

変更前	変更後
QUORUM_MODE=majority	QUORUM_MODE=storage
WITNESS_MODE=remote_verify	WITNESS_MODE=storage

さらに、以下の内容を追加します。

QWK_STORAGE_TYPE=file QWK_STORAGE_HBEATTIME=6 QWK_STORAGE_NUMHBEATS=4 QWK_STORAGE_OBJECT_Clstr_nodeA=/mnt/qwkmnt/Clstr_nodeA QWK_STORAGE_OBJECT_Clstr_nodeB=/mnt/qwkmnt/Clstr_nodeB

変更していないパラメーター「Quorum_LOSS_ACTION」は Quorum が失われた場 合の動作を指定しています。本ドキュメントではデフォルトとなっており、その値は 「fastkill」です。Quorum を喪失した場合、そのノードは直ちに停止のアクションを 取ります。

※各パラメーターに関しましてはテクニカルドキュメントをご参照ください。

Quorum パラメーター一覧

2. クラスター両方のノードで初期化を行います。

ノード A で以下のコマンドを入力してください。

[root@Clstr-nodeA ~]# qwk_storage_init

すると、以下のように表示されます。

ok:	LifeKeeper is running.
ok:	The LifeKeeper license key is successfully installed.
ok:	QWK parameter is valid.
	QWK object of /mnt/qwkmnt/Clstr_nodeA is not yet avail.
ok:	The path of QWK object is valid.
ok:	<pre>down: /opt/LifeKeeper/etc/service/qwk-storage: 8159s</pre>
ok:	Initialization of QWK object of own node is completed.

上記が表示された状態でもう一つのノードについても初期化を行います。

[root@Clstr-nodeB ~]# qwk_storage_init

Successful と表示されれば成功です。

これで QWk の設定が完了しました。

3. 各ノード上の QWK オブジェクトが、指定された間隔で更新されていることを確認し ます。 /mnt/qwkmnt/Clstr_nodeA の中身を確認します。

/mnt/qwkmnt/Clstr_nodeB も同様の内容になっているか確認します。

なお、両ノードのタイムスタンプは同じになるとは限りません。

9.3 フェイルオーバーシナリオ

実際にサーバに障害を起こし、フェイルオーバーを誘発させます。

本ドキュメントでは稼働系ノード A に通信障害を発生させ、他ノードとの通信を切断します。

この場合、ノード A は以下の動作を行います。

- 1. 他ノードとの通信が切断されていることを検知します。
- Witness ノードの両方と通信ができないため、共有ストレージにアクセスすることが できません。そのため Quorum チェックに失敗し、Quorum を喪失していることを 検知します。
- 3. QUORUM_LOSS_ACTION を実行します。本ドキュメントでは LifeKeeper とリ ソースが停止します。

ノード B は以下の動作を行います。

- 1. ノード A とのコミュニケーションパスが全断していることを検知します。
- 2. Witness ノードとの通信が可能、かつ共有ストレージにアクセスできるため、自ノードが Quorum を得ることになります。
- 障害が起こったとされるノード A の情報を Witness ノードにも問い合わせ、ノード A の状態に関して Witness ノードと自ノードで同意見であることを確認し、 Witness チェックが成功となります。
- Witness チェックが成功となったことでリソース起動の許可を得て、フェイルオー バーを開始し、保護対象リソースがノード B 上で起動されます。

では実際にフェイルオーバーを誘発させます。

リソースが全てノード A にあることを確認します。

Hierarchies	🤯 Clstr-nodeA			Clstr-nodeB	
←	Active	1		StandBy	10
- 📀 /data	Active	1		StandBy	10
L 📀 datarep-data	Source	1	•	Target	10
— 🧭 ip-10.5.1.100	Active	1		StandBy	10
L 🔗 lbhc-12345	Active	1	•	StandBy	10

図 9.3-1 障害発生前のリソースの状態

この状態でノード A の NIC を無効に設定し、他ノードとの通信を切断します。

以下のコマンドで一時的に NIC を無効に設定します。

ifconfig eth1 down
ifconfig eth0 down

コマンド実行後、SSH 接続は切断されるので、ノード A の再起動は手動で Azure ポータル上から行う必要があります。

再起動後に NIC は自動的に接続されます。

NIC 無効後はノード B の GUI は以下のように、表示されます。

Hierarchies						
🔀 Not Active		Clstr-nodeB			Clstr-nodeA	
የ− 🔀 pgsql-5432		StandBy	10	?	Unknown	1
- 🔀 /data	•	StandBy	10	?	Unknown	1
L 🔀 datarep-data	S	Target	10	?	Unknown	1
— 🔀 ip-10.5.1.100	•	StandBy	10	?	Unknown	1
L 🚺 lbhc-12345		StandBy	10	?	Unknown	1

図 9.3-2 NIC 無効後のリソースの状態

一定時間後、フェイルオーバーが開始され、ノード B で自動的にリソースが起動されます。

LK for Linux QWK Storage mode on Azure Step by step Guide



図 9.3-3 フェイルオーバーが発生したリソースの状態

上記でフェイルオーバーのテストは終了です。

Quorum / Witness Kit で想定される動作につきましては以下のドキュメントも参照ください。

<u>Quorum Witness Server Support Package for Linux 利用ガイド</u>

9.4 留意事項

Quorum / Witness サーバで障害が発生した場合、コミュニケーションパスの全断が起き るとすべてのサーバでリソースが停止します。

そのため、Witness サーバに障害が発生した場合は直ちに正常の状態へ戻すためのアク ションを行う必要があります。

10. お問い合わせ

本書の記載内容についてのお問い合わせ先

■ LifeKeeper 製品の導入を検討中のお客様

弊社パートナー営業部までお問い合わせください。

お問い合わせメールフォーム

https://mk.sios.jp/BC_Web_Free-entry_Inquiry.html

■ LifeKeeper 製品をご購入済みのお客様

お問い合わせの一次窓口が弊社ではない場合があります。 お問い合わせの際はサポート証書よりサポート窓口をご確認ください。

サポート窓口が弊社の場合は、下記の Web サイトよりお問い合わせください。

弊社のサポート窓口

https://bccs.sios.jp/contact/

11. 免責事項

- 本書に記載された情報は予告なしに変更、削除される場合があります。最新のものを ご確認ください。
- 本書に記載された情報は、全て慎重に作成され、記載されていますが、本書をもって、その妥当性や正確性についていかなる種類の保証もするものではありません。
- 本書に含まれた誤りに起因して、本書の利用者に生じた損害については、サイオステクノロジー株式会社は一切の責任を負うものではありません。
- 第三者による本書の記載事項の変更、削除、ホームページ及び本書等に対する不正な アクセス、その他第三者の行ためにより本書の利用者に生じた一切の損害について、 サイオステクノロジー株式会社は一切の責任を負うものではありません。
- システム障害などの原因によりメールフォームからのお問い合せが届かず、または延着する場合がありますので、あらかじめご了承ください。お問い合せの不着及び延着に関し、サイオステクノロジー株式会社は一切の責任を負うものではありません。

【著作権】

本書に記載されているコンテンツ(情報・資料・画像等種類を問わず)に関する知的財 産権は、サイオステクノロジー株式会社に帰属します。その全部、一部を問わず、サイオ ステクノロジー株式会社の許可なく本書を複製、転用、転載、公衆への送信、販売、翻案 その他の二次利用をすることはいずれも禁止されます。またコンテンツの改変、削除につ いても一切認められません。

本書では、製品名、ロゴなど、他社が保有する商標もしくは登録商標を使用しています。

サイオステクノロジー株式会社 住所 : 〒106-0047 東京都港区南麻布 2 丁目 12-3 サイオスビル URL : https://sios.jp