Magic xpi

In Memory

Middleware

目次

このドキュメントでは、IMM クラスタこのドキュメントでは、IMM クラスタ、Magic xpi、および IMM セットアップのインストールと設定について説明します。このドキュメントのすべての手順を正常に完了すると、次のようになります。:

- IMM クラスタの作成
- Magic xpi のインストール
- クラスタでの IMM の構成
- Magic xpi プロジェクトのデバッグとデプロイの準備
- xpi モニタへのアクセス

このドキュメントでは、シングルノード構成とマルチノード構成の両方で IMM クラスタを設定する方法について説明します。要件に応じて導入タイプを選択し、このドキュメントの適切なセクションで設定手順に進むことができます。

高可用性の概要

高可用性 (HA とも呼ばれる) は、ダウンタイムを最小限に抑え、アプリケーションの継続的な可用性を保証するシステムを提供し、インフラストラクチャ内のユニットの 1 つが故障したことによるシステムの完全な障害を回避します。

IMM の HA は、複数のノードを持つ Kubernetes クラスタを作成することで実現されます。また、フォールト トレランスや負荷分散などの他の機能をクラスタに追加することで、高可用性システムを実現できます。

異なる電源とより優れたネットワーク ハードウェアを使用することで、HA セットアップをさらに安全にする他の方法もあります。

オンプレミス IMM クラスタ

このアプローチでは、Magic xpi は Windows マシン上にインストールされ、IMM クラスタは Ubuntu サーバ上で実行されます。

IMM クラスタをセットアップするには、まず Ubuntu サーバを作成する必要があります。高可用性クラスタを選択する場合は、少なくとも 3 台の Ubuntu サーバが必要です。

ツールとソフトウェアの高レベル要件

1. docker イメージやその他の成果物をダウンロードするためのすべてのマシンでのインターネット接続。
2. Magic xpi 4.14 以降の Windows インストーラ。
3. IMM インストール用の Ubuntu サーバ 24.04 LTS を搭載した Linux マシン。既存の Ubuntu サーバーがある場合は、マシンから MicroK8s、Helm、kubectl をアンインストールします。
4. Linux マシン上の IMM クラスタに接続するための Windows マシン上の PuTTY または同等の SSH クライアント (オプション)。
5. Windows マシン上の Chocolatey (choco) または同等のパッケージ マネージャ。
6. Windows マシン上の Kubernetes CLI (kubectl) および Helm のインストール。

シングルノード クラスタまたはマルチノード クラスタのいずれかをインストールできます。マルチノード クラスタを選択した場合は、ダウンタイムがゼロになる高可用性サポートが得られます。

選択内容に応じて、以下の適切なセクションを参照してください。

ハードウェア要件

Magic xpi R&D チームのテストは、次のハードウェア構成で実行されます。これらは最小要件であり、必要なスループットやその他のパフォーマンスの考慮事項に基づいて、要件に応じて変更できます。

開発環境

Microsoft Windows Server 2022

- CPU: 最小 2 Core
- RAM: 最小 8 GB
- HDD: 最小 60 GB

Ubuntu Sever

- OS 24.04LTS
- RAM: 最小 8 GB

- 動的に割り当てられた CPU とメモリで構成された Ubuntu VM は、Microk8s Kubernetes クラスタと互換性がありません。
- Ubuntu システムをセットアップする際、ホスト名は小文字で設定する必要があります。

Magic xpi をインストールするための前提条件

Magic xpi をインストールする前に、次の前提条件を確認する必要があります:

- xpi サーバは Windows サービスをインストールするため、xpi インストールが存在するユーザプロファイルをインストールします。サーバーには、サービスを実行するための適切なアクセス許可が必要です。
- すべてのアプリケーションサーバーには、Magic xpi Admin ユーザーが共有プロジェクトフォルダにアクセスするためのネットワークアクセスと読み取り/書き込み権限が必要です。

IMM をインストールするための前提条件

- IMM は Kubernetes クラスタとして展開されるため、Kubernetes を備えた Linux ホストが必要です。これは、Windows で WSL を使用してインストールされた VM または Ubuntu アプリによって提供される場合があります。
- IMM 用の K8S クラスタがあるホスト マシンは、インターネットにアクセスできる必要があります。IMM 用の K8S クラスタをセットアップするために必要な docker イメージやその他の成果物をプルするには、インターネット アクセスが必要です。

Magic xpi サーバと Kubernetes クラスタ間の適切な通信のためには、すべてのホストのクロックを 15 秒以内のオフセット許容値で同期する必要があります。
詳細については、Magic xpi モニタのタイムゾーン設定を参照してください。

セットアップ ウィザードの手順

各アプリケーション サーバで、インストール メディアからセットアップ プログラム setup.exe を実行します。この実行ファイルはウィザードを開き、インストール プロセスをガイドします。詳細については、Magic xpi Installation Guide.pdf を参照してください。

セットアップ ウィザード後の設定手順

1. すべてのアプリケーション サーバで、**コントロール パネルの サービス** に移動します。
2. サービス ユーザを、プロジェクトに必要なすべてのリソースにアクセスできる十分な権限を持つユーザに変更します。

たとえば、プロジェクトがあるネットワーク ドライブの読み取りと書き込みの権限が必要です。

サーバライセンスのインストール

Magic xpi 4.14.1 を実行するにはライセンスが必要です。インストールするには、以下の手順でインストールします。:

1. 購入時に受け取ったライセンス ファイルをコピーします。
2. すべてのサーバまたは共有の場所にライセンスを貼り付けます。
3. すべてのサーバで、共有ライセンス ファイルを指すように [MAGIC_ENV] LicenseFile エントリを更新して、ライセンス ファイルの場所を含む Magic.ini ファイルを変更します。
   例：
   ```
   LicenseFile =\\10.1.1.6\licenses\License.dat
   ```
4. プロジェクトの ifs.INI ファイルがプロダクション ライセンスで構成されていることを確認します。
   つまり、
   [MAGIC_ENV]LicenseName = IBPRSRVI (Windows マシンの場合)

> ホスト ロック ライセンスの最初の使用は、ライセンスが登録されているホストから行う必要があります。

IMM をデプロイするための前提条件

IMM ミドルウェアにアクセスし、デプロイし、管理するには、次のツールが必要です:

- IMM インストーラーを実行する Windows ホスト:
  - Kubernetes CLI (kubectl)
  - Helm

- PuTTY あるいは同等の SSH および Telnet クライアント ：TeraTerm が便利です。
- Chocolatey (choco) または同等のパッケージマネージャ

- Kubernetes が稼働している Linux サーバ

Windows から IMM をデプロイするためのツールのインストール

主な前提条件である Kubernetes CLI と helm は、さまざまな方法でインストールできます。このドキュメントでは、Chocolatey パッケージ マネージャを使用してこれらのツールをインストールする方法について説明します。Magic xpi インストーラには、セットアップ用の inst_k8s_tools.BAT バッチ ファイルも同梱されています。バッチ ファイルは、¥InMemoryMiddleware¥deploy フォルダにあります。

まず、Chocolatey をインストールします。

Chocolatey のインストール

Chocolatey は管理者としてインストールする必要があります。

1. Windows マシンで PowerShell を管理者として開きます。
2. 次のコマンドを実行して Chocolatey をインストールします

```
Set-ExecutionPolicy Bypass -Scope Process -Force;
[System.Net.ServicePointManager]::SecurityProtocol =
[System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex
((New-Object
System.Net.WebClient).DownloadString('https://community.chocolatey.o
rg/install.ps1'))
```

Kubernetes CLI と Helm のインストール

Chocolatey が正常にインストールされたら、Kubernetes CLI と Helm をインストールできます。

管理者としてコマンド プロンプトを開きます(PowerShell)。

以下のコマンドを実行します:

```
choco install kubernetes-cli
choco install kubernetes-helm
```

ツールをインストールする際には、マシンの再起動が必要になる場合があります。

サーバから kubeconfig ファイルを取得する

Kubernetes CLI (kubectl) は、通常ユーザー プロファイル フォルダに配置されている構成ファイルで定義されているサーバと対話します。SSH/Telnet ツールを使用して、Kubernetes を実行しているサーバからこの構成ファイルを取得してみましょう。このファイルを設定するには、：

1. Putty または同等のツールを開きます。
2. IMM セットアップのためにホスト マシンに上記ツールで接続します。
3. 接続したら、コマンドを実行します：
   ```
   microk8s config > config
   ```
4. 出力ファイル「config」を Windows ホストの %UserProfile%/.kube フォルダーにコピーします。このフォルダが存在しない場合は、コマンド プロンプトから次の 2 つのコマンドを実行して作成します。
   ```
   cd /D %UserProfile%
   mkdir .kube
   ```
5. 次のコマンドを使用して構成をテストします:
   ```
   kubectl cluster-info
   ```

セキュリティ、ファイアウォール、プロキシ サーバに関する考慮事項

特定のウイルス対策ソフトウェアは、Microk8s および IMM の適切なインストールを妨げる可能性があります。インストールが完了しない場合は、それらを無効にして再インストールする必要がある場合があります。問題を引き起こす可能性のあるソフトウェアを以下に示します。このリストは網羅的なものではないことに注意してください。

- Sentinel　https://www.sentinelone.com/surfaces/endpoint/
- Microsoft Defender

システムがファイアウォールの背後に配置されている場合は、特定のポートを開く必要があります。このドキュメントでは、Windows、Ubuntu、および RHL 上の IMM 用に開く必要があるポートのリストを示します。

外部ポート（「E」で表示）はインターネット向けポートであり、内部ポート（「I」で表示）は社内向けファイアウォール ポートです。これら 2 種類のポートは、外部ネットワークと内部ネットワークからの許可された接続のみを許可します。

ほとんどの企業では、ファイアウォール ポートの開放に関して厳格なセキュリティ ガイドラインを設けています。セキュリティ プロトコルに従う手段としてファイアウォール ポートを開放する前に、IT インフラストラクチャ チームに必ず通知してください。
サービスを使用しなくなった場合は、必ずポートを閉じてください。

ポート#	目的	コメント	I=内部, E=外部
80	http	xpi サーバが機能するために必要。	E
443	https	デプロイメントが保護(暗号化)されている場合、xpi サーバが機能するために必要。	E
4789/udp	calico	calico VXLAN が有効になっている Calico ネットワーク。	I
5353/udp	VLAN DNS	ノード間でサービスを解決するために必要な VLAN DNS サービス。	I
5000 5119	これらのポートは、IMM_DB と Sentinel Pod を健全な状態に保ち、後者の両方が稼働し続けるために必要です。これらのポートは、イメージをプルしたりサービスを開始したりするのに役立ちます。		I
6379	IMM-DB	xpi サーバが機能するために必要。	I
10250	kubelet	匿名認証は無効です。X509 クライアント証明書が必要です。	I
10255	kubelet	Kubelet の読み取り専用ポートです。	I
10257 10259	kube-controller kubescheduler	認証と認可を伴う HTTPS を提供します。	I
12379	etcd	SSL 暗号化。接続するにはクライアント証明書が必要です。	I
16443	API Server	SSL 暗号化。クライアントは静的パスワード ファイルから有効なパスワードを提示する必要があります。	I
19001	dqlite	SSL 暗号化。接続するにはクライアント証明書が必要です。	I
25000	cluster-agent	アクションを承認するには適切なトークンが必要です。	I

27017	LogDB	xpi サーバーが機能するために必要です。	I

強調表示されたポートは、Kubernetes クラスタのインストールに必要です。ポートの詳細については、Microk8s website を参照してください。

Ubuntu でポートを開ける例

例えば 10.9.3.77, 10.9.3.191, 10.9.3.245 の 3 台のマシンがある場合、10.9.3.77, 10.9.3.191, 10.9.3.245 でポート 25000 を有効にするには以下のコマンドを使用します。:

```
sudo ufw allow from <destination IP> to any port <port>
sudo ufw allow from <10.9.3.191> to any port <25000>
sudo ufw allow from <10.9.3.245> to any port <25000>
```

ブロックされた Web サイトのファイアウォール設定に関連するその他の問題については、トラブルシューティング セクションの質問 4 を参照してください。

プロキシ サーバを使用してインターネットに接続している場合は、プロキシ サーバの詳細を追加する必要があります。詳細については以下の How-To セクションに記載されている情報を参照してください。

デプロイ前の構成

デプロイする前に、values.yaml ファイルにいくつかの構成パラメータを設定する必要があります。

パラメータ	目的
LOG_HISTORY_THRESHOLD_IN_DAYS	このパラメータはアクティビティ ログに関連しています。アクティビティをログに記録する必要がある期間を指定します。 1 日以内に MongoDB 内に生成されたすべてのデータをクリアする場合は、このパラメータを 1 に設定します。このしきい値に達すると、IMM の準備ができ、MongoDB が実行中であれば、1 日分のデータがクリアされます。 1 日に複数回 MongoDB データをクリアできますが、1 時間ごとにデータをクリアする場合は、0.04 などの小数値を設定する必要があります。 デフォルト値は 7 日です。
REQ_HISTORY_THRESHOLD_IN_DAYS	このパラメータは、MongoDB から個々のプロジェクトのフロー リクエスト履歴データを削除するタイミング(日数)を指定します。削除期間のデフォルト値は 3 日です。
 ・上記の 2 つのパラメータのいずれか一方のみが設定されている場合は、もう一方のパラメータに独自のデフォルト値が割り当てられます。 ・Log-DB のディスク プレッシャーが重大な場合 (5 GB に制限)、IMM はアクティビティ ログを自動的にクリアします。これは、プロジェクト アクティビティ ログの一部が失われるという代償を払って、差し迫った障害からシステムを自己保護するためです。 長期ログ保持のための Open-telemetry 統合機能を参照ください。	
WIRE_TIGER_CACHE_GB	このパラメータは、MongoDB のキャッシュ ストレージの最大量 (GB 単位) を設定するために使用されます。

MOVE_FRH_TO_LOGDB_SCHEDULER_INTERVAL_MILI	フロー リクエスト履歴は、ミリ秒単位で設定された間隔ごとに LogDB Scheduler に転送されます。 設定された間隔値ごとに、Redis 内の「完了」ステータスにあるフロー リクエスト履歴エントリが、FRH_<プロジェクト名> という名前で ¥collection フォルダー内の MongoDB に転送されます。このパラメータが指定されていない場合、デフォルト値は 10000 ミリ秒です。
TZ	xpi モニターのタイムゾーンは、¥InMemoryMiddleware¥deploy フォルダーにある values.yaml ファイルの immMonitor デプロイメント セクションの下にある TZ 環境変数を使用して定義できます。このタイムゾーンは、モニターのすべてのタブで定義されたタイムゾーンに基づいて xpi モニターによって時間を表示するために使用されます。デフォルトのタイムゾーンは US/Central です。必要に応じて変更できます。 日本時間に設定する場合は timeZone: &timeZone "Asia/Tokyo" とします。

values.yaml の内容

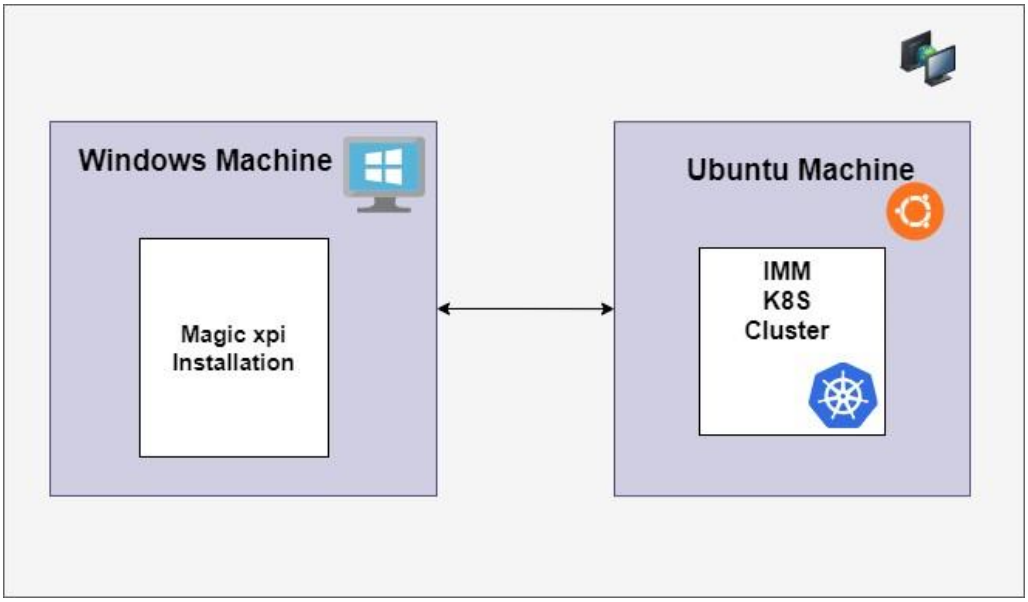
values.yaml ファイルは以下のようになります:

```
timeZone: &timeZone "Asia/Calcutta"
immController:
   deployment:
     container:
      immClusterController:
        image:
devmcsworkspaceacr.azurecr.io/4.14/int/win/patches/images/immcontroller
        tag: 4.14.1.318
        imagePullPolicy: IfNotPresent
        env:
          LOG_LEVEL: "warn"
          REQ_BODY_MAX_SIZE_IN_BYTES : "100000000"
          TZ: *timeZone
immTunnel:
   deployment:
     container:
       immTunnel:
        image:
devmcsworkspaceacr.azurecr.io/4.14/int/win/patches/images/immtunnel
        tag: 4.14.1.318
        imagePullPolicy: IfNotPresent
        env:
          LOG_LEVEL: "warn"
          REQ_BODY_MAX_SIZE_IN_BYTES : "100000000"
          TIMEOUT : "300000"
          HEADER_TIMEOUT : "130000"
          KEEP_ALIVE_TIMEOUT : "120000"
 immMonitor:
   deployment:
     replicas: 1
     container:
       immMonitor:
```

```
        image:
devmcsworkspaceacr.azurecr.io/4.14/int/win/patches/images/immmonitor
        tag: 4.14.1.318
        imagePullPolicy: IfNotPresent
        env:
          LOG_LEVEL: "warn"
          TZ: *timeZone
immServer:
   label:
    app: xpi-imm-server-deployment
   deployment:
     replicas: 1
     container:
      immServer:
        image:
devmcsworkspaceacr.azurecr.io/4.14/int/win/patches/images/immserver
        tag: 4.14.1.318
        imagePullPolicy: IfNotPresent
        env:
          TZ: *timeZone
logdb-chart:
   logdb:
      env:
       WIRE_TIGER_CACHE: 1
logdb-ha-chart:
  logdb:
   env:
       WIRE_TIGER_CACHE: 1
```

シングルノードクラスタの設定

シングルノード クラスターは、Kubernetes のコンポーネントを 1 台のマシンで実行するシンプルな構成です。シングルノード クラスターは、スケーラビリティ、冗長性、フォールト トレランスなどの機能を提供しませんが、セットアップと使用が簡単で、リソース オーバーヘッドが少なく、コストを節約できます。



前提条件

- Linux ディストリビューション (Ubuntu 22.04 LTS) で構成された Ubuntu サーバ。
- すべてのマシンでインターネットにアクセスできること。

Linux マシンが起動したら、以下の手順に従って MicroK8s クラスタをセットアップします。

> ユーザー権限に応じて、sudo を使用してコマンドを実行する必要がある場合があります。

シングルノードクラスタのセットアップ手順

1. ターミナルからコマンドを実行します。

```
sudo snap install microk8s --classic --channel=1.26/stable
```

```
xpiuser@xpiqaubuntuvm23:~$ sudo snap install microk8s --classic --channel=1.26/stable
microk8s (1.26/stable) v1.26.10 from Canonical** installed
```

2. MicroK8s が正常にインストールされたかどうかを確認するには、コマンドを実行します。

```
sudo microk8s status
```

このコマンドにより、Microk8s のステータスが *Running(*実行中*)*として表示されます。

MicroK8s が実行されていない場合は、次のコマンドで起動します:

```
sudo microk8s start
```

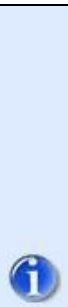
3. MicroK8s をインストールしたら、以下の Microk8s のアドオンをインストールする必要があります 。:

   i. **dns**

   アドオンを有効にするには、次のコマンドを実行します

   ```
   sudo  microk8s  enable  dns
   ```

   ii. **metallb**



metallb アドオンには、サポートされているネットワーク インターフェース (イーサネット名) の IP アドレスが必要です。たとえば、インターフェースのタイプが eth0 の場合、次のいずれかのコマンドを使用して IP アドレスを取得できます。

```
ifconfig eth0
```

OR

```
ip addr show eth0
```

コマンドの実行時にエラーが発生した場合は、まず以下のコマンドを実行してください。

```
sudo apt install net-tools
```

   このアドオンを有効にするには、以下のコマンドを実行します。

   ```
   sudo microk8s enable metallb
   ```

MicroK8s が IP 範囲の入力を要求したら、Ubuntu サーバの IP アドレスを範囲 (ハイフン文字で区切る) として、<IP_address>-<IP_address> の形式で指定します。

たとえば、Ubuntu サーバの IP アドレスが 1.2.2.1 の場合、範囲を次のように指定します。:

```
1.2.2.1-1.2.2.1
```

   iii. **storage**

   このアドオンを有効にするには、以下のコマンドを実行します

   ```
   sudo microk8s enable storage
   ```

これらの 3 つのアドオンは、以下のコマンドで一度に有効にできます。
`sudo microk8s enable dns metallb storage`

これで IMM をインストールする準備が整いました。

4. MicroK8s をインストールしたら、Kubernetes 構成ファイル (kubeconfig) を生成する必要があります。このファイルは、Linux マシンで実行されている IMM クラスタに接続を確立し、コマンドを実行するために、Windows サーバにコピーする必要があります。

   i. config ファイルを生成するには、Ubuntu サーバから以下のコマンドを実行します。

   ```
   sudo microk8s config
   ```

   このコマンドは、端末に config を出力します。

   ii. 画面の内容をコピーして空のファイルに貼り付け、ファイル名を çonfig にします。このファイルを Windows マシンの以下のフォルダにコピーします。

   <Drive>:\Users\<user>\.kube\

   iii. Linux マシンの IP アドレス エントリを Windows マシンの hosts ファイルに追加します。そのためには、C:¥Windows¥System32¥drivers¥etc フォルダに移動し、テキスト エディタで hosts ファイルを開いて、以下の形式で入力します。

   ```
   <Linux Machine IP Address> <Domain name>
   ```

   例えば、 `1.2.1.1 mydomainname`

   ドメイン名は、Magic xpi のインストール時に指定したものと同じであることに注意してください。

RHEL

Snap Daemon がプリインストールされていないディストリビューションを利用する場合は、以下の手順を実施してください。

i. 使用している RedHat のバージョンを確認するには、次のコマンドを実行します:

```
cat /etc/redhat-release
```

ii. EPEL リポジトリがまだディストリビューションに追加されていない場合は、次のコマンドで追加できます:

```
sudo  dnf  install
https://dl.fedoraproject.org/pub/epel/epelrelease-latest-
9.noarch.rpm

sudo dnf upgrade
```

RedHat マシンに Microk8s をインストールする前に、DNF と SNAPD をインストールする必要があることに注意してください。

SNAPD のインストール

1. EPEL リポジトリが RHEL インストールに追加されたら、snapd パッケージをインストールします:

   ```
   sudo yum install snapd
   ```

2. メインの snap 通信ソケットを管理する systemd ユニットを有効にします:

   ```
   sudo systemctl enable --now snapd.socket
   ```

3. Classic Snap サポートを有効にするため、次のコマンドを実行します。このコマンドは、**/var/lib/snapd/snap** と **/snap** の間にシンボリック リンクを作成します:

   ```
   sudo ln -s /var/lib/snapd/snap /snap
   ```

4. ログアウトして再度ログインするか、システムを再起動して、Snap のパスが正しく更新されていることを確認します。

IMM with SSL を使用した IMM のデプロイ

証明書が Letsencrypt.org から生成された場合、そのままでは機能しません。Troubleshooting セクションを参照ください。

SSL を使用して IMM をデプロイするには、
<xpi_installation_folder>¥InMemoryMiddleware¥config フォルダに 3 つの証明書ファイルが必要です。:

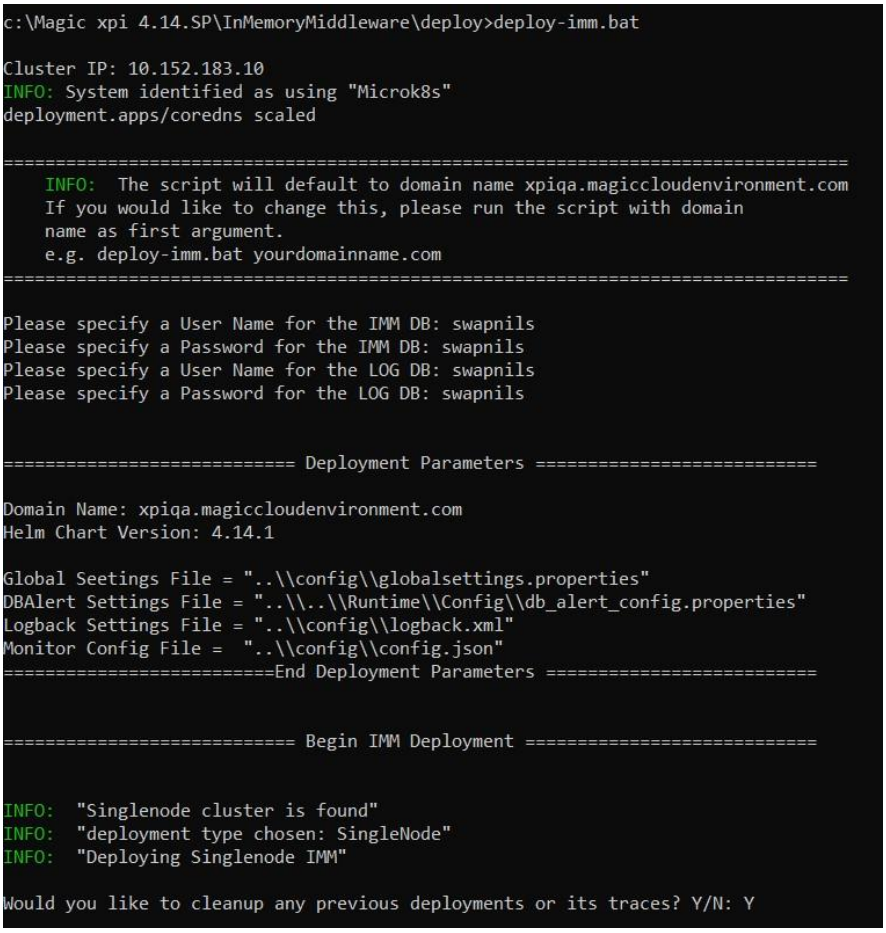
i. imm.PEM

ii. imm.KEY

iii. imm.CRT

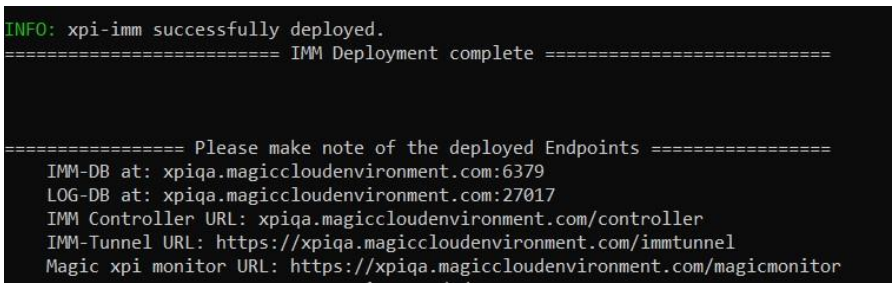
1. 証明書ファイルの存在を確認したら、deploy-imm.bat を使用して、IMM をデプロイします。このファイルは以下のフォルダに存在します。
   "<Magic_xpi_installation_folder>\InMemoryMiddleware\deploy".

2. IMM DB (Redis) および LOG DB (MongoDB) セキュリティの一部として、認証用のユーザー名とパスワードを入力します。



3. 「Singlenode cluster is found(シングルノード クラスターが見つかりました)」というメッセージは、IMM が高可用性環境にデプロイされていることを示しています。「Deploying IMM with HTTPS(HTTPS を使用して IMM を導入しています)」というメッセージは、デプロイが SSL を使用して行われていることを示しています。

   導入が完了すると、次のメッセージが表示されます。:

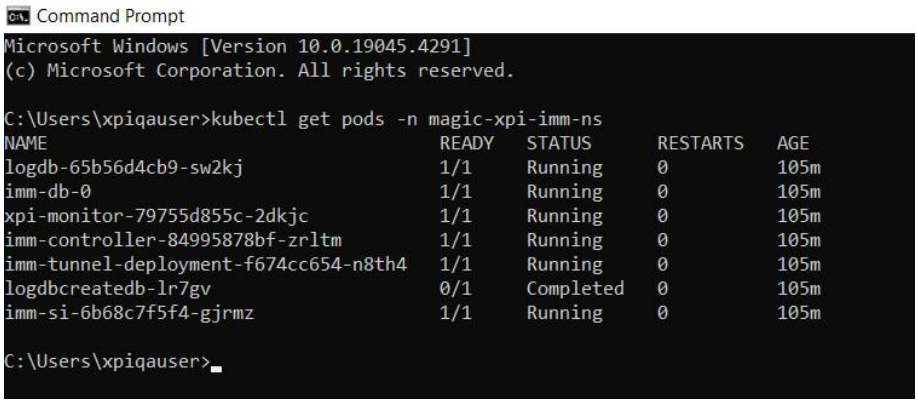


4. すべてのポッドが稼働中かどうかを確認するには、次のコマンドを使用します:

   ```
   kubectl get pods -n <namespace>
   ```

   たとえば、ネームスペースが magic-xpi-imm-ns の場合、コマンドは次のようになります:

```
kubectl get pods -n magic-xpi-imm-ns
```



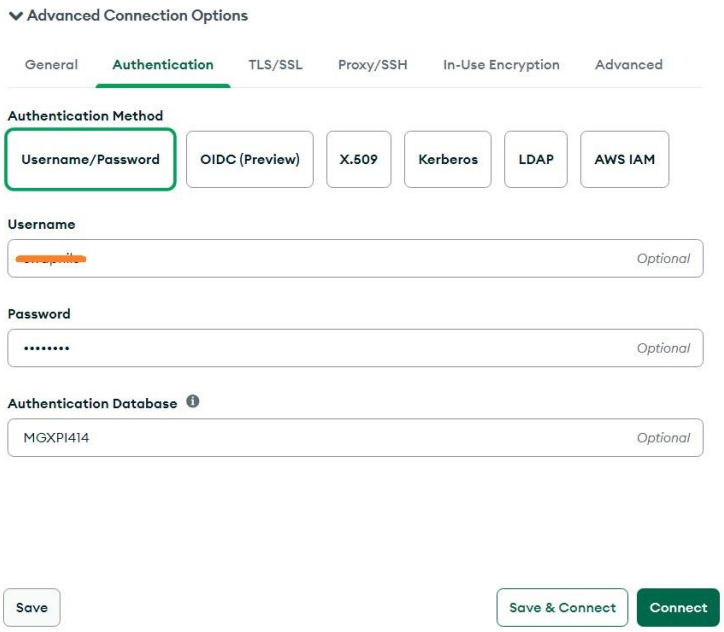
SSL デプロイでは、エージェントを保護するために CA 証明書を使用する必要があります。これらの証明書はどこにでも配置できます。証明書ファイルのパスを、以下のフォルダ内の.ENV ファイル内に TLS_CA_FILE_PATH として設定します。
"<Magic_xpi_installation_folder>\InMemoryMiddleware\agent".

たとえば、

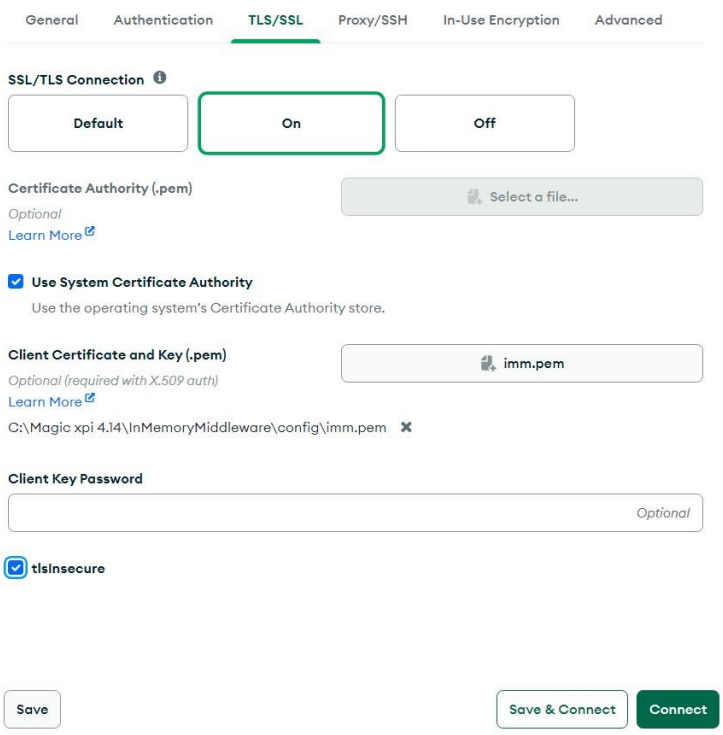
```
TLS_CA_FILE_PATH ="c:\temp\cacerts\gd_bundle-g2-g1.crt"
```

5. 次に、以下のフォルダ内にある、imm-agent.exe を使用して IMM エージェントを起動します。
   "<Magic_xpi_installation_folder>\InMemoryMiddleware\agent"
   「Handshake Success: OK」というメッセージが表示されます。これは、エージェントが IMM に正常に接続したことを意味します。
6. MongoDB Compass などの外部ツールを開きます。

   このように外部から Redis データにアクセスする場合は、資格情報を提供する必要があることに注意してください。
7. 資格情報を入力します。



8. CA 証明書ファイルを選択します。



証明書ファイル imm.PEM が選択されていることに注意してください。また、MongoDB が認証を使用してアクセスされていることも確認してください。

SSL なしで IMM をデプロイする

CA 証明書が存在しないか無効である場合、デプロイプロセス自体は SSL なしでデプロイされます。\config フォルダに証明書を追加する必要はありません。さらにエージェント用の以下のフォルダにある.ENV ファイルに TLS_CA_FILE_PATH="<Absolute Path¥CA_certificate_filename>”を設定する必要はありません。

<Magic_xpa_installation_folder>\InMemoryMiddleware\agent.

Magic xpi のインストール

1. Magic xpi 4.14.1 を Windows マシンにインストールします。
2. SSL デプロイメントを使用するには、使用している JAVA cacerts ファイルに CA 証明書をインポートする必要があります。

   たとえば、次のコマンドを使用して証明書をインポートできます:

```
keytool -importcert -file "c:\Magic xpi
4.14.1\InMemoryMiddleware\ config\imm.crt" -alias imm -keystore
"c:\Magic xpi 4.14.1\Runtime\JDK8x64\jre\lib\security\cacerts"
```

ここで

- "c:\Magic xpi 4.14.1\InMemoryMiddleware\config\imm.crt" は証明書ファイルのパスです。
- "c:\Magic xpi 4.14.1\Runtime\JDK8x64\jre\lib\security\cacerts" は、JAVA cacerts ファイルのパスです。

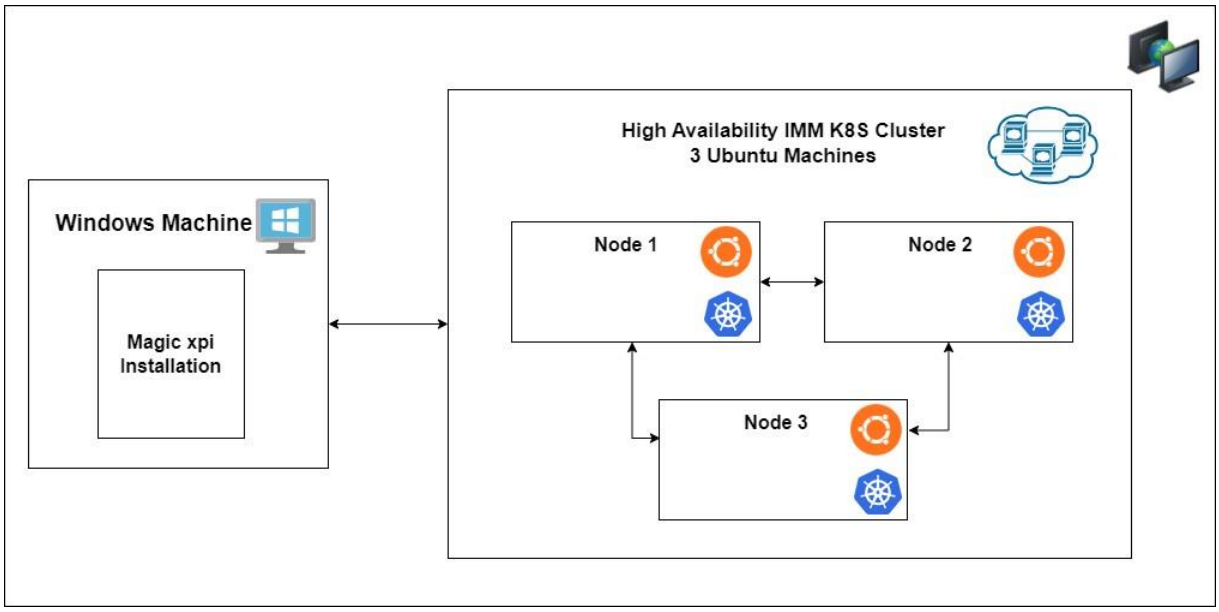
デバッガー サービスを使用するには、32 ビット JAVA でも証明書をインポートする必要があります。

コマンドの例

```
keytool -importcert -file "c:\Magic xpi
4.14.1\InMemoryMiddleware\config\imm.crt" -alias imm -keystore
"c:\Magic xpi 4.14.1\Runtime\JDK8x86\jre\lib\security\cacerts"
```

マルチノード クラスターのセットアップ

マルチノード Kubernetes クラスタは、Kubernetes コンポーネントを複数のマシンに分散させることを含み、スケーラビリティ、高可用性、フォールトトレランスなどの利点を提供します。セットアップとデプロイには複雑さが伴い、これらのクラスタには追加のリソースとコストのオーバーヘッドがあります。



前提条件

1. クラスタには、Kubernetes クラスタのノードとして機能する Ubuntu サーバが少なくとも 3 台必要です。
2. すべてのマシンでインターネットにアクセスできる必要があります。
3. 3 台のマシンの IP アドレスとは異なるもう一つの IP アドレスが必要です。この IP アドレスにより、Windows マシン上の xpi インストールがこの高可用性クラスタに接続できるようになります。この IP アドレスは、ロード バランサがユーザからのリクエストを処理し、それを Ingress コントローラに転送するために使用されます。この追加の IP は、ロード バランサに割り当てることができるように、ネットワーク内で未使用の IP である必要があります。

Linux マシンが起動したら、以下の手順に従って MicroK8s クラスタをセットアップします。

ユーザ権限に応じて、sudo を使用してコマンドを実行する必要がある場合があります。

1. ターミナルから以下のコマンドを実行します。

```
sudo snap install microk8s --classic --channel=1.26/stable
```

```
xpiuser@xpiqaubuntuvm23:~$ sudo snap install microk8s --classic --channel=1.26/stable
microk8s (1.26/stable) v1.26.10 from Canonical** installed
```

2. MicroK8s が正常にインストールされたことを確認するため、以下のコマンドを実行します。

```
sudo microk8s status
```

このコマンドは Running(実行中)というステータスを返します。

マルチノードクラスタのセットアップ手順

1. これらの各 Linux マシンのホスト名と IP アドレスを明らかにします。

ホスト名は **hostname** コマンドで調べることができます。

サポートされているネットワーク インターフェイスの IP アドレス (イーサネット名) が必要になります。たとえば、インターフェイスのタイプが *eth0* の場合、IP アドレスは次のコマンドのいずれかを使用して取得できます。:

```
ifconfig eth0
```

OR

```
ip addr show eth0
```

コマンドの実行時にエラーが発生した場合は、まず次のコマンドを実行してください。:

```
sudo apt install net-tools
```

たとえばマシンの IP アドレスとホスト名は次のようになります。:

- 10.1.2.1 xpiubuntuvm1 (ノード 1)
- 10.1.2.2 xpiubuntuvm2 (ノード 2)
- 10.1.2.3 xpiubuntuvm3 (ノード 3)

設定時に必要となるため、すべてのノードの IP アドレスとホスト名をメモしておいてください。

2. 上記のすべてのノードのエントリを**/etc/hosts** ファイルに追加し、各ノードに適用します。

   たとえばノード 1 のホストファイルには以下のような追加行が含まれます。：

```
10.1.2.2 xpidevubuntuvm2
10.1.2.3 xpidevubuntuvm3
```

ユーザがホストファイルにエントリを追加するための必要な権限を持っていることを確認してください。

3. 次に、ノード 1 から次のコマンドを実行します。このコマンドはすべてのノードを結合して高可用性クラスタにするのに役立ちます。

```
sudo microk8s add-node
```

   他の 2 つのノードを現在のノードに結合するための結合コマンドが以下のように生成されます。

   たとえば以下のようなコマンドが出力されます:

```
microk8s join
10.1.2.1:25000/9c1b7596dbda8e9de6eea4179bb70c4c/e1c1ddfee74d
```

4. このコマンドをコピーします。
5. クラスタに参加させたいノードで、上記の手順からコピーしたコマンドを実行します。
6. クラスタ内の残りのノードに参加するには、手順 3 と 4 (上記) を繰り返します。

コマンドがすべてのノードで正常に実行されると、ノードが結合されて高可用性クラスタが形成されます。

上記のコマンドにはトークンが含まれており、有効期限が限られています。有効期限が切れると、コマンドは「接続に失敗しました。無効なトークン (500)」という応答を返します。このような場合は、上記の手順 3 と 4 を繰り返す必要があります。

7. すべてのノードが結合されたら、ノード 1 から指定されたコマンドを実行します。

```
sudo microk8s status
```

このコマンドによって生成された出力に「high-availability: yes」と表示された場合、HA は正常に構成されています。この表示とともに、コマンドは「datastore master nodes」という情報の後に、クラスタに参加しているすべてのマシンの IP アドレスも出力します。

```
xpiuser@xpiqaubuntuvm8:~$ sudo microk8s status
microk8s is running
high-availability: yes
  datastore master nodes: 10.9.3.179:19001 10.9.3.180:19001 10.9.3.181:19001
  datastore standby nodes: none
addons:
  enabled:
    dns                  # (core) CoreDNS
    ha-cluster           # (core) Configure high availability on the current node
    helm                 # (core) Helm - the package manager for Kubernetes
    helm3                # (core) Helm 3 - the package manager for Kubernetes
    metallb              # (core) Loadbalancer for your Kubernetes cluster
  disabled:
    cert-manager         # (core) Cloud native certificate management
    community            # (core) The community addons repository
    dashboard            # (core) The Kubernetes dashboard
    gpu                  # (core) Automatic enablement of Nvidia CUDA
    host-access          # (core) Allow Pods connecting to Host services smoothly
    hostpath-storage     # (core) Storage class; allocates storage from host directory
    ingress              # (core) Ingress controller for external access
    kube-ovn             # (core) An advanced network fabric for Kubernetes
    mayastor             # (core) OpenEBS MayaStor
    metrics-server       # (core) K8s Metrics Server for API access to service metrics
    minio                # (core) MinIO object storage
    observability        # (core) A lightweight observability stack for logs, traces and metrics
    prometheus           # (core) Prometheus operator for monitoring and logging
    rbac                 # (core) Role-Based Access Control for authorisation
    registry             # (core) Private image registry exposed on localhost:32000
    storage              # (core) Alias to hostpath-storage add-on, deprecated
xpiuser@xpiqaubuntuvm8:~$
```

MicroK8s アドオンのインストール

MicroK8s をインストールしたら、以下の MicroK8s アドオンをインストールする必要があります:

> dns および metallb アドオンは、マスターまたはノード 1 にのみ必要です。ストレージ アドオンは、3 台のマシンすべてに必要です。

i. **dns**

このアドオンを有効にするには、次のコマンドを実行します。

```
sudo microk8s enable dns
```

ii. **metallb**

このアドオンを有効にするには、次のコマンドを実行します。

```
sudo microk8s enable metallb
```

IP 範囲を指定するように求められた場合は、IP アドレスを範囲として指定してください。

```
<IP Address>-<IP Address>
```

高可用性モードでは、ネットワークで使用されていない、3 台のマシンの IP アドレスとは異なる追加の IP アドレスを入力します。

たとえば使用可能な IP が 10.9.1.1 の場合、範囲を 10.9.1.1-10.9.1.1 と入力します。

iii. **storage**

このアドオンを有効にするには、次のコマンドを実行します。

```
sudo microk8s enable storage
```

以下のコマンドを使用して、これら 3 つのアドオンを一度に有効にすることができます

```
sudo microk8s enable dns metallb storage
```

Kubernetes への接続

Kubernetes への接続 Kubernetes に接続するには、まず Kubeconfig を生成する必要があります。

1. ノード 1 サーバで kubeconfig ファイルを生成します。kubeconfig を生成するには、以下のコマンドを実行します:

   ```
   sudo microk8s config
   ```

   このコマンドはターミナルに構成を出力します。

2. 画面の内容をコピーして空のファイルに貼り付け、config という名前を付けます。このファイルを Windows マシンの **<Drive>:¥Users¥<user>¥.kube¥** にコピーする必要があります。

3. 上記で入力した (metallb の有効化時に) 3 台のマシンの IP アドレスとは異なる IP アドレスのエントリを Windows マシンの hosts ファイルに追加します。このためには、**C:¥Windows¥System32¥drivers¥etc** フォルダに移動し、テキスト エディタで hosts ファイルを開いて、次の形式でエントリを追加します:

   <追加の IP アドレス> <ドメイン名>

   例:

1.2.1.1 mydomainname

ドメイン名は、Magic xpi のインストール時に指定したものと同じであることに注意してください。

SSL を使用した IMM のデプロイ

1. SSL を使用してデプロイする場合、
   <xpi_installation_folder>¥InMemoryMiddleware¥config フォルダに次の 3 つの証明書ファイルが必要です:

   i. imm.PEM
   ii. imm.KEY
   iii. imm.CRT

9. ファイルの存在を確認したら、次のコマンドで IMM をデプロイします:

   `deploy-imm.bat`

   このファイルは以下のフォルダに格納されています。
   "<Magic_xpi_installation_folder>\InMemoryMiddleware\deploy"

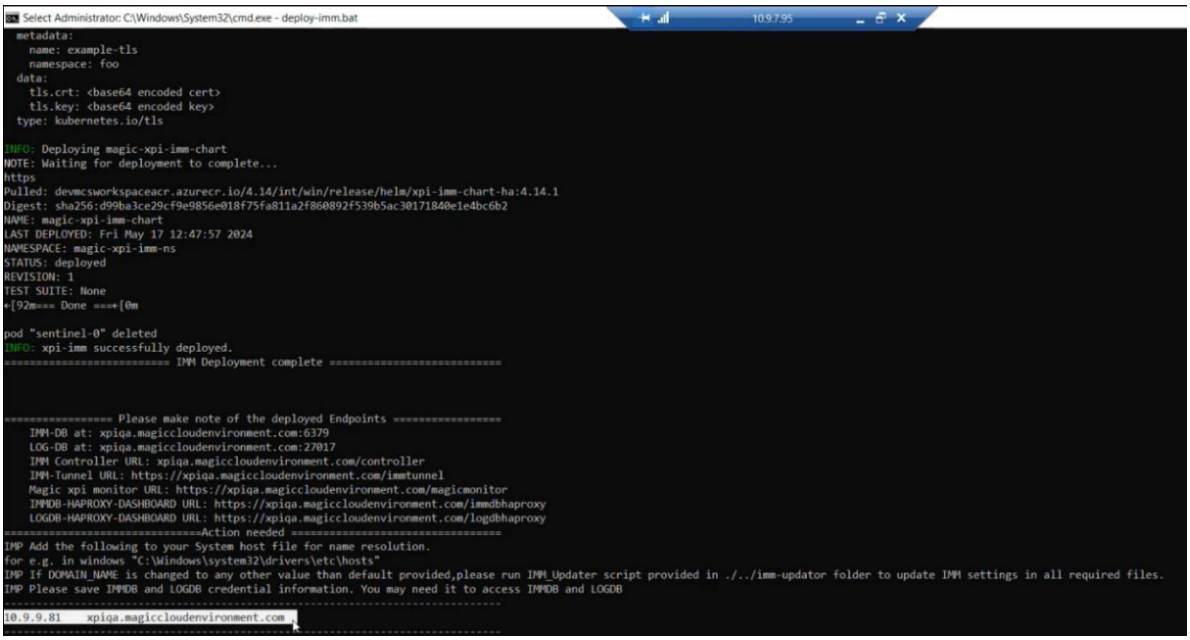
10. IMM DB (Redis) および LOG DB (MongoDB) セキュリティの一環として認証のためのユーザ名とパスワードを入力します。
11. デプロイメント タイプ (シングルノード、マルチノード) を選択するように求められます。デフォルトのデプロイメントはマルチノードです。
12. マルチノード クラスタを続行するには、**y** と入力して Enter キーを押します。

```
c:\Magic xpi 4.14.SP\InMemoryMiddleware\deploy>deploy-imm.bat

Cluster IP: 10.152.183.10
INFO: System identified as using "Microk8s"
deployment.apps/coredns scaled

================================================================================
    INFO:  The script will default to domain name xpiqa.magiccloudenvironment.com
    If you would like to change this, please run the script with domain
    name as first argument.
    e.g. deploy-imm.bat yourdomainname.com
================================================================================

Please specify a User Name for the IMM DB: swapnils
Please specify a Password for the IMM DB: swapnils
Please specify a User Name for the LOG DB: swapnils
Please specify a Password for the LOG DB: swapnils


========================== Deployment Parameters ==========================

Domain Name: xpiqa.magiccloudenvironment.com
Helm Chart Version: 4.14.1

Global Seetings File = "..\\config\\globalsettings.properties"
DBAlert Settings File = "..\\..\\Runtime\\Config\\db_alert_config.properties"
Logback Settings File = "..\\config\\logback.xml"
Monitor Config File =  "..\\config\\config.json"
==========================End Deployment Parameters ==========================


========================== Begin IMM Deployment ============================


INFO:  "Multinode cluster is found"
Multinode cluster in identified, by default multinode IMM will be deployed, press 'y' key to continue.If you wish to deploy singlenode IMM then press 's': y
INFO:  "deployment type chosen: MultiNode."
INFO:  "Deploying Multinode IMM"
```

13. 「**Multinode cluster is found(マルチノード クラスターが見つかりました)**」というメッセージは、IMM が高可用性環境に導入されていることを示しています。「**Deploying IMM with**

HTTPS(HTTPS を使用して IMM を導入しています)」というメッセージは、デプロイが SSL を使用して行われていることを示しています。

デプロイが完了すると、次のメッセージが表示されます。:



14. 稼働中のすべてのポッドを確認するには、以下のコマンドを実行します:

```
kubectl get pods -n <namespace>.
```

たとえば、ネームスペースが magic-xpi-imm-ns の場合、コマンドは以下のようになります:

```
kubectl get pods -n magic-xpi-imm-ns
```

Administrator: C:\Windows\System32\cmd.exe

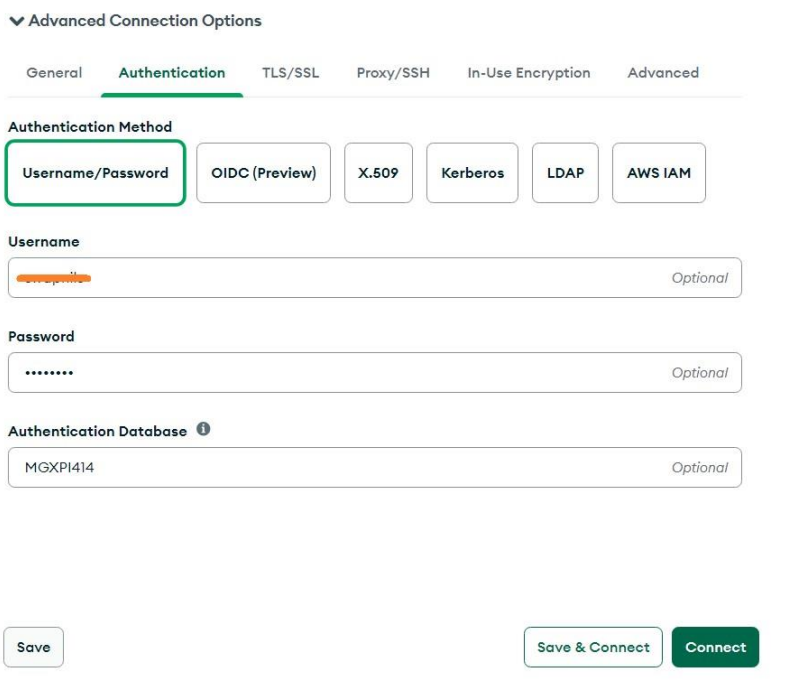
```
c:\Magic xpi 4.14\InMemoryMiddleware\deploy>kubectl get pods -n magic-xpi-imm-ns
NAME                                    READY   STATUS      RESTARTS   AGE
haproxy-deployment-5ff8b9f664-lkl8v     1/1     Running     0          101m
logdb-haproxy-deployment-686d84979f-2xk9r   1/1     Running     0          101m
logdb-haproxy-deployment-686d84979f-2bxvh   1/1     Running     0          101m
haproxy-deployment-5ff8b9f664-fmpk2     1/1     Running     0          101m
haproxy-deployment-5ff8b9f664-vp8nm     1/1     Running     0          101m
logdb-haproxy-deployment-686d84979f-ffb99   1/1     Running     0          101m
logdb-0                                 1/1     Running     0          101m
imm-db-0                                1/1     Running     0          101m
imm-db-1                                1/1     Running     0          100m
logdb-1                                 1/1     Running     0          100m
imm-db-2                                1/1     Running     0          100m
logdbcreatedb-r8m7t                     0/1     Completed   0          101m
logdb-2                                 1/1     Running     0          100m
imm-tunnel-8cf54674-x4pqk               1/1     Running     0          101m
sentinel-1                              1/1     Running     0          100m
sentinel-2                              1/1     Running     0          99m
imm-tunnel-8cf54674-xmns9               1/1     Running     0          101m
xpi-monitor-977cb679d-m75w9             1/1     Running     0          101m
imm-si-76d8d77fb5-jc86p                 1/1     Running     0          101m
imm-controller-78d6c59c76-wfn6f         1/1     Running     0          101m
imm-tunnel-8cf54674-dlpxk               1/1     Running     0          101m
sentinel-0                              1/1     Running     0          97m
```

SSL を使用したデプロイにおいて、エージェントを保護したい場合は、CA 証明書を使用する必要があります。これらの証明書はどこにでも配置できます。証明書ファイルのパスを「**<Magic_xpi_installation_folder>InMemoryMiddleware¥agent**」にある**.env** ファイルの **TLS_CA_FILE_PATH** というフラグに割り当てる必要があります。

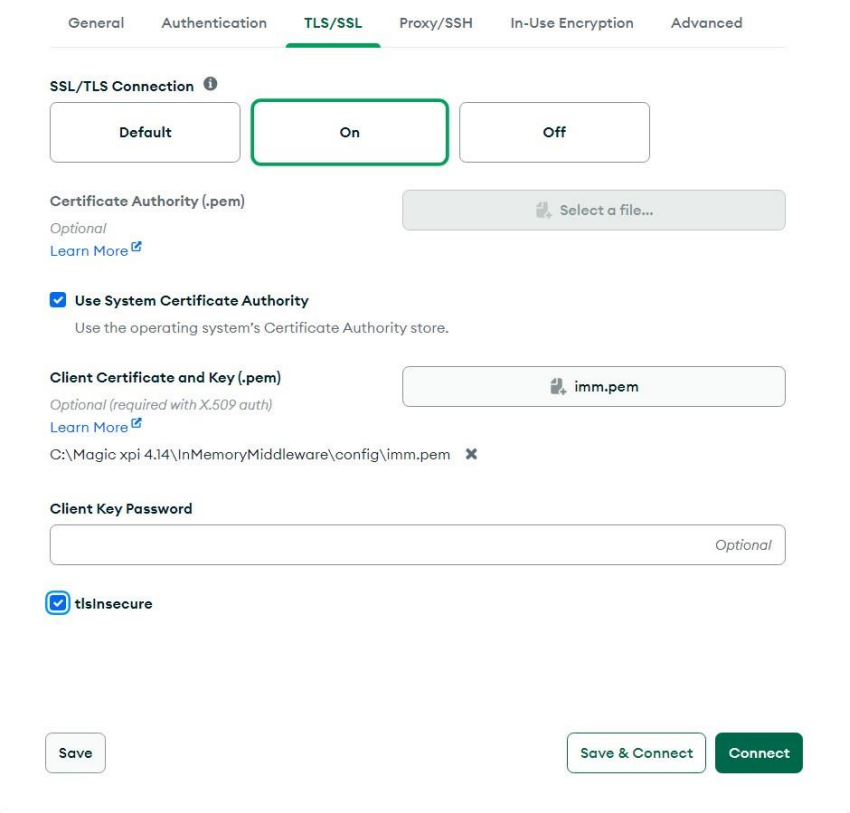
例:

```
TLS_CA_FILE_PATH ="c:\temp\cacerts\gd_bundle-g2-g1.crt"
```

15. 次に、「**<Magic_xpi_installation_folder>¥InMemoryMiddleware¥agent**」にある imm-agent.exe を使用して IMM エージェントを起動します。「Handshake Success: OK」というメッセージが表示されます。これは、エージェントが IMM に正常に接続したことを意味します。
16. **MongoDB Compass** などの外部ツールを開きます。このように外部から Redis データにアクセスする場合、資格情報の提供が必須であることに注意してください。
17. 資格情報を入力します。



18. CA 証明書ファイルを選択します。



証明書ファイル **imm.pem** が選択されていることに注意してください。

また、MongoDB が認証を使用してアクセスされていることも確認してください。

SSL を使用しない IMM のデプロイ

CA 証明書が存在しないか無効である場合、デプロイプロセスは SSL なしでデプロイされます。**¥config** フォルダに証明書を追加する必要はありません。さらにエージェントについては **<Magic_xpa_installation_folder>¥InMemoryMiddleware¥agent** にある **.ENV** ファイルにフラグ **TLS_CA_FILE_PATH="<絶対パス¥CA_certificate_filename>"** を追加する必要はありません。

Magic xpi のインストール

1. Windows マシンに Magic xpi 4.14.1 バージョンをインストールします。詳細な手順については、Magic xpi インストール ガイドを参照してください。

2. SSL デプロイメントを利用するには、使用する JAVA cacerts ファイルに CA 証明書をインポートする必要があります。たとえば、次のコマンドを使用して証明書をインポートできます。:

keytool -importcert -file "c:\Magic xpi 4.14.1\InMemoryMiddleware\ config\imm.crt" -alias imm -keystore "c:\Magic xpi 4.14.1\ Runtime\ JDK8x64\jre\lib\security\cacerts"

ここで

- **"c:\Magic xpi 4.14.1\InMemoryMiddleware\config\imm.crt"** は証明書ファイルのパスです。
- **"c:\Magic xpi 4.14.1\Runtime\JDK8x64\jre\lib\security\cacerts"** は JAVA cacerts ファイルのパスです。

> デバッガー サービスを使用するには、32 ビット JAVA でも証明書をインポートする必要があります。

例:

```
keytool -importcert -file "c:\Magic xpi
4.14.1\InMemoryMiddleware\config\imm.crt" -alias imm -keystore "c:\Magic
xpi 4.14.1\Runtime\JDK8x86\jre\lib\security\cacerts"
```

HA ノードの障害耐性

3 ノード HA クラスタは、1 つのノードの障害のみを許容します。それ以上の障害が発生すると、クラスタが不安定になります。障害が発生したノードを適時に処理するために、適切な監視を行うことをお勧めします。

IMM のデプロイ

> IMM エージェントを実行しているマシンと IMM ホスト マシンの時刻とタイムゾーンの値は同じである必要があります。時刻とタイムゾーンの値が異なると、リカバリ ポリシーが正しく実行されない可能性があります。

IMM をデプロイするには、以下の手順を実行します。:

- \InMemoryMiddleware\deploy フォルダに移動します。
- コマンド プロンプトを開き、deploy-imm.bat バッチ ファイルを実行します。このファイルでは、以下の引数を順番に受け取ります。引数が指定されていない場合は、デフォルト値でクラスタがセットアップされます。

```
deploy-imm.bat <DOMAIN_NAME> <HELM_VERSION> <GLOBAL_SETTINGS_PATH>
<DB_ALERT_CONFIG_PATH> <LOGBACK_PATH> <MONITOR_CONFIG_PATH>
```

- **GLOBAL_SETTINGS_PATH**: globalsettings.properties のパスを指定します。このファイルでは、IMM の起動と初期化のための時間間隔とリトライメカニズムのプロパティを設定できます。
- **DB_ALERT_CONFIG_PATH**: db_alert_config.properties ファイルのパスを指定します。このファイルでは、IMM の重大な状態、CPU 使用率の高さ、メモリ使用量に関する電子メール アラートを設定できます。
- **LOGBACK_PATH**: logback.xml のパスを指定します。このファイルでは、ロガー、アペンダーを設定したり、さまざまな xpi コンポーネントのログ レベルを変更したりできます。
- **MONITOR_CONFIG_PATH**: モニタの設定ファイルのパスを指定します。このファイルでは、xpi モニタの SSO ログインを設定したり、ロケール (言語) を英語 (en-US) または日本語 (jp-JP) に変更したりできます。
- 引数が指定されていない場合、セットアップは config フォルダ内の構成ファイル (DB Alert を除く) から必要な値を取得します。DB Alert のデフォルト ファイルは、Runtime\config フォルダから取得されます。
- IMM セットアップが完了したら、次のコマンドを使用して、すべてのポッドが IMM クラスタ内で実行されているかどうかを確認できます。:

```
kubectl get pods -n magic-xpi-imm-ns
```

ブラウザで URL を実行して、すべてのサービスが実行されているかどうかを確認することもできます。

- **Magic モニタ:**

  https://[IMM Domain Name]/magicmonitor/#/auth/login

  例: https://xpi.magic.com/magicmonitor/#/auth/login

- **IMM-トンネル:**

  https://[IMM Domain Name]/immtunnel

  例: https://xpi.magic.com/immtunnel

AKS インスタンスで実行されているエージェントは、4 ～ 5 分間隔で IMM-DB サービスに再接続され、「Client could not connect(クライアントが接続できませんでした)」というエラーが表示される場合があります。この問題を解決するには、TCP タイムアウトを既定値の 4 からより大きな値に増やすか、TCP リセットを無効にします。

タイムアウトを構成するには、Microsoft の公式ドキュメントを参照してください。同じことを行うには、システム管理者の支援が必要になる場合があります。

IMM ホストへのアクセス

インストール中に指定した IMM ホスト アドレスがパブリックにアクセス可能なホストでない場合は、hosts ファイルで設定する必要があります。

テキストエディタで **C:\Windows\System32\drivers\etc** フォルダにある hosts ファイルを開きます。

このファイルに次の形式でエントリを追加します:

`<Kubernetes` クラスタの IP アドレス> <ホストドメイン名>

たとえば、ホストを xpi.magic.com として指定し、マシンの IP アドレスが 10.1.2.3 の場合、エントリは次のようになります

10.1.2.3 xpi.magic.com

エージェントの起動

エージェントを起動するには、

- **¥InMemoryMiddleware¥agent** フォルダに移動します。
- コマンド プロンプトを開き、**imm-agent.exe** ファイルを実行します。
- SSL を使用して IMM をデプロイしている場合は**.ENV** ファイルに以下のパラメータを追加します。:

```
TLS_CA_FILE_PATH="<path of the crt file>/gd_bundle-g2-g1.crt"
```

> IMM エージェントは、Magic xpi プロジェクトを実行またはデバッグする前に、常に実行状態になっている必要があります。

Windows で必要な Magic xpi の変更

IMM を SSL を使用して構成した場合、プロジェクトで設定された TCP トリガーなどの一部のトリガーがリクエストを処理しない可能性があります。これを修正するには、mgreq.ini に多少の変更を加える必要があります。

19. 以下の 2 つのフォルダに mgreq.ini ファイルが存在します。:

- <Magic xpi_Installation>\Runtime
- <Magic xpi_Installation>\Runtime\scripts\config

20. 双方の場所にあるファイルをテキスト エディタで開き、IMM_TUNNEL_HOST URL を http から https に変更します。
    たとえば、URL が

IMM_TUNNEL_HOST=http://xpi.factoryeye.net/immtunnel/

の場合、

IMM_TUNNEL_HOST=https://xpi.factoryeye.net/immtunnel/

に変更します。

この変更後、トリガはリクエストの処理を開始します。

xpi インストールでの IMM ドメイン名の管理

インストール中に指定する IMM ドメイン名は、以下にリストされている複数のファイルで参照されます。ドメイン名が変更された場合は、これらすべてのファイルで変更を行う必要があります。:

- [Magic xpi Installation]\Runtime\Magic xpa\magic.ini
- [Magic xpi Installation]\InMemoryMiddleware\agent\.env
- [Magic xpi Installation]\InMemoryMiddleware\immlogger\IMM_Controller_Log.bat
- [MagicxpiInstallation]\InMemoryMiddleware\imm-logger\IMM_Tunnel_Log.bat
- [Magic xpi Installation]\Runtime\MgxpiCmdl.bat
- [Magic xpi Installation]\Runtime\mgreq.ini
- [Magic xpi Installation]\Runtime\scripts\config\mgreq.ini
- [Magic xpi Installation]\xpi_webserver\webservice-config\config.properties
- [Magic xpi Installation]\Studio\Debugger\Install_Debug_services.bat
- [Magic xpi Installation]\Studio\Debugger\apache-tomcat\bin\service.bat

xpi インストールのファイルとは別に、hosts ファイルでもドメイン名を更新する必要があります。Magic xpi インストールには、imm-updater ユーティリティも同梱されており、これは、更新された IMM ドメイン名で上記で指定したファイルのリストを自動的に更新します。このユーティリティは、\InMemoryMiddleware\imm-updator にあります。このユーティリティを実行するための詳細な手順は、同じフォルダの readme.txt に記載されています。

複数のネームスペースが利用できる場合の IMM のデプロイ

デプロイ時に複数の名前空間が利用可能な場合は、次の手順を行います。:

21. imm-deploy.bat ファイルを開きます。
22. DOMAIN_NAME の値を変更します (ネームスペースのデプロイごとに必ず一意の DOMAIN_NAME を使用してください)。
23. NAMESPACE の値を変更します。
24. IMMDB_INGRESS_CONTROLLER_PORT と LOGDB_INGRESS_CONTROLLER_PORT の値を変更します。

25. 変更を保存します。

複数のネームスペースで作業している場合、ネームスペースの以前のデプロイメントをクリーンアップしたい場合、デプロイメントプロセスは **CoreDNSCache** と **xpi_ingress_controller** をアンインストールするかどうか確認します。**CoreDNSCache** と **xpi_ingress_controller** のインストールを削除しないことをお勧めします。これらを削除すると、他のネームスペースで実行されている IMM にアクセスできなくなります。

CoreDNSCache と **Ingress Controller** を削除することを選択した場合、他のネームスペースで実行されている IMM にアクセスするためには、**CoreDNSCache** と **xpi_ingress controller** をインストールする必要があります。

複数のネットワーカカードの設定

アプリケーション サーバに複数のネットワーク カードがある場合は、次のようにして、Magic xpi 4.x サーバに特定のカードを使用するように構成します。:

<Magic_xpi_installation>\InMemoryMiddleware \agent\.env ファイルにある **NIC_ADDR** の値を、このネットワーク カードに割り当てられた IP またはネットワーク カード自体の名前を保持するように変更します。

たとえば **NIC_ADDR=10.1.1.11** あるいは **NIC_ADDR="#eth0:ip#"**と指定します。ここで eth0 はネットワーク カードの名前です。

ホスト名または IP アドレスは引用符で囲まないでください。

EKS に IMM をデプロイするための前提条件

以下の必須要件を満たす必要があります:

1. EKS クラスタが構成されている必要があります。
2. クラスタに接続してアクセスするには、管理者権限が必要です。

EKS への接続

1. ローカルマシン/ジャンプサーバ/BASTON サーバ/管理者アクセスを持つその他のリモートサーバから AWS アカウント/クラスタに接続します。
2. .kubeconfig ファイルを更新します。

以下の構文を参照してください:

```
aws eks update-kubeconfig --region <region_name>
                                                    --name <cluster_name>
```

EKS への IMM のデプロイ

IMM をデプロイする方法は 2 つあります。–

1. **deploy-imm.bat** を実行する。
2. Magic Cloud Manager **(MCM)**を使用する。 IMM デプロイにおける cloud manager の役割。

MCM を使用して IMM をデプロイする

1. インストーラーが配置されている場所 (**Magic xpi 4.14.1\InMemoryMiddleware\deploy** など) を開きます。
2. バッチファイル **deploy-cloud-manager.bat** を実行します。

詳細は Magic xpi Cloud Native のドキュメントを参照してください。

How-To

1. **質問**：プロキシ サーバを使用してインターネットに接続するように Ubuntu サーバを構成するにはどうすればよいでしょうか？

   回答：Ubuntu システムがプロキシ サーバを使用してインターネットに接続されている場合、プロキシ サーバの情報は 2 つのファイルで提供される必要があります。

   a. /etc/environment ファイルにプロキシ サーバを追加する

      このファイルをテキストエディタで開き、以下の行を追加します。:

      http_proxy="<http://<Proxy Host>:<Proxy Port>"
      https_proxy="<http://<Proxy Host>:<Proxy Port>"
      HTTP_PROXY="<http://<Proxy Host>:<Proxy Port>"
      HTTPS_PROXY="<http://<Proxy Host>:<Proxy Port>"
      NO_PROXY=10.0.0.0/8,192.168.0.0/16,127.0.0.1,172.16.0.0/16,.svc,.svc.cluster.local

   b. MicroK8s の設定にプロキシ サーバを追加する

      /var/snap/microk8s/current/args/containerd-env ファイルをテキストエディタで開き、以下の行を追加します:

      http_proxy="<http://<Proxy Host>:<Proxy Port>"
      https_proxy="<http://<Proxy Host>:<Proxy Port>"
      HTTP_PROXY="<http://<Proxy Host>:<Proxy Port>"
      HTTPS_PROXY="<http://<Proxy Host>:<Proxy Port>"
      NO_PROXY=10.0.0.0/8,192.168.0.0/16,127.0.0.1,172.16.0.0/16,.svc,.svc.cluster.local

   上記の変更を加えて両方のファイルを保存したら、次のコマンドでコンテナ ランタイム サービスを再起動します:

   ```
   sudo systemctl restart snap.microk8s.daemon-containerd.service
   ```

2. **質問**：Windows システムからプロキシ サーバーを使用して IMM をデプロイするにはどうすればよいですか?

回答: IMM は、deploy-imm.bat を使用して Windows システムからデプロイされます。組織でプロキシ サーバを使用している場合は、IMM をインストールする前にプロキシ サーバを構成する必要があります。

プロキシ サーバを設定するには、次の手順を実行します。:

i. Windows コマンド プロンプトで次のコマンドを実行してプロキシを設定します:

`SET HTTPS_PROXY= <HTTP proxy の URL >`

ii. Windows コマンド プロンプトで次のコマンドを実行してプロキシをバイパスする IP アドレスを設定します:

`SET NO_PROXY = <MicroK8s` がインストールされている Ubuntu システムの IP アドレス>

これらの設定が完了したら、deploy-imm.bat ファイルを実行して IMM をセットアップできます。Windows 設定で、IMM がデプロイされているドメインを指定して、ブラウザから xpi モニタにアクセスするときにプロキシをバイパスします。これを行うには、

iii. **Windows のスタート メニュー > プロキシ**に移動します。

iv. プロキシメニュの**手動プロキシセットアップ**で、**プロキシ サーバを使う** オプションを有効にします。これにより、サブメニューが有効になります。 ここで、**アドレス**と**ポート**に必要な値を入力します。プロキシ除外設定のアドレスに、IMM がデプロイされているドメインを入力します。

> ドメインでデプロイがすでに完了している場合は、プロキシを使用する必要はありません。

または、Chrome > 設定 > コンピューターのプロキシ設定を開く オプションより移動することもできます。これにより、上記と同じプロキシ設定メニューが表示されます。

3. **質問**：IMM マルチノード クラスタ セットアップのアクティビティ ログをバックアップおよび復元するにはどうすればよいですか?

   回答: Magic xpi インストールには、アクティビティ ログをバックアップして復元するためのバッチ ファイルが含まれています。

   i. バックアップを作成する前に、最新バージョンの MongoDB ツールをインストールする必要があります。インストールするには、以下の公式 Mongo サイトにあるインストール手順に従ってください:

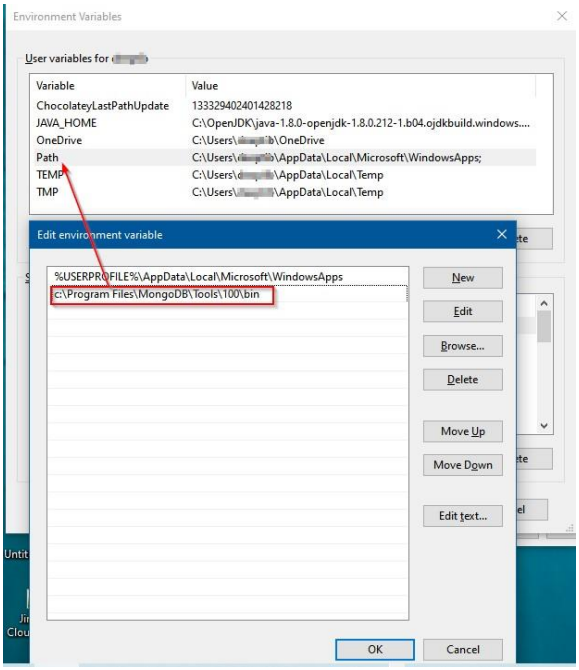
   https://www.mongodb.com/docs/databasetools/installation/installation/

   Windows OS にインストールする場合は、インストール後にシステム環境変数にインストール パスを二重引用符で囲んで追加します。:

   a) コントロール パネルに移動し、システム環境変数の編集 オプションを探します。または、コントロール パネル > システムとセキュリティ > システム > システムの詳細設定 を選択します。システムのプロパティ ウィンドウが開きます。

   b) 環境変数ボタンを選択します。システム環境ウィンドウが開きます。

   iii. Path 変数に移動し、編集ボタンをクリックして、MongoDB ツールのインストールパスを追加します。



iv. 次に、
<Magic_xpi_installation_folder>\InMemoryMiddleware\act_log_dump にあるバッチ ファイルに移動します。

v. ログをバックアップし、バックアップを復元するには、バッチ ファイル **act_log_backup.bat** を使用します。

4. **質問**：IMM が正常にデプロイされているかどうかを確認するにはどうすればよいですか?

回答: IMM が正常にデプロイされているかどうかを確認するには、次のコマンドを実行します:

```
kubectl get pods -n magic-xpi-imm-ns
```

```
C:\Users\dattatrays>kubectl get pods -n magic-xpi-imm-ns
NAME                                      READY   STATUS    RESTARTS   AGE
imm-db-0                                  1/1     Running   0          16h
logdb-5d4f8f99f8-z79lt                    1/1     Running   0          16h
imm-tunnel-deployment-6b784d449f-zk25j    1/1     Running   0          16h
xpi-monitor-6894b75bbd-729tg              1/1     Running   0          16h
imm-controller-76c4bc9db5-kg5k8           1/1     Running   0          16h
xpi-imm-server-deployment-6dd5bf78fb-vkj7f   1/1     Running   0          16h
```

あるいは

```
kubectl get pods -n magic-xpi-imm-ns –watch
```

```
C:\Users\dattatrays>kubectl get pods -n magic-xpi-imm-ns --watch
NAME                                      READY   STATUS    RESTARTS   AGE
imm-db-0                                  1/1     Running   0          16h
logdb-5d4f8f99f8-z791t                    1/1     Running   0          16h
imm-tunnel-deployment-6b784d449f-zk25j    1/1     Running   0          16h
xpi-monitor-6894b75bbd-729tg              1/1     Running   0          16h
imm-controller-76c4bc9db5-kg5k8           1/1     Running   0          16h
xpi-imm-server-deployment-6dd5bf78fb-vkj7f   1/1     Running   0          16h
```

このコマンドは、ネームスペース magic-xpi-imm-ns で実行されているすべてのポッドのリストを取得します。各ポッドの名前、ステータス、再起動回数などの基本情報が表示されます。

```
kubectl get pods -n magic-xpi-imm-ns -o wide
```

このコマンドは、magic-xpi-imm-ns ネームスペース内のポッドに関する情報を取得しますが、追加の詳細はワイド形式で提供されます。ワイド形式は、ポッドのスケジュール情報やノード割り当てなど、ポッドの包括的なビューを取得するのに役立ちます。

5. **質問**：Kubernetes が正常に動作しているかどうかを確認するにはどうすればいいですか?
   回答：Kubernetes が正常に動作しているかどうかを確認するには、次のコマンドを実行します:

```
kubectl get pods -n kube-system
```

このコマンドは、kube-system ネームスペースで実行されているすべてのポッドのリストを取得します。Kubernetes の kube-system ネームスペースには通常、Kubernetes コントロール プレーン、ネットワーク、監視、ログなどの重要なシステム コンポーネントとインフラストラクチャ ポッドが含まれています。これらのポッドは、Kubernetes クラスタが適切に機能するために不可欠です。

```
C:\Users\dattatrays>kubectl get pods -n kube-system
NAME                                       READY   STATUS    RESTARTS       AGE
coredns-6f5f9b5d74-bzqlw                   1/1     Running   18 (12d ago)   117d
calico-kube-controllers-65687f8779-v8ql7   1/1     Running   13 (12d ago)   38d
node-local-dns-bc7ch                       1/1     Running   13 (12d ago)   33d
coredns-6f5f9b5d74-lgghs                   1/1     Running   13 (12d ago)   62d
coredns-6f5f9b5d74-2ljbf                   1/1     Running   13 (12d ago)   62d
calico-node-74spc                          1/1     Running   13 (12d ago)   38d
metrics-server-6f754f88d-j87w5             1/1     Running   13 (12d ago)   49d
```

```
kubectl get pods -n kube-system -o wide
```

このコマンドは、kube-system ネームスペースで実行されているすべてのポッドのリストを取得しますが、追加の詳細がワイド形式で提供されます。ワイド形式は、ポッドのスケジュール情報やノードの割り当てなど、ポッドのより包括的なビューを取得するのに役立ちます。

```
C:\Users\dattatrays>kubectl get pods -n kube-system -o wide
NAME                                       READY   STATUS    RESTARTS       AGE    IP             NODE              NOMINATED NODE   READINESS GATES
coredns-6f5f9b5d74-bzqlw                   1/1     Running   18 (12d ago)   117d   10.1.125.48    xpiqaubuntuvm38   <none>           <none>
calico-kube-controllers-65687f8779-v8ql7   1/1     Running   13 (12d ago)   38d    10.1.125.33    xpiqaubuntuvm38   <none>           <none>
node-local-dns-bc7ch                       1/1     Running   13 (12d ago)   33d    10.9.3.78      xpiqaubuntuvm38   <none>           <none>
coredns-6f5f9b5d74-lgghs                   1/1     Running   13 (12d ago)   62d    10.1.125.10    xpiqaubuntuvm38   <none>           <none>
coredns-6f5f9b5d74-2ljbf                   1/1     Running   13 (12d ago)   62d    10.1.125.3     xpiqaubuntuvm38   <none>           <none>
calico-node-74spc                          1/1     Running   13 (12d ago)   38d    10.9.3.78      xpiqaubuntuvm38   <none>           <none>
metrics-server-6f754f88d-j87w5             1/1     Running   13 (12d ago)   49d    10.1.125.42    xpiqaubuntuvm38   <none>           <none>
```

Magic xpi のインストール

1. Amazon EC2 インスタンスに Magic xpi 4.14 .1 バージョンをインストールします。詳細な手順については、Magic xpi インストール ガイドを参照してください。

2. すべての HA アーティファクトがインストール ディレクトリの下のそれぞれの場所にコピーされていることを確認します。

> • 詳細および上記の HA アーティファクトのコピーについては、HA セットアップのインストール フォルダにある InstallationGuide.txt を参照してください。
>
> · Subnet_ID_1, Subnet_ID_2 はパブリックである必要があります。

EKS クラスタへの IMM のデプロイ

1. IMM のデプロイの前に、HA ビルドと一緒に提供されている **CustomIngressConfiguration.yaml** ファイルを以下の場所にコピーします:
   /InMemoryMiddleware/Deploy
   このファイルをテキスト エディタで開き、CustomIngressConfiguration.yaml で token をセキュリティ グループ ID に、tokens をサブネット ID に置き換えます。変更が完了したら、ファイルを保存します。

2. コマンド プロンプトを開き、deploy-imm.bat を実行します。デプロイメント タイプ (シングルノード、マルチノード) を選択するように求められます。デフォルトのデプロイメントはマルチノードです。マルチノード クラスタで続行するには、y と入力して Enter キーを押します。

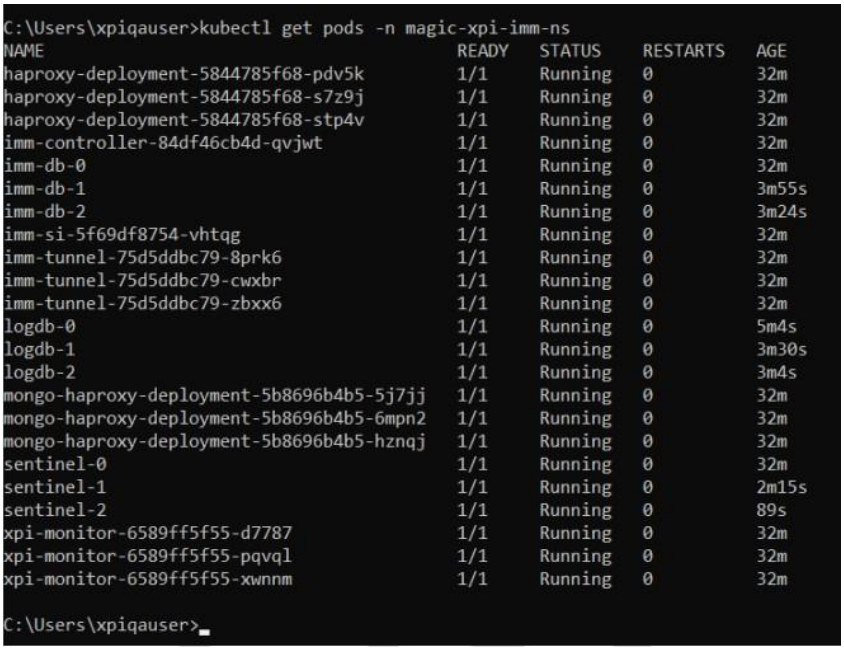
3. デプロイメントが完了したら、x を押して終了します。 これで、マルチノード クラスタのデプロイメントの準備が整いました。

4. すべてのポッドが稼働中であることを確認するには、コマンドを入力します:

   ```
   kubectl get pods -n
   ```

たとえば、ネームスペースが magic-xpi-imm-ns の場合、コマンドは次のようになります:

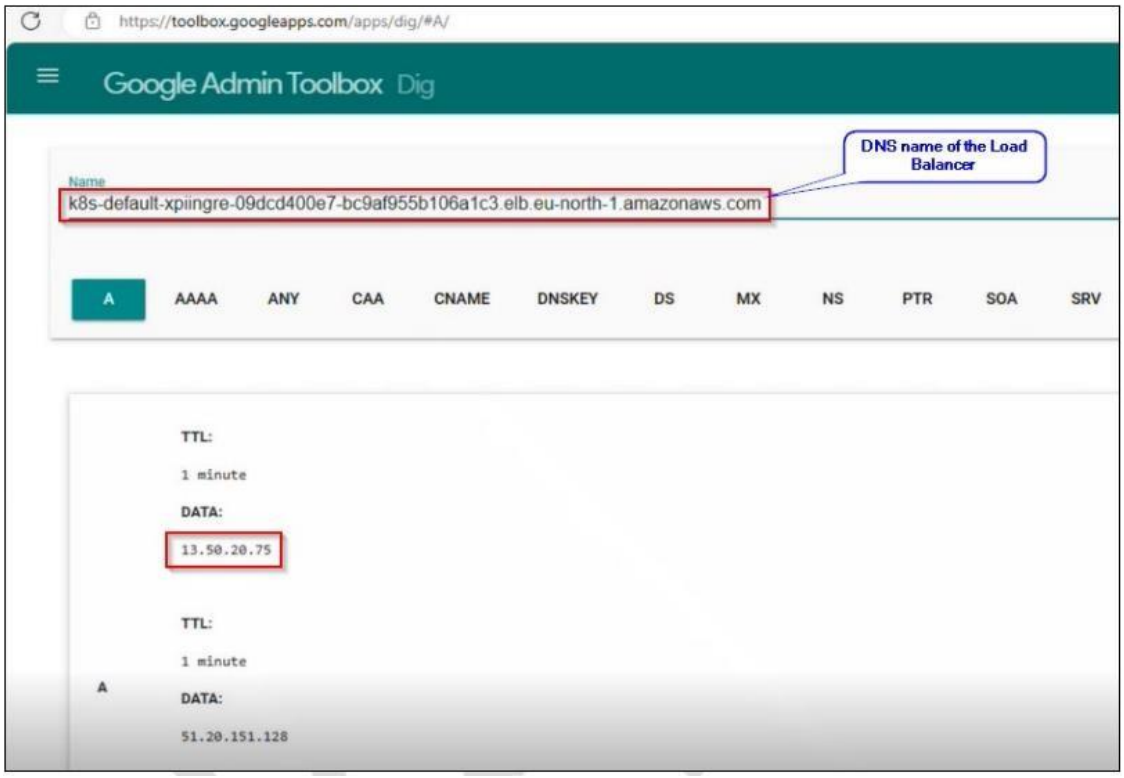
```
kubectl get pods -n magic-xpi-imm-ns
```



EKS クラスタ IP アドレスの取得

xpi モニタやその他の xpi サービスにアクセスするには、EKS クラスタの IP アドレスが必要です。IP アドレスを取得するには、次の手順を実行します。:

1. AWS ポータルにログインし、ロード バランサを検索します。
2. 適切なロード バランサを選択して開きます。
3. ロード バランサの詳細で DNS 名を探します。
4. DNS 名をコピーします。
5. 次に、ブラウザで次のリンクを開きます:
   https://toolbox.googleapps.com/apps/dig/
6. このポータルが開いたら、テキスト フィールドに DNS 名を貼り付けて検索します。
7. IP アドレスのリストが開きます。最初の IP アドレスをコピーします。この IP アドレスは EKS クラスタの外部 IP アドレスです。



IP アドレスを取得したら、Windows マシンの hosts ファイルに IP アドレスのエントリを追加する必要があります。そのためには、次の操作を行います。:

1. C:\Windows\System32\drivers\etc フォルダに移動します。
2. hosts ファイルをテキストエディタで開きます。
3. 次の形式でエントリを追加します。例：

   ```
   1.2.1.1 mydomainname
   ```

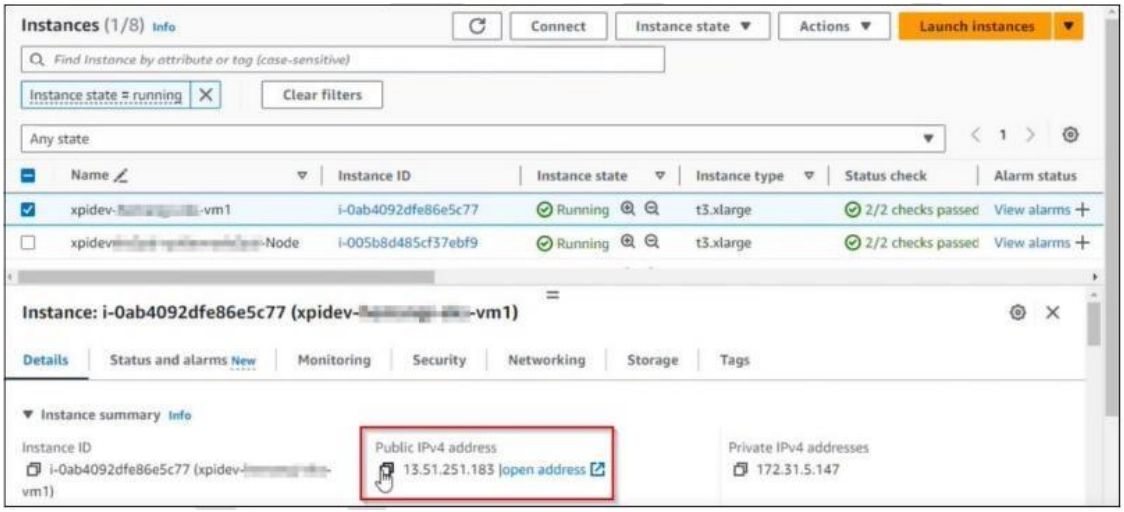
ドメイン名は、Magic xpi のインストール時に指定したものと同じであることに注意してください。

セキュリティグループ(Security Group)の設定(This section needs change)

セキュリティ グループは、受信トラフィックと送信トラフィックを制御します。クラスタを作成する前に、一連のルールを持つセキュリティ グループを作成する必要があります。まず、Magic xpi がインストールされ、IMM エージェントがホストされる EC2 インスタンスの IP アドレスが必要です。IP アドレスを見つけるには、次の手順を実行します。:

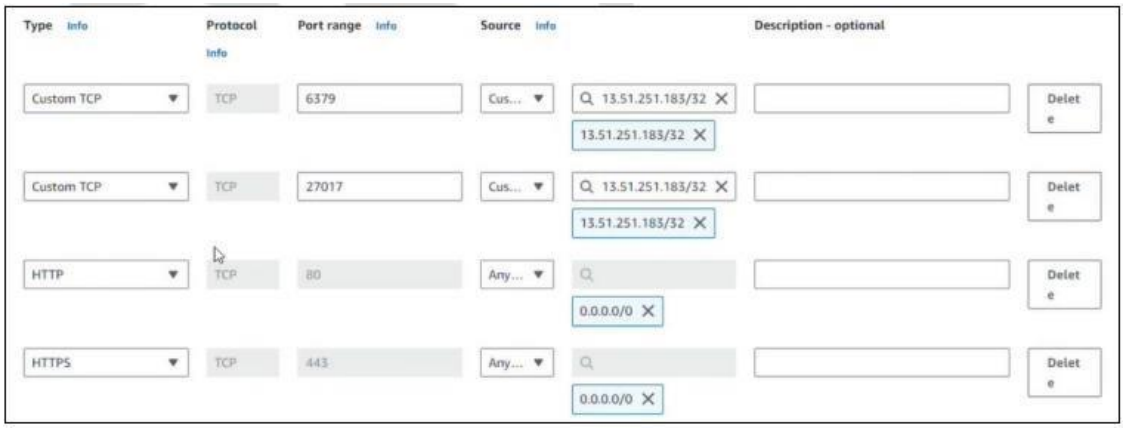
1. AWS ポータルにログインし、EC2 を検索します。
2. インスタンスに移動して、必要なマシンを選択します。

3. このマシンのパブリック IP アドレスをコピーします。 これは、セキュリティ グループを定義するときに必要になります。



IP アドレスを取得したら、セキュリティ グループの作成に進みます。セキュリティ グループを作成するには:

1. AWS ポータルにログインし、セキュリティ グループを検索して、セキュリティ グループ セクションに移動します
2. 新しいセキュリティ グループを作成し、必要な名前と説明を入力します。VPC は自動的に選択されます。
3. 次の内容を含む 4 つのインバウンドルールを追加します:
   Rule 1: Type: Custom TCP Port Range: 6379 IP Address: /32
   Rule 2: Type: Custom TCP Port Range: 27017 IP Address: /32
   Rule 3: Type: HTTP Source: Anywhere-IPv4
   Rule 4: Type: HTTPS Source: Anywhere-IPv4.



4. 4 つのルールがすべて追加されたら (上の画像を参照)、一番下までスクロールし、セキュリティ グループの作成 ボタンをクリックしてセキュリティ グループを保存します。

カスタムストレージクラス(Custom Storage Class)を使用して EKS クラスタに IMM をデプロイする

1. IMM をデプロイする前に、HA ビルドと一緒に提供された CustomIngressConfiguration.yaml ファイルを以下の場所にコピーします: <Magic xpi installation>/InMemoryMiddleware/Deploy
2. このファイルをテキストエディタで開き、CustomIngressConfiguration.yaml ファイルで <SECURITY_GROUP_ID> トークンをセキュリティ グループ ID に、<SUBNET_ID_1>、<SUBNET_ID_2>、<SUBNET_ID_3> トークンをサブネット ID に置き換えます。
3. ファイルを保存します。

AWS での Elastic File Store (EFS) の構成

このセクションでは、AWS で Elastic File Store (EFS) を設定する手順と、Magic xpi Cloud バージョン 4.14.1 で設定するために必要な変更について説明します。

前提条件

1. AWS 上のインフラストラクチャで EFS が利用可能なこと。
2. 永続ボリュームと永続ボリューム クレームに対して RWX モードが有効になっている EFS であること。

AWS インフラストラクチャでこのソリューションを構成、デプロイ、および提供するには、次の手順を実行してください

1. Magic xpi AWS 互換ストレージ クラス ソリューションを構成します。
2. AWS に Magic xpi をデプロイします。

AWS クラウド インフラストラクチャで実行するための Magic xpi 4.14.1 の準備

1. EFS ストレージにマウントするために必要な次のフォルダ構造を次のように作成します:
2. AWS 上の EKS クラスタのマスター ノードに接続します。
3. ルートの場所「/」に移動します。
4. 以下のコマンドを実行します:

```
mkdir -p /mnt/efs
```

5. /mnt/efs に移動して、次のコマンドを実行します:

```
sudo mount -t nfs4 -o nfsvers=4.1 <DNS of your EFS>://mnt/efs
```

6. Magic xpi IMM に必要なフォルダ構造を作成します。
7. /mnt/efs の場所から次のコマンドを実行します:

```
mkdir -p home/data
mkdir -p home/mgxpi-monitor/app/config
mkdir -p home/mgxpi-imm-server/app/logback

mkdir -p home/mgxpi-imm-server/app/dbalertconfig
```

8. 最後に以下のコマンドを実行します。:

```
sudo umount /mnt/efs
```

9. Magic xpi 4.14.1 をインストールします。
10. <インストール先>\Magic xpi 4.14.1\InMemoryMiddleware \deploy フォルダに移動します。
11. フォルダ内の values.yaml ファイルを編集します。
12. Magic xpi 4.14.1 IMM の RWX ストレージとして EFS を使用するために作成した EFS 準拠ストレージ クラス名を設定します。

例:

```
timeZone: &timeZone "US/Central"

        global:

        STORAGECLASS: efs-sc
```

EFS ストレージ クラス定義のサンプルは次のとおりです。:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass metadata:
  name: efs-sc parameters:
  directoryPerms: "777"
fileSystemId: fs-047d27b1cb47695d8
provisioningMode: efs-ap provisioner:
efs.csi.aws.com reclaimPolicy: Retain
volumeBindingMode: Immediate
```

13. deploy_imm.bat を実行します。これにより、AWS インフラストラクチャで RWX サポートを備えた EFS ストレージクラスと永続ボリュームクレームを使用して環境がセットアップされ、実行されます。

AWS ロードバランサー インフラストラクチャのセキュリティ グループでポートが開いていることを確認してください。

IMM のデプロイ

1. IMM をデプロイする前に、HA ビルドと一緒に提供される CustomIngressConfiguration.yaml ファイルを以下の場所にコピーします: **<Magic xpi installation>/InMemoryMiddleware/Deploy**
2. このファイルをテキスト エディタで開き、**CustomIngressConfiguration.yaml** 内の <SECURITY_GROUP_ID> トークンをセキュリティ グループ ID に、<SUBNET_ID_1>、<SUBNET_ID_2>、<SUBNET_ID_3> トークンをサブネット ID に置き換えます。
3. ファイルを保存します。

外部 NLB を使用した AWS クラウドへの IMM のデプロイ

このセクションでは、AWS インフラストラクチャで構成された既存のネットワーク ロード バランサ (NLB) を使用して Magic xpi 4.14.1 IMM をアタッチしてデプロイする手順について説明します。

前提条件の準備

1. 以下のコマンドで**<InstallationDirectory>\InMemoryMiddleware\deploy** フォルダにある yaml ファイルを実行します:

   ```
   kubectl apply -f tgb-prod-magic-xpi-http.yaml
   ```

2. 特に HTTP、HTTPS、TCP ポート 80、443、6379、および 27017 の **Target Group Binding(ターゲットグループ バインディング)**を作成するための YAML アーティファクトを実行します。

   ここで、YAML でのバインディングに必要なターゲット グループ ARN を取得する必要があります。

前提条件

1. **AWS** インフラストラクチャ内に **NLB** が準備されていること。
2. **HTTP、HTTPS、TCP ポート 80、443、6379、27017 に対してターゲットのない個別のターゲット グループ**が手動で作成され、作成された **NLB** にアタッチされていること。

IMM のデプロイ

1. deploy_imm.bat を実行します。これにより、AWS インフラストラクチャ上に自動的に作成される専用の Ingress コントローラとともに IMM が起動して実行されます。
2. 開きたい関連ポートのターゲット グループを AWS インフラストラクチャ内に作成します。

> ターゲット グループのタイプは理想的には IP タイプです。たとえば、通信ポート 80、443、6379、および 27017 を開きたい場合、IP タイプで 4 つのターゲット グループ (各ポートに 1 つ) を作成する必要があります。

3. **ターゲット グループ**が設定されたら、それぞれのポートに対して **TargetGroupBinding (TGB)** yaml ファイルを 1 つずつ実行します。
   以下はサンプルの **TargetGroupBinding** ファイルです。:

```
apiVersion: elbv2.k8s.aws/v1beta1
       kind: TargetGroupBinding
       metadata:
         name: my-tgb
       spec:
         serviceRef:
             name:awesome-service
              port:80
         targetGroupARN: <arn-to-targetGroup>
         vpcID: <vpcID>
```

1. ファイル テンプレートは、**<Installation Folder>\InMemoryMiddleware\deploy** にあります。これらのテンプレート ファイル名は **tgb-***.yaml です。
2. AWS インフラストラクチャで既に作成してあるターゲット グループの ARN をコピーします。
3. 各 **TargetGroupBinding の yaml** ファイル内の ARN を更新します。
4. ファイルの準備ができたら、yaml ファイルの場所にある **CLI** を開きます。
5. 次のコマンドを実行して、通信用に開くそれぞれのポートの yaml ファイルを 1 つずつ実行します。:

```
kubectl apply -f <TargetGroupBinding file yaml name> -n <namespace>
```

6. サブドメインとマップされた NLB DNS を使用して、**CloudFlare** に **CNAME** を追加します。

7. 例:magicxpi.magicsoftware.com nlb-qa-only-855f2b6176b4d6f5.elb.eu-west-1.amazonaws.com

NLB を手動で作成するために選択されたネットワーク サブネットはパブリック
である必要があります。

トラブルシューティング

免責事項 - 以下の質問と回答は非常に特定のシナリオに関するものである可能性があります。これらが問題解決に役立たない場合は、さらなる支援のために MSE サポートにお問い合わせください。

1. **質問**: IMM のデプロイが「Problem: Faced error “Error: INSTALLATION FAILED: unexpected status from HEAD request to https://devmcsworkspaceacr.azurecr.io/v2/magic-xpi-helm/xpi-ingress routes/manifests/4.14.02: 401 Unauthorized ERROR: Failed to deploy xpi-imm. Exiting setup.”?(問題: エラーが発生しました。エラー: インストールに失敗しました: HEAD 要求から https://devmcsworkspaceacr.azurecr.io/v2/magic-xpi-helm/xpi-ingress-routes/manifests/4.14.02 への予期しないステータス: 401 権限がありません。エラー: xpi-imm のデプロイに失敗しました。セットアップを終了します。)」というエラーで失敗するのはなぜですか?

回答: このエラーは、Helm パッケージ マネージャの特定のバージョンによって発生します。このエラーを修正するには、deploy フォルダで **inst_k8s_tools.bat** を実行してください。

2. **質問**: IMM デプロイメント スクリプトが停止し、長時間応答しないのはなぜですか?

   回答: この問題を解決するには、IMM Kubernetes クラスタがインストールされている Linux マシンにログインする必要があります。ターミナルから以下のコマンドを実行します。

```
NS=`microk8s kubectl get ns |grep Terminating | awk 'NR==1
{print $1}'` && microk8s kubectl get namespace "$NS" -o
json | tr -d "\n" | sed "s/\"finalizers\":
\[[^]]\+\]/\"finalizers\": []/" | microk8s kubectl replace
--raw /api/v1/namespaces/$NS/finalize -f -
```

これらのコマンドを実行すると、スクリプトの実行が続行されます。

4. **質問**: **letsencrypt.org** によって生成された証明書を使用して SSL でデプロイする場合、fullchain.crt を使用して imm-agent を起動する方法は？:

   回答: **letsencrypt.org** によって生成された証明書には、**.crt**、**.key**、および **fullchain.crt** ファイルが含まれます (このファイルには 2 つの証明書が含まれており、CA ルート証明書は含まれていません)。以下は、**fullchain.crt** に CA ルート証明書が含まれていないため (デフォルト)、imm-agent が接続されない場合の手順です。:

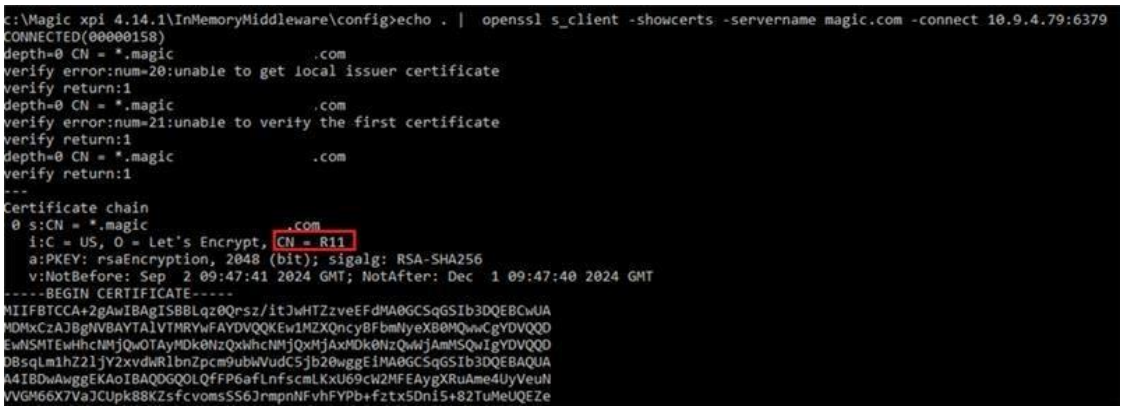
   i. 以下のコマンドでルート証明書の共通名を確認します。:

   ```
   echo . |  openssl s_client -showcerts -servername
   <domain_name> -connect <Static_IP>:6379
   ```

   例：

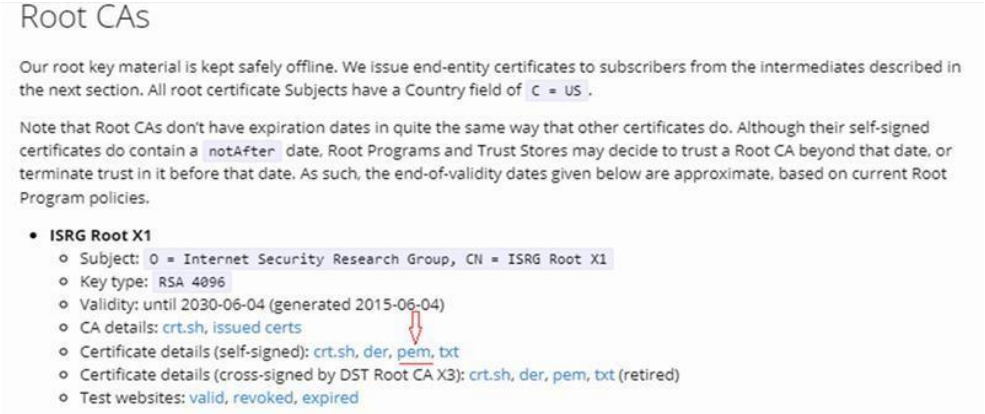
   ```
   echo . |  openssl s_client -showcerts -servername magic.com -
   connect 10.9.4.79:6379
   ```

   以下のような結果が得られます:



   ii. CN の値に応じて、このページに移動します: [Chains of Trust - Let's Encrypt]

iii. 図から **Root CA(ルート CA)** を確認し、それぞれの **.pem** ファイルをダウンロードします。例：ページのルート CA セクションから:



iv. **.pem** ファイルの内容をコピーし、**fullchain.crt** ファイルの内容に追加します。

v. fullchain.crt のパスを <**Magic xpi 4.14.1>\InMemoryMiddleware\agent\.env** ファイルの **TLS_CA_FILE_PATH** パラメータに設定します。

vi. **imm-agent.exe** あるいは Windows サービスを起動します。

5. **質問**: CA 証明書を使用して IMM エージェントを IMM as a Service に接続するにはどうすればいいですか?

   回答: IMM エージェントを IMM に接続するには、TLS デプロイを使用して証明書を検証している場合、<Magic_xpi_Installation_Folder>\InMemoryMiddleware\agent\.env ファイルの TLS_CA_FILE_PATH フラグに有効な CA 証明書パスを設定してください。

   > 証明書を検証しない場合は、<Magic_xpi_Installation_Folder>\runtime\OS_Service\bin\win64\imm_agent_service.xml ファイルで NODE_TLS_REJECT_UNAUTHORIZED の値をゼロに設定し、imm-agent サービスを再起動します。

6. **質問**: ファイアウォールをオンにすると、どのポートを開く必要がありますか?

   回答: ファイアウォールをオンにすると、次のポートを開く必要があります。:

   i. Microk8s のすべてのポートを有効にします。詳細については、次を参照してください: https://microk8s.io/docs/services-and-ports

ii. 次のコマンドでポートを開くことができます。:sudo ufw allow <port_number>

iii. Redis の IMM ポート

iv. 全マシンのポート 16443

これらのポート以外に、プロジェクトで他のポートを開く必要がある場合は、ファイアウォールを通過させてください。マシンがファイアウォール/プロキシの背後にある場合、各 Windows および Ubuntu システムの URL リストは次のとおりです。:

Windows の URL

i. *.chocolatey.org

ii. *.get.helm.sh

iii. public.ecr.aws/mgcproductionecr

Ubuntu の URL

i. *.snapcraft.io

ii. *.docker.io

iii. *.k8s.io

iv. *.asia-south1-docker.pkg.dev

この URL は地域によって異なります。IT 専門家にご相談ください。*.*-docker.pkg.dev のようなより広い範囲は、一部のプロキシで機能する場合があります。

i. *.docker.com

ii. *.quay.io

iii. *.snapcraftcontent.com

iv. *.amazonaws.com

7. **質問**: 以前のデプロイメントから残っているデータを消去するにはどうすればよいですか?

回答：imm ノードにアクセスできるホストから次のコマンドを実行します:

i. まず、LOGDB ポッド名 <logdb_pod_name> を特定します:

```
kubectl get pod -l app=xpi-logdb-deployment -n magic-xpi-imm-ns
```

ii. **ifs_actlog** データを削除するには:

```
kubectl exec -it -n magic-xpi-imm-ns <logdb_pod_name> -- mongosh
MGXPI414 -u <User_Name> -p <Password> --authenticationDatabase admin
--eval 'db.ifs_actlog.deleteMany({})'
```

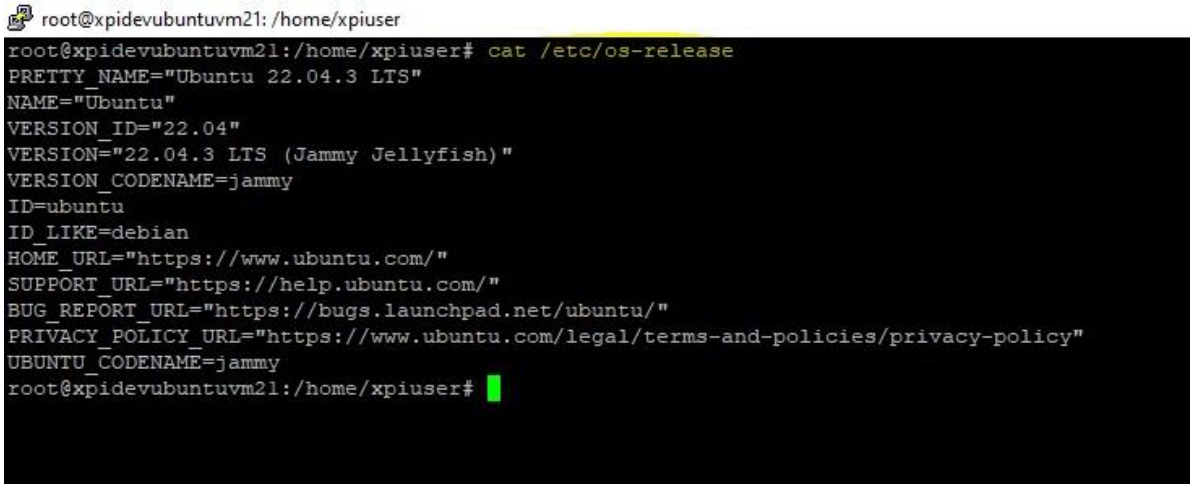
iii. **ifs_ods** データを削除するには:

```
kubectl exec -it -n magic-xpi-imm-ns <logdb_pod_name>  -- mongosh
MGXPI414 -u <User_Name> -p <Password> --authenticationDatabase admin
--eval 'db.ifs_ods.deleteMany({})'
```

7. **質問**: Ubuntu のバージョンを確認するにはどうすればよいですか?

   回答: Ubuntu のバージョンを確認するには、次のコマンドを実行します:

   cat /etc/os-release



8. **質問**: Ubuntu システムにウイルス対策ソフトウェアがインストールされているかどうかを確認するには、どうすればよいでしょうか?

   回答: ウイルス対策ソフトウェアがインストールされているかどうかを確認するには、次のコマンドを実行します:

   ```
   dpkg -l | grep -i <antivirus-name>
   ```

   上記のコマンドを実行した後に何らかの出力が表示される場合、それはアンチウイルスソフトウェアがインストールされていることを意味し、それを無効にするか削除する必要があります。

9. **質問**: Microk8s のインストール中にウイルス対策が無効になっていることを確認するにはどうすればよいですか?

   回答：Microk8s をインストールし、Microk8s 上に IMM をデプロイする際は、Defender ソフトウェアなどのアンチウイルスサービスを無効にしてください。IMM のデプロイが成功した後にアンチウイルスサービスを有効にすることができます。：Microk8s をインストールし、Microk8s 上に IMM をデプロイする際は、Defender ソフトウェアなどのアンチウイルスサービスを無効にしてください。IMM のデプロイが成功した後にアンチウイルスサービスを有効にすることができます。

10. **質問**: Ubuntu システムからアンチウイルスを無効にするにはどうすればよいですか？

    回答: システムのアンチウイルスを無効にするには、次のコマンドを実行してください:

    ```
    sudo systemctl disable [your-antivirus-service]
    sudo systemctl stop [your-antivirus-service]
    ```

    IMM を正常にインストールした後、ウイルス対策ソフトウェアをインストールする必要があります。または、すでにインストールされている場合は、上記のコマンドを使用して無効にしてください。

    ソフトウェアを削除する場合は、次のコマンドを使用してください:

    ```
    sudo apt purge <antivirus-name>
    ```

11. **質問**: microk8s クラスターをインストールして IMM を展開するには、どの URL にアクセス可能でなければなりませんか?

    回答: Docker イメージやその他の成果物をダウンロードするには、すべてのマシンでインターネット接続が可能であることを確認してください。次の URL は Linux ホストからアクセス可能でなければならず、ホワイトリストに登録されている必要があります。

    - docker.io
    - github.com
    - devmcsworkspaceacr.azurecr.io
    - quay.io
    - k8s.io
    - get.helm.sh
    - api.snapcraft.io
    - dashboard.snapcraft.io

- community.chocolatey.org
- snapcraftcontent.com

12. **質問**: ImagePullBackOff または CrashLoopBackoff 関連のエラーを調査するにはどうすればよいですか?

    回答:

    i. 次のコマンドを実行してポッドの状態を確認します:

    kubectl describe pod <pod-name> -n <namespace>

    このコマンドは、現在の状態、イベント、構成など、ポッドに関する詳細情報を提供します。

    ii. 次のコマンドを実行し、アプリケーション ポッドまたはシステム ポッドのログを取得します:
    kubectl logs <pod name> -n <namespace> > path\to\filename.log

13. **質問**: Actlog_Write_settings.xml と Actlog_Display_settings_admin.xml がプロジェクト ディレクトリで更新されていない場合はどうすればよいですか?

    回答: IMM デプロイ時、xpi モニタのアクティビティ表示設定は xpi モニタのインターフェイスを使用して行うことをお勧めします。プロジェクト構成またはデプロイ中に構成ファイル (Actlog_Display_settings_admin.xml と Actlog_Write_settings.xml) に加えられた変更は機能しません。IMM ノードのローカル コピーに加えられた変更は、再デプロイ後は保持されません。

14. **質問**: WSL を使用して Windows Server 2022 経由で IMM をインストールして実行することは可能ですか?

    回答: Azure でホストされている Windows Server 2022 で Hyper-V セットアップを実行する場合、Standard_D4s v3 以上の構成のみがサポートされます。その他の構成では、IMM を正しく実行するために必要な仮想化レイヤーがサポートされない可能性があります。

15. **質問**: Redis による CPU 使用率の高さ、xpi サーバの継続的なクラッシュ、IMM の不安定さなどのさまざまな問題に対処するにはどうすればよいですか?

回答: 以下のチェックリストに従って、収集したすべての情報を問題に添付してください:

i. 問題が再現された正確な日付と時刻を記入してください。

ii. 次のコマンドを実行し、その結果のスクリーンショットを記入してください:

```
kubectl top pod -n magic-xpi-imm-ns
kubectl top pod -n kube-system
kubectl get pods -A
```

> 上記のコマンドで「error: Metrics API not available(エラー: Metrics API が利用できません)」というエラーが発生した場合は、Linux マシンで次のコマンドを使用して Matrix API を有効にします:
> `Microk8s enable metrics-server`

iii. 次のコマンドを実行し、その結果を提供してください。:

```
kubectl get events -A
```

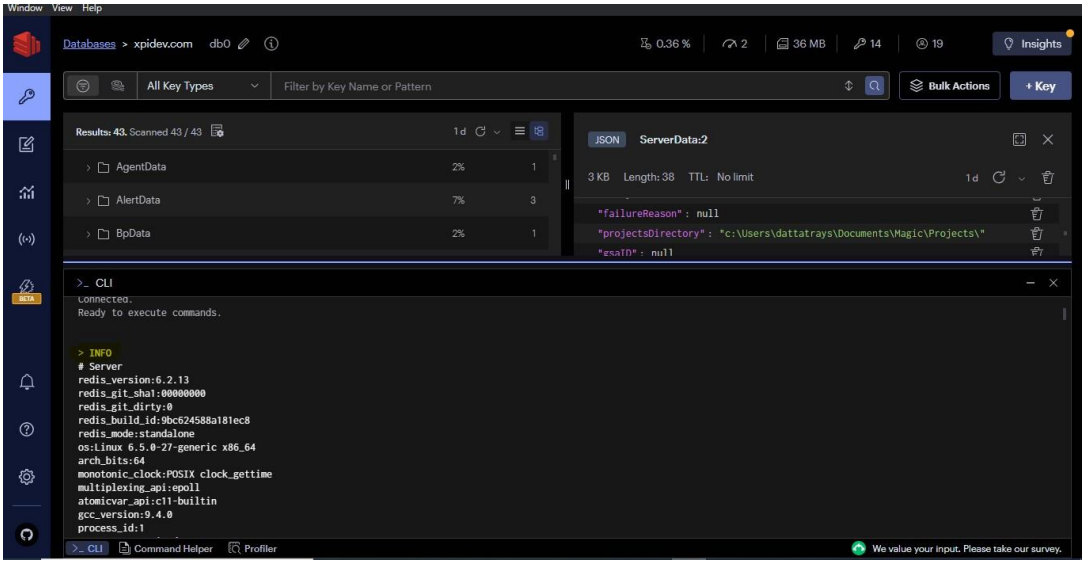
iv. システム ログとアプリケーション ログを含むすべてのポッドのログを取得してください。ログを取得するには、次のコマンド構文を使用します。:

```
kubectl logs <pod name> -n <namespace>
> path\to\filename.log
```

例:

```
kubectl logs imm-tunnel-deployment-7b79f4655f-9zmmn -n
magic-xpi-imm-ns > c:\temp\imm_tunnel_log.log

kubectl logs coredns-6f5f9b5d74-tgtv4 -n kube-system >
c:\temp\coreDNs.log
```

v. Redis CLI にコマンド「INFO」を入力し、このコマンドの出力結果を提供してください。詳細については以下の画像を参照してください。



vi. すべてのプロジェクトを停止し、Redis の CPU 使用率が高い問題が解決されたかどうかを確認します (5 分間待機します)。

はい/いいえ

vii. 手順 5 の後も Redis の CPU 使用率が高い問題が解決しない場合は、次のコマンドを実行します:

```
kubectl delete pod imm-db-0 -n magic-xpi-imm-ns
```

viii. 手順 5 または手順 6 のいずれかで Redis の CPU 使用率が高い問題が解決した場合は、すべてのプロジェクトを再起動し、すべてが正常に動作しているかどうかを確認します。

16. IMM Ubuntu に関するいくつかの質問

i. **質問**：同じハイパーバイザー上で異なる IP アドレス（192.17.0.11）とホスト名（imm.xpi.net）を持つ 2 つの IMM VM（2 つの vhdx イメージ）を実行することは可能ですか？

回答: はい、可能です。以下の手順で行ってください:

a. 「sudo nano /etc/netplan/00-installer-config.yaml」を実行し、新しい IP に変更します。

b. 「sudo netplan apply」を実行します。

c. 必要に応じて VM を再起動します。

d. コマンド「sudo microk8s reset」を実行して Microk8s をリセットします。

e. コマンド「sudo microk8s enable metallb dns storage」を実行して metallb と dns を有効にします。

f. コマンド「microk8s config view」を実行して、Windows からローカル マシンに新しい構成ファイルを取得します。

g. IMM を再デプロイします。

ii. **質問**: Ubuntu サーバの詳細なインストール手順を知ることは可能ですか?「デフォルト」インストールを行うことはできますか、それとも「HWE カーネル付きの Ubuntu サーバ」を選択する必要がありますか?

回答：HWE はスキップしてもよく、プロダクションサーバーではスキップすべきです。

iii. **質問**：IMM は 24.04 LTS でサポートされていますか？このバージョンの「サポート終了日」は 22.04 LTS よりも良いですか？

回答: 問題なく動作するはずです。

17. Magic xpi 4.14 SP オンプレミス Ubuntu Linux: セキュリティ管理者の質問

i. **質問**: どのバージョンの Redis と MongoDB を使用すればよいですか?

回答: 回答 Redis バージョン = 6.2.13 MongoDB バージョン = 6.0.7

ii. **質問**: Redis と MongoDB のセキュリティ/更新パッチ処理はどのように行いますか?
回答：特定のタグの Docker イメージを使用します。

iii. **質問**: Redis または MongoDB の新しいバージョンが利用可能になった場合、自動的に更新が行われますか行われますか、それとも手動で行う必要がありますか?

回答: IMM の Redis および mongo イメージを更新することは必須ではありません。ただし、更新したい場合は (推奨されません)、手動で 1 つの kubectl コマンドを実行することで更新できます。

iv. **質問**: デプロイ前に Docker イメージのセキュリティはどのようにチェックされますか?
回答:デプロイするイメージは、Clair や Trivy などのツールで脆弱性がないかスキャンされます。

v. **質問**: イメージを展開する前にスキャンするのにどのツールを使用しますか?
回答: Docker デスクトップを使用して、Docker イメージの脆弱性をチェックしました。

vi. **質問**: Kubernetes クラスタとの間で送受信されるデータやポッド間のデータのセキュリティをどのように確保しますか? 使用するプロトコルと暗号化の種類は何ですか?
回答: HTTPS

vii. **質問**: ポッド内の永続データのセキュリティをどのように確保しますか? どのプロトコルと暗号化タイプを使用しますか?
回答: クラウドデプロイの場合、EFS と S3 BLOB でのデータの永続化をサポートします。

viii. **質問** : これは、オンプレミスに構成された IMM ポッドに保存されたデータが暗号化されていないことを意味しますか ?

回答: オンプレミスノードに保存されたデータは暗号化されていません。

ix. **質問**:ポッドのアクセスを制限するために、ポッドセキュリティポリシー (PSP) としてどのようなセキュリティポリシーを使用していますか ? 例えば、ルートとしてのコード実行を防ぐことや、ボリューム制限などです。
回答: このためには、クラスター レベルで RBAC を適用する必要があります。

x. **質問** : これは、設計上/デフォルトでポッドのアクセス (ルートアクセスおよびボリューム制限) に制限がないことを意味しますか ?

回答: IMM がデプロイされている K8s クラスタのセキュリティ保護は、IMM デプロイの範囲外です。これは、クラスタを作成したチームによってインフラストラクチャ レベルで実行する必要があります。

xi. **質問**: プロジェクトがまったく実行されていないのに、実行中のポッドのステータスが「evicted」と表示されるのはなぜですか?

回答: この問題を回避するには、静的 RAM Ubuntu マシンを使用します。

xii. **質問**: IMM が頻繁にクラッシュする問題を修正するにはどうすればよいですか?

回答: Microk8s で Grafana ダッシュボードを有効にする方法を参照してください。

xiii. **質問**: MongoDB の RAM 消費の問題を克服するにはどうすればよいですか?

回答: MongoDB は、読み取り/書き込み操作を効率的に実行できるキャッシュ ストレージを提供します。ただし、実行する R/W 操作の勢いが強い場合、LOGDB ポッドは大量の RAM を消費します。この問題を克服するために、WIRE_TIGER_CACHE_GB という構成可能なパラメータが導入されました。このパラメータを設定すると、キャッシュ サイズを制限できます。 このパラメータは、MongoDB で使用されるキャッシュ ストレージの最大量 (GB 単位) を設定するために使用されます。キャッシュ サイズを ½ GB にしたい場合は、WIRE_TIGER_CACHE_GB の値を 0.5 に設定してください。

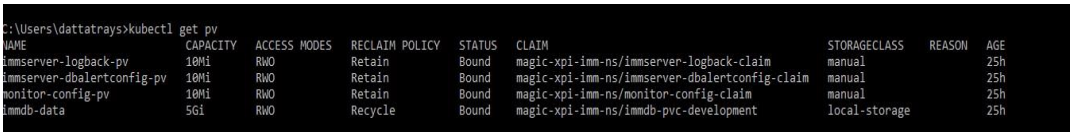
Kubernetes コマンドリスト

1. トラブルシューティングの目的でポッドを削除するには、次のコマンドを実行します。Kubernetes は削除されたポッドを自動的に再起動します。

```
kubectl delete pod <pod-name> -n magic-xpi-imm-ns
```

2. ストレージのプロビジョニングと管理に関連する問題をトラブルシューティングするには、次のコマンドを実行します。:

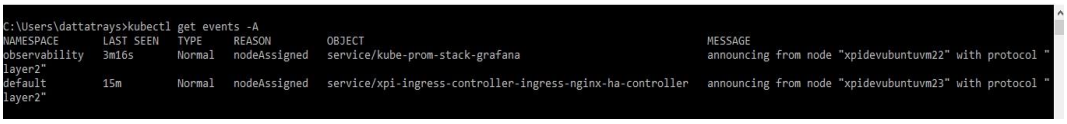
```
kubectl get pv
```



```
C:\Users\dattatrays>kubectl get pv
NAME                        CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM                                          STORAGECLASS    REASON   AGE
immserver-logback-pv        10Mi       RWO            Retain           Bound    magic-xpi-imm-ns/immserver-logback-claim       manual                   25h
immserver-dbalertconfig-pv  10Mi       RWO            Retain           Bound    magic-xpi-imm-ns/immserver-dbalertconfig-claim manual                   25h
monitor-config-pv           10Mi       RWO            Retain           Bound    magic-xpi-imm-ns/monitor-config-claim          manual                   25h
immdb-data                  5Gi        RWO            Recycle          Bound    magic-xpi-imm-ns/immdb-pvc-development         local-storage            25h
```

3. Kubernetes 環境の全体的な状態を理解するには、次のコマンドを実行します。:

```
kubectl get events –A
```



```
C:\Users\dattatrays>kubectl get events -A
NAMESPACE       LAST SEEN   TYPE     REASON         OBJECT                                                  MESSAGE
observability   3m16s       Normal   nodeAssigned   service/kube-prom-stack-grafana                         announcing from node "xpidevubuntuvm22" with protocol "
layer2"
default         15m         Normal   nodeAssigned   service/xpi-ingress-controller-ingress-nginx-ha-controller   announcing from node "xpidevubuntuvm23" with protocol "
layer2"
```

4. MicroK8s クラスタをクリーンな状態に戻すには、次のコマンドを実行します。:

```
sudo microk8s reset
```

Microk8s をリセットしたら、以下のコマンドを使用してコンテナのランタイム構成ファイルでプロキシ設定を設定します。Microk8s のリセットには数分かかる場合があります。

```
nano /var/snap/microk8s/current/args/containerd-env
```

プロキシ設定が完了したら、次のコマンドを実行して microk8s を再起動します:

```
sudo snap restart microk8s
```

5. システム ポッドがステータス CrashloopBackoff でクラッシュした場合、microk8s を停止するには、次のコマンドを実行します。:

```
sudo microk8s stop
```

6. システムから microk8s を削除するには、次のコマンドを実行します。:

```
sudo snap remove microk8s --purge
```

7. imm をアンデプロイした後、imm ネームスペースが終了状態のままになっている場合は、次のコマンドを実行します。:

```
NS=`microk8s kubectl get ns |grep Terminating | awk 'NR==1
{print $1}'` && microk8s kubectl get namespace "$NS" -o json
| tr -d "\n" | sed
"s/\"finalizers\":\[[^]]\+\]/\"finalizers\": []/"  | microk8s
 kubectl replace  --raw /api/v1/namespaces/$NS/finalize -f -
```

8. kubectl コマンドの実行中にセキュリティ証明書関連のエラーが見つかった場合は、次のコマンドを実行して問題を修正します。:

```
sudo microk8s refresh-certs -e ca.crt
```

9. デプロイされたイメージのバージョンを確認するには:

```
kubectl get pods -A
kubectl get pod <pod-name> -n <namespace> -o=jsonpath=
                                  '{.spec.containers[*].image}'
```

10. IMM が正常にデプロイされたかどうかを確認するには、以下のコマンドを実行します。:

```
kubectl get pods -n magic-xpi-imm-ns
```

OR

```
kubectl get pods -n magic-xpi-imm-ns –watch
```

```
C:\Users\dattatrays>kubectl get pods -n magic-xpi-imm-ns --watch
NAME                                     READY   STATUS    RESTARTS   AGE
imm-db-0                                 1/1     Running   0          16h
logdb-5d4f8f99f8-z791t                   1/1     Running   0          16h
imm-tunnel-deployment-6b784d449f-zk25j   1/1     Running   0          16h
xpi-monitor-6894b75bbd-729tg             1/1     Running   0          16h
imm-controller-76c4bc9db5-kg5k8          1/1     Running   0          16h
xpi-imm-server-deployment-6dd5bf78fb-vkj7f   1/1     Running   0          16h
```

このコマンドは、magic-xpi-imm-ns ネームスペースで実行されているすべてのポッドのリストを取得します。各ポッドの名前、ステータス、再起動回数などの基本情報が表示されます。

```
kubectl get pods -n magic-xpi-imm-ns -o wide
```

このコマンドは、magic-xpi-imm-ns ネームスペースのポッドに関する情報を取得しますが、追加の詳細はワイド形式で提供されます。ワイド形式は、スケジュール情報やノード割り当てなど、ポッドの包括的なビューを取得するのに役立ちます。

Magic Software Enterprises について

Magic Software Enterprises (NASDAQ: MGIC) empowers customers and partners around the globe with smarter technology that provides a multi-channel user experience of enterprise logic and data.

We draw on 30 years of experience, millions of installations worldwide, and strategic alliances with global IT leaders, including IBM, Microsoft, Oracle, Salesforce.com, and SAP, to enable our customers to seamlessly adopt new technologies and maximize business opportunities.

For more information, visit www.magicsoftware.com.