ORACLE

Oracle RAC on Dockerのデプロイメントのベスト・プラクティス

Oracle RAC on Dockerのデプロイメントと構成

2021年11月 | バージョン1.3

本書の目的

本書では、DevOps（開発と運用）が現在直面している多くの課題に対するソリューションとして、Oracle RAC on Dockerをデプロイおよび構成するためのベスト・プラクティスを示します。

対象読者

Oracle Database管理者とアプリケーション開発者。

免責事項

本文書には、ソフトウェアや印刷物など、いかなる形式のものも含め、オラクルの独占的な所有物である占有情報が含まれます。この機密文書へのアクセスと使用は、締結および遵守に同意したOracle Software License and Service Agreementの諸条件に従うものとします。本文書と本文書に含まれる情報は、オラクルの事前の書面による同意なしに、公開、複製、再作成、またはオラクルの外部に配布することはできません。本書は、ユーザーとのライセンス同意書の一部をなすものではなく、またオラクルやその子会社および関連会社とのいかなる契約上の合意事項にも含まれるものではありません。

本書は情報提供のみを目的としたものであり、ここで説明する製品の機能を実装およびアップグレードする際の資料として使用されることのみを意図しています。マテリアルやコード、機能の提供をコミットメント（確約）するものではなく、購買を決定する際の判断材料になさらないでください。本書に記載されている機能の開発、リリース、および時期については、弊社の裁量により決定されます。

製品アーキテクチャの性質上、本書に記述されているすべての機能を安全に組み込むことができず、コードの不安定化という深刻なリスクを伴う場合があります。

目次

はじめに

Oracle Real Application Clusters（Oracle RAC）を使用すると、あらゆるパッケージ・アプリケーションやカスタム・アプリケーションを、クラスタ化されたサーバー全体にわたってOracle Database上で実行できます。これにより、ノードとインスタンスで障害が発生した場合や、大抵の計画メンテナンスを実行している間に、またOracle RACを拡張する場合でも、データベースのサービスを継続できます。クラスタ構成のノードの1つが停止した場合には、動作を続けているいずれかのノードでOracle Databaseサービスが実行され続けます。処理能力を増やす必要が生じた場合には、データベースやデータへのユーザー・アクセスを中断せずに別のノードを追加できます。

Dockerは、マイクロサービスの開発とデプロイメントを加速する、オープンソースのコンテナ化プラットフォームです。Dockerは、その俊敏性のために、ITコミュニティから多くの支持を得ています。このため、多くの組織がDockerコンテナでサービスを実行しています。

- **統合の課題：**アプリケーション用のデータベース環境を構築するには、多数存在するソフトウェア依存関係を解決する必要があります。Dockerを使用すると、開発者またはデータベース管理者は、すべてのパッケージの依存関係、Oracle Grid Infrastructureソフトウェア、Oracle Databaseのリリース更新、アプリケーション・デプロイメント・スクリプトを使用してOracle RAC on Dockerイメージを構築できます。

- **環境プロビジョニングの課題：**テスト環境および開発環境では、ユーザーには迅速な環境プロビジョニングが必要です。パッケージ化されたデータベースのDockerイメージがなければ、多くの時間とスキルが費やされます。しかし、Dockerを使用すると、ユーザーは、Oracle RACイメージをダウンロードし、Oracle RACとアプリケーション・スキーマのデプロイメントをトリガーできます。ユーザーは、このイメージに含まれる完全なソフトウェア・スタックを取得し、環境を迅速にプロビジョニングできます。

- **テストの課題：**Oracle RACはDockerコンテナ内で実行されるため、ユーザーは、テスト環境をフリーズし、その他のテスト担当者や開発者と共有することで、コラボレーションと生産性を強化できます。

ソリューションの概要

このセクションでは、Oracle Grid Infrastructure、Oracle RAC、Oracle Linux、Dockerについて説明します。

Oracle Clusterware

Oracle Clusterwareを使用すると、サーバー同士が連携することで、可用性の高い1つの集合体として機能するように見えます。このようなサーバーの連結は一般に、クラスタとして知られています。サーバーはそれぞれスタンドアロン・サーバーですが、各サーバーが他のサーバーと通信するため、アプリケーションやエンドユーザーからは、別々のサーバーが1つのシステムのように見えます。また、Oracle Clusterwareは、クラスタ内のコンポーネントを監視し、リソースを再起動またはフェイルオーバーすることで、高可用性を確保します。

Oracle Clusterwareは、Oracle RACの実行に必要なインフラストラクチャを提供します。また、Oracle Clusterwareは、仮想IP（VIP）アドレス、データベース、リスナー、サービスなど多くのリソースも管理します。

Oracle Automatic Storage Management（Oracle ASM）

Oracle Automatic Storage Managementは、Oracle RACとシングル・インスタンスのOracle Databaseの双方で使用できる推奨のクラスタ・ボリューム・マネージャです。Oracle ASMでは、"Stripe And Mirror Everything"（SAME）の原則を基にストレージ管理を簡素化します。インテリジェントなミラー化機能によって、管理者は2方向または3方向のミラー化を定義して、重要なデータを保護できます。読取り処理でディスクの破損ブロックが識別されると、Oracle ASMは、有効なブロックをミラー・コピーからディスクの破損していない部分に自動的に再配置します。

Oracle Real Application Clusters（Oracle RAC）

Oracle Real Application Clustersを使用すると、相互接続された複数のインスタンスで連携してOracle Databaseサービスを提供できます。Oracle RAC環境の場合、Oracle Databaseは、エンドユーザーとアプリケーションからは単一データベースのように見えながら、クラスタ内の複数のシステムで稼働します。その結果、1つのデータベースが複数のハードウェア・システムで稼働することになり、Oracle RACではクラスタ内で障害が発生している間も高可用性と冗長性を維持することができます。Oracle RACは、読取り専用のデータウェアハウス・システムから、更新を集中的に行うオンライン・トランザクション処理（OLTP）システムまで、すべてのシステム・タイプに対応しています。

Oracle Linux

オペレーティング・システム、コンテナ、仮想化は、最新のITインフラストラクチャの基盤となる構成要素です。オラクルでは、これらすべてを組み合わせて、統合された1つの製品であるOracle Linuxを構築しています。データセンターまたはクラウド内で選り抜きのハードウェア上で動作するOracle Linuxは、要求の厳しいエンタープライズ・ワークロードに対して信頼性、スケーラビリティ、セキュリティ、パフォーマンスを提供しています。

Oracle Container Runtime for Docker

Oracle Container Runtime for Dockerを使用すると、Dockerをサポートしているoracle Linuxシステムやその他のオペレーティング・システム全体にわたってアプリケーションを簡単に作成して配布できます。Oracle Container Runtime for Dockerは、アプリケーションをパッケージ化して実行するDocker Engineによって構成されています。これは、Docker Hub、Docker Store、Oracle Container Registryと統合され、Software-as-a-Service（SaaS）クラウド内のアプリケーションを共有します。

デプロイメントのベスト・プラクティス

このセクションでは、Oracle RAC on Dockerのデプロイと構成のベスト・プラクティスの詳細について説明します。

手順1：Oracle Linux 7のインストールと構成

Oracle RAC on Dockerを設定する場合、Oracle Linux 7.xと一緒にUEK4を使用します。インストール手順とシステム要件の詳細については、Oracle Linuxドキュメントにある『Installation Guide for Oracle Linux Release 7』を参照してください。

手順2：Docker Engineのインストールと構成

Oracle RAC on Dockerは、非特権モード機能をサポートしています。非特権モードで動作することで、Oracle RACは、他のDockerコンテナから干渉されることなく、単一ホストまたは複数ホスト上で安全かつセキュアに動作できるようになります。Oracle Container Runtime for Dockerの構成には、多くの手順が含まれます。これらの手順は、ドキュメントで指定されている順序で実行してください。以下に説明する各セクションの詳細については、Oracle Container Runtime for Docker User's Guideを参照してください。

手順3：Oracle RAC on Dockerのデプロイメント・シナリオ

パブリック・アクセス可能なIPアドレスを使用した単一ホストでのデプロイメント

単一のDockerホストに複数のOracle RACクラスタをデプロイできます。すべてのOracle RACクラスタが、Dockerコンテナ内で動作する専用ストレージと割当て済みIPを使用します。別のホスト上で動作できるアプリケーションからスキャンIPを直接使用して、単一ホスト上で動作しているOracle RACデータベースに接続できます。DockerコンテナIPを物理ネットワークに公開するには、Docker MACVLANブリッジが必要です。

コンテナ専用IPアドレスを使用した単一ホストでのデプロイメント

コンテナの外部からスキャンIPにアクセスできない場合、Oracle Connection Managerを使用してアプリケーションからOracle RACクラスタにアクセスできます。Oracle Connection Managerは、データベース・プロキシ・サーバーのように動作し、一般公開されている単一のIPアドレスを提供することで、複数のコンテナ・プライベートIPアドレスに接続できるようにします。図1は、Oracle Connection Managerを使用して単一ホストでOracle RAC on Dockerを実行するためのアーキテクチャを示しています。

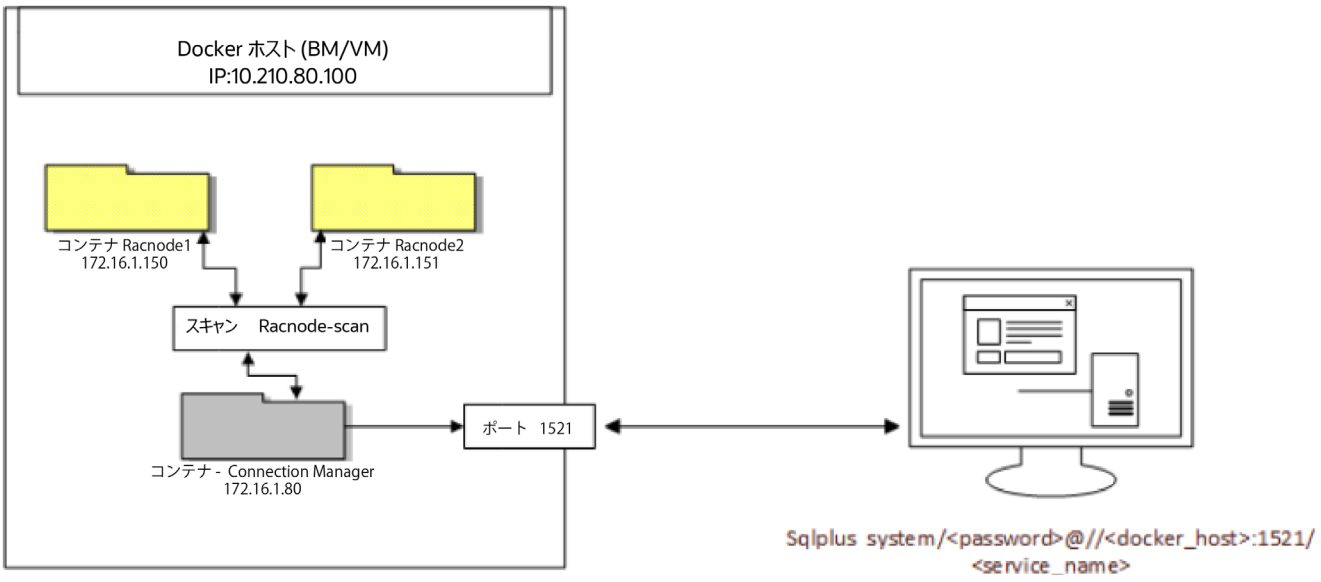

図1：Oracle Connection Managerを使用したOracle RAC on Dockerのデプロイメント・シナリオ

複数ホストでのデプロイメント

Docker上の複数ホストに複数のOracle RACクラスタをデプロイできます。すべてのOracle RACクラスタが、Dockerコンテナ内の専用ストレージと割当て済みIPアドレスを使用します。複数ホストでOracle RAC on Dockerを実行するには、同じサブネット上にコンテナIPが存在し、異なるホスト上で動作しているコンテナ間でこれらに到達できる必要があります。Docker MACVLANブリッジは、コンテナをパブリック物理ネットワークに直接接続し、異なるホスト上で動作しているコンテナが相互通信できるようにします。

図2は、複数ホストでOracle RAC on Dockerを実行するためのアーキテクチャを示しています。

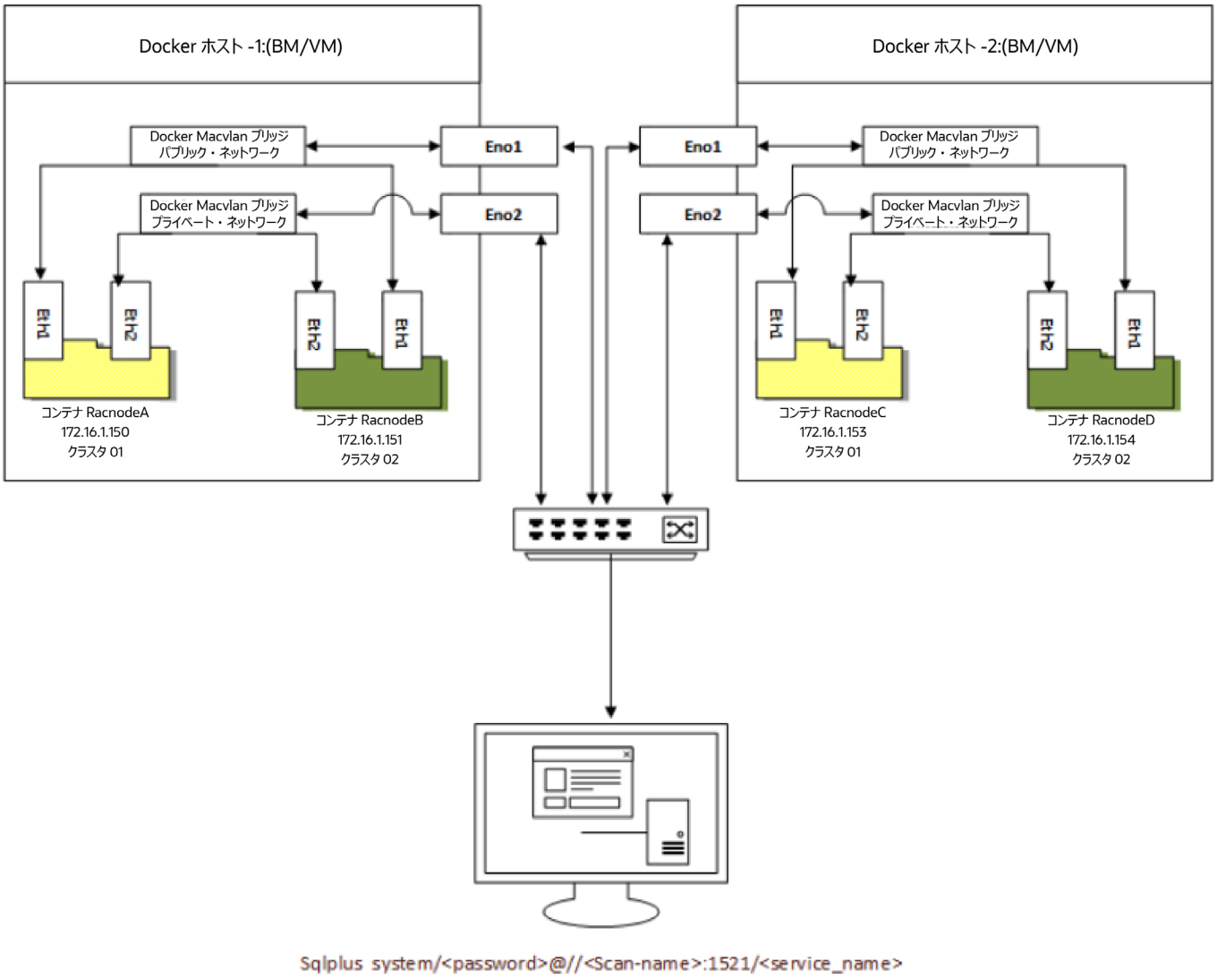

図2：複数ホストでDocker Macvlanドライバを使用してユーザー・ネットワーク上にOracle RACがデプロイされている場合のOracle RAC on Docker

ブロック・ストレージを使用したデプロイメント

Oracle ClusterwareとOracle RACデータベースは両方とも、クラスタ内のすべてのノードで使用できる必要があるファイルを使用します。Oracle ASM用として直接接続ストレージ（DAS）またはストレージ・エリア・ネットワーク（SAN）を使用する場合、ディスクごとにパーティション表が必要です。ディスク全体を網羅するパーティションをディスクごとに厳密に1つ作成することを推奨します。

詳細については、Oracle Grid Infrastructureインストレーションおよびアップグレード・ガイドfor Linuxの「Oracle Grid Infrastructureでサポートされている記憶域オプション」セクションを参照してください。

"--privileged=true"オプションを"docker create"コマンドまたは"docker run"コマンドに指定する場合、コンテナは、ホスト上のすべてのデバイスにアクセスできます。これは、セキュリティ上のリスクをもたらす可能性があります。Oracle RACをインストールする前にストレージ要件を計画し、Oracle RAC DBストレージ要件に基づいてコンテナに特定のブロック・デバイスを割り当て、より正確に制御する必要があります。

特定のコンテナで使用できるホスト上で特定のブロック・デバイスを割り当てるには、docker runコマンドおよびdocker createコマンドと一緒に"ーdevice"オプションを使用します。--device=host_devname[:container_devname[:permissions]]

図3は、ブロック・デバイスを共有ストレージとして使用してOracle RAC on Dockerをデプロイするためのアーキテクチャを示しています。

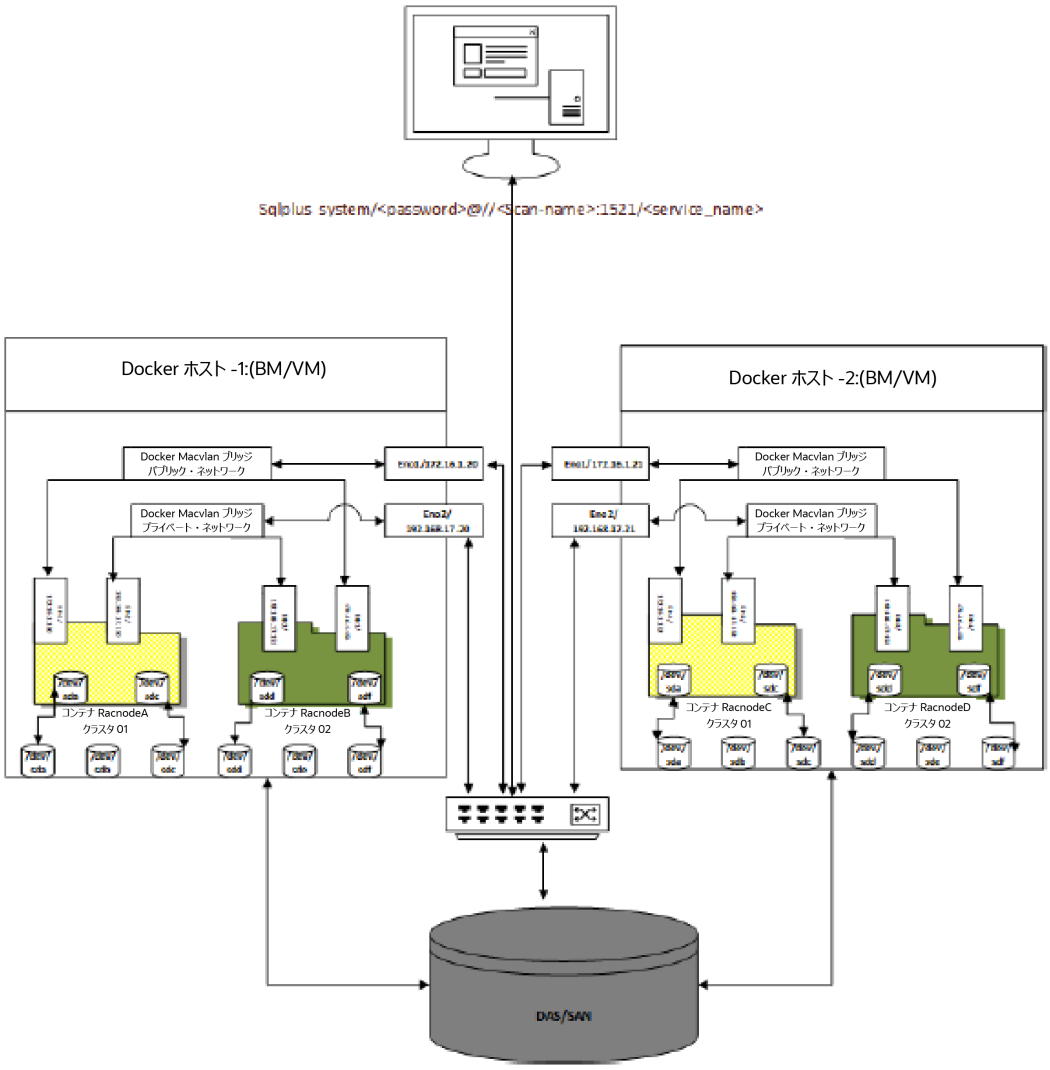

図3：ブロック・ストレージを共有ストレージとして使用したOracle RAC on Docker

NASストレージを使用したデプロイメント

認定済みNASストレージ・デバイスがある場合、NFSマウント・ディレクトリ内にゼロ埋めされたファイルを作成し、これらのファイルをOracle ASMディスク・グループ用のディスク・デバイスとして使用できます。この場合、ゼロ埋めされた1 GBのサイズのファイルを作成するために、ddコマンドを使用します。

例：dd if=/dev/zero of=/oradata/asm_disk01.img bs=1M count=1000

詳細については、Oracle Grid Infrastructureインストレーションおよびアップグレード・ガイドfor Linuxの「Oracle Grid Infrastructureでサポートされている記憶域オプション」セクションと「NASデバイスでのOracle Automatic Storage Management用のファイルの作成」セクションを参照してください。

Oracle RAC on Dockerを実行するためのブロック・デバイスまたはNASデバイスがないとします。この場合、Oracle RACコンテナに公開されるNFSサーバーとNFSボリュームをエミュレートするOracleRACStorageServerイメージをデプロイできます。次の図は、Oracle RAC on DockerをNASデバイスと一緒に使用する事例を説明しています。

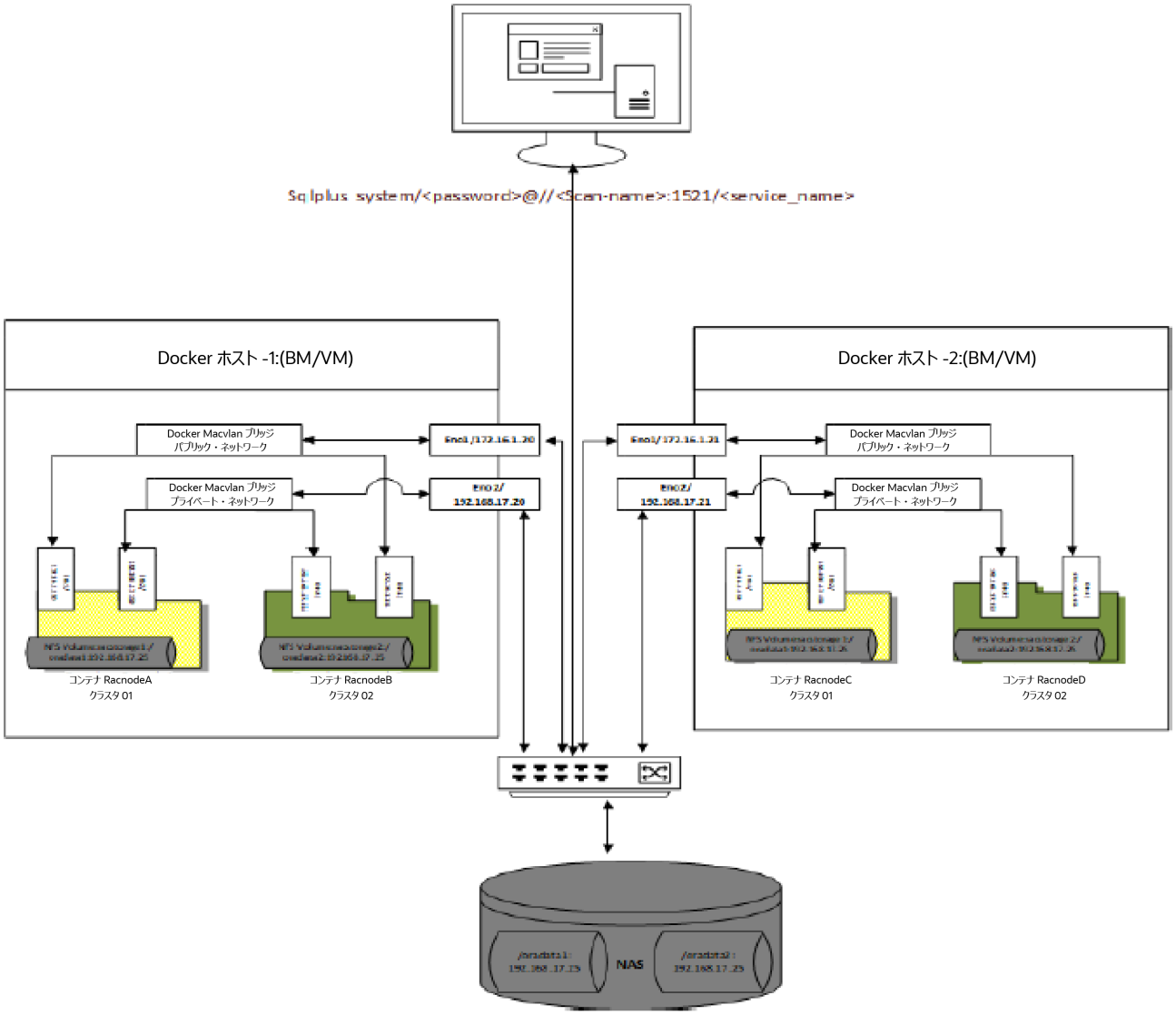

図4：NASを共有ストレージ・デバイスとして使用したデプロイメント

注：OracleRACStorageServerストレージ・イメージは、テスト・デプロイメントおよび開発デプロイメント用としてのみサポートされています。

手順4：Dockerリポジトリ・ファイルとOracleソフトウェアのダウンロード

デプロイメントの概要

Oracle RAC on Dockerイメージを構築するには、GitHubからOracle/Dockerイメージ・リポジトリ・ファイルをクローニングし、環境に基づいてイメージを構築します。Oracle RAC Dockerイメージは、Oraclelinux:Slim-7イメージに基づいて構築されます。図5は、Oracle RAC Dockerイメージ、Oracle Connection Managerイメージ、Oracle RACストレージ・イメージ用の完全な構築シナリオを示しています。

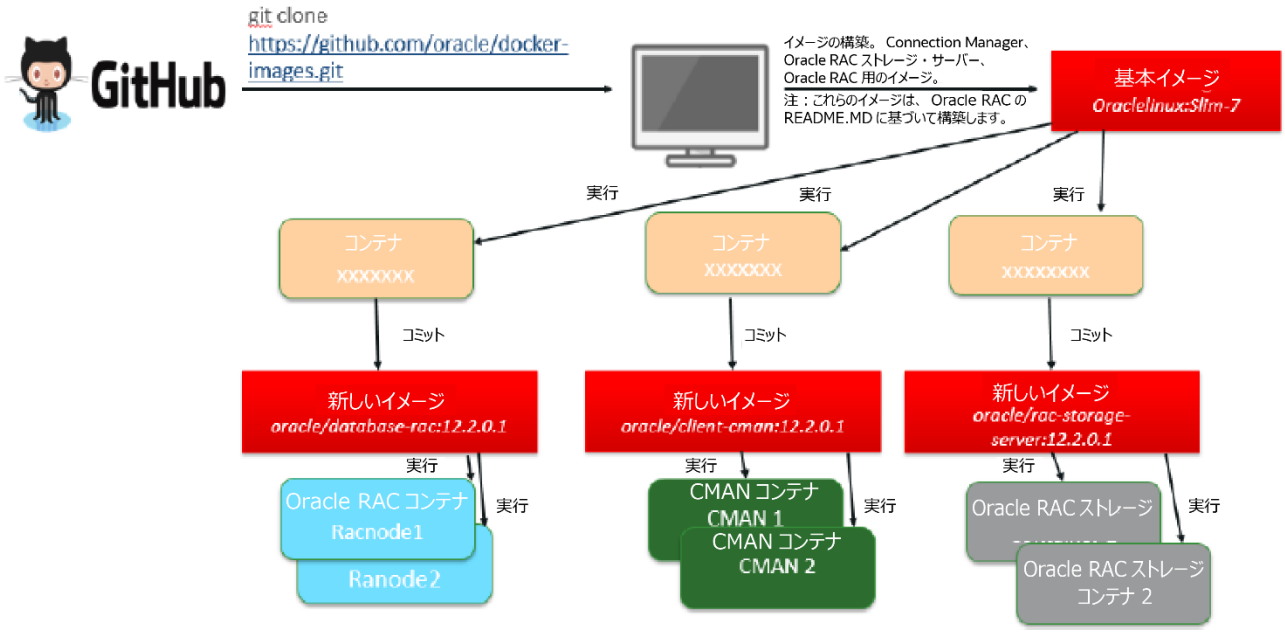

図5：GitHubからOracle/Dockerリポジトリ・ファイルをクローニングする手順

GithubからのOracle RAC on Dockerファイルのクローニング

GitHubからOracle/Dockerイメージ・リポジトリ・ファイルをクローニングしてイメージを構築するには、次のコマンドを使用します。

git clone https://github.com/oracle/docker-images.git

ファイルをダウンロードするには、サーバーからGitHubに接続できる必要があります。サーバーからGitHubに到達できない場合は、環境に基づいてプロキシを設定してGitHubからファイルをダウンロードしてみることができます。詳細については、Oracle Container Runtime for Docker User's Guideを参照してください。

Oracleソフトウェアのダウンロード

Oracle Dockerリポジトリ・ファイルには、Oracleソフトウェアのバイナリは含まれていません。ソフトウェアをOracle Technology Networkからダウンロードし、dockerfiles/<version>フォルダに配置します。詳細については、OracleRealApplicationClusters DockerファイルのREADME.md内の"Pre-requisites for RAC on Docker"セクションを参照してください。

手順5：Dockerイメージの構築

Oracle RAC on Dockerリポジトリ・ファイルとOracleバイナリをサーバーにダウンロードしたら、環境に基づいてイメージを構築します。インフラストラクチャの設定によっては、次の3つのイメージを構築することが必要な場合があります。

1. Oracle Connection Managerイメージ
   a. Oracle RACデータベースにアクセスしたいが、物理ネットワークから到達できるOracle RACのIPアドレスがない場合に必要です。
2. Oracle RACストレージ・サーバー・イメージ
   a. ブロック・デバイスまたはNASデバイスがない場合に必要です。
   b. OracleRACStorageServerは、テストのみを目的としており、Oracle RACファイルを保存する方法としては推奨されません。
3. Oracle RACイメージ
   a. Oracle RAC DBを実行するために必要です。
   b. Oracle RAC on Dockerのプロビジョニング手順の詳細を理解するには、GitHubのREADME.MDを参照してください。

手順6：Dockerホスト環境の構成

必要なイメージを構築した後、Oracle RAC on Dockerを実行するためのDockerホスト環境を構成します。カーネル・パラメータ、ネットワーク設定、ASMデバイス割当ての構成、およびコンテナでのリアルタイム・プロセス設定の構成が必要な場合があります。

Oracle RAC on Dockerの事前設定手順を理解するには、GitHubのREADME.MD内の"Pre-requisites for Oracle RAC on Docker"セクションを参照してください。

手順7：Oracle RAC on Dockerのデプロイメント

イメージを構築し、Oracle RAC on Dockerデプロイメント用のDockerホスト環境を事前設定したら、必要に応じてイメージから次のコンテナを作成します。

1. Oracle Connection Managerイメージ
   a. Oracle Connection Managerのデプロイメント手順の詳細を理解するには、GitHubのREADME.MDの"Creating the Docker GI and RAC Container"セクションを参照してください。
2. Oracle RACストレージ・サーバー・イメージ
   a. GitHubのREADME to understand the RAC Storage Server deployment steps in detail.MDの"Running RACStorageServer Docker container"セクションを参照してください。
3. Oracle RACイメージ
   a. Oracle RAC on Dockerのデプロイメント手順の詳細を理解するには、GitHubのREADME.MDを参照してください。

この時点で、**単一コンテナで動作する単一ノードのOracle RACデータベース**がデプロイされています。テスト環境と開発環境では、単一ノードのOracle RAC環境で十分である場合があります。しかし、複数インスタンスのOracle RACデータベースを実行したい場合は、Oracle RACクラスタに別のノードを追加する必要があります。別のノードを追加するには、手順8に従ってください。それ以外の場合は、手順9にスキップしてください。

手順8：addnodeスクリプトの実行

Oracle RACには、ワークロード要件を満たすために、水平方向に拡張できるaddnode.shスクリプトが用意されています。サーバー上のワークロードに対応するためにクラスタを拡張する必要がある場合は、Oracle RACクラスタにノードを追加します。Oracle RAC on Dockerへのノードの追加は、環境変数として渡されるパラメータを使用して自動化されます。

このホワイトペーパーの手順6で説明されている事前設定手順を実行したことを確認してください。ノードを追加する手順の詳細については、README.MDの"Adding a RAC Node using a Docker container"セクションを参照してください。

手順9：Oracle RACへの接続

Oracle RACデータベース環境がDockerコンテナ内で稼働したら、アプリケーションをOracle RACデータベースに接続できます。Oracle Connection Managerを使用することを選択し、Dockerホスト上のポート1521をパブリック・ネットワークに公開したら、"docker hostname"とポート1521を使用してホストの外部のクライアントから接続できます。Docker Macvlan構成を使用した場合、SCAN名を使用してOracle RACコンテナに直接接続することもできます。

GitHubのOracle RAC on Dockerに関するREADME for more details.MD内の"Connecting to RAC Database"セクションを参照してください。

WebLogicとOracle RAC on Dockerコンテナ

Oracle WebLogic（Oracle WLS）は、多層分散エンタープライズ・アプリケーションを開発およびデプロイするための、スケーラブルなエンタープライズ対応のJ2EEベースのアプリケーション・サーバーです。WebLogic Serverには、アプリケーションを簡単に管理するためのエンタープライズ・レベルのセキュリティ・ツールと管理ツールが用意されています。

WebLogicとOracle RACデータベースとの緊密な統合により、可用性の向上、より優れたリソース共有、構成の容易さ、自動化された管理機能を活用してアプリケーションを開発およびテストするための、堅牢性、完全性、高可用性を備えたインフラストラクチャを実現します。WebLogicとOracle RACデータベースはコンテナ内で現在動作しているため、開発者と企業は、Dockerのコンテナの独立性、移植性、能力の利点を活用して、これらのアプリケーションの開発とテストを自動化できます。WebLogic on Dockerのデプロイメント・シナリオを理解するには、OTNでWebLogicによって公開されているDockerコンテナ上のOracle WebLogic Serverに関するドキュメントを参照してください。

MedRec（Avitek Medical Recordsアプリケーション）は、WebLogic Serverに付属しているエンド・ツー・エンドのサンプルのJava EEアプリケーションで、独立した一元的な医療記録管理システムをシミュレートします。MedRecアプリケーションは、さまざまなクライアントを使用して患者データを管理するためのフレームワークを患者、医師、管理者に提供します。MedRecは、WebLogic ServerとJava EEの機能を実演し、オラクルが推奨するベスト・プラクティスを際立たせます。MedRecは、WebLogic Serverディストリビューションにインストールされます。MedRecは、ここ10年以上にわたって信頼できる現実的な、アプリケーションのパフォーマンス・ベンチマークとして使用されてきました。

図6は、Oracle RAC on Dockerを使用したWebLogicデプロイメントのシステム・アーキテクチャを示しています。図6は、Oracle RACデータベースに接続された、MedRecアプリケーションを実行しているWebLogicコンテナを示しています。いずれかのインスタンスが停止しても、このアプリケーションは引き続きその他のインスタンスに接続できます。

管理サーバーによって管理されている異なるコンテナ内の複数のドメインを実行することで、WebLogicドメインをクラスタ化されたドメインに拡張できます。クラスタ化されたWebLogicコンテナは、同じホスト上または複数のホスト上で動作できます。

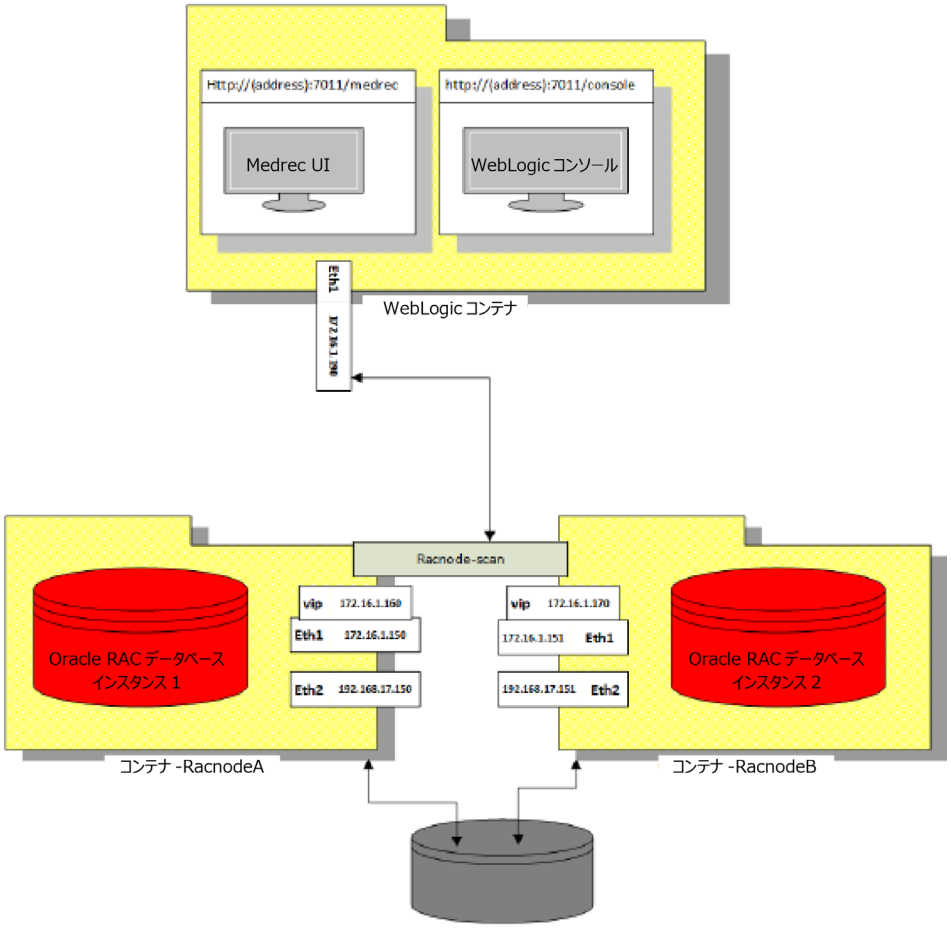

図6：Docker/コンテナ上で動作するWebLogicを使用したOracle RAC

手順1：Oracle RAC on Dockerのインストールと構成

DockerにWebLogicをインストールするには、最初にOracle RAC on Dockerをインストールします。

手順2：WebLogicのインストールと構成

Oracle WebLogicイメージがまだダウンロードされていない場合は、GitHubからダウンロードします。ディレクトリを<DOCKER_IMAGES>/OracleWebLogic/dockerfiles/12.2.1に変更します。

手順3：Oracle WebLogic Serverソフトウェアのダウンロード

Oracle WebLogic Server 12.2.1.2のクイック・インストーラをダウンロードし、<DOCKER_IMAGES>/OracleWebLogic/dockerfiles/12.2.1に配置します。README.mdを参照し、ソフトウェアの正確な詳細を把握し、ダウンロード・リンクを取得します。

手順4：Oracle WebLogicイメージの構築

次のコマンドを実行し、Oracle WebLogic開発者イメージを構築します。

docker build -t oracle/weblogic:12.2.1.2-developer

手順5：MedRecイメージの作成

ディレクトリを<DOCKER_IMAGES/OracleWebLogic/samples/12212-oradb-medrecに変更し、container-scripts/oradatasource.propertiesを編集します。必要に応じ、環境に基づいてパラメータを変更します。Oracle RACデータベースをデフォルト設定で作成した場合、次のパラメータを設定します。

dsurl=jdbc:oracle:thin:@//172.16.1.70:1521/ORCLCDB dsusername=system

dspassword=Oracle_12c

手順6：WebLogicサプリメンタル・インストーラのダウンロード

Oracle WebLogic 12.2.1.2.0サプリメンタル・クイック・ディスク・インストーラをダウンロードし、現在のディレクトリに配置します。詳細については、現在のディレクトリのREADME.MDを参照してください。

手順7：イメージの構築

ソフトウェアが配置されたら、イメージを構築します。次のコマンドを実行し、イメージを構築します。# docker build -t 12212-oradb-medrec

詳細については、現在のディレクトリのREADME.MDを参照してください。

手順8：コンテナの実行

イメージが構築されたら、次のコマンドを実行し、コンテナを作成して実行します。# docker run -d -p 7011:7011 –network rac_pub1_nw

12212-oradb-medrec

注：–network dockerパラメータを渡し、Oracle RAC用として使用したパブリック・ネットワークでWebLogicコンテナを作成します。

手順9：コンソールへのアクセス

次のURLを使用してブラウザからWebLogicコンソールにアクセスできます。http://<Docker_Host>:7011/medrec

まとめ

従来のデプロイメント・ワークフローでは、さまざまな手順が必要であるため、複数のチームがデプロイメント全体に関する苦痛を感じていました。アプリケーションまたはデータベースのデプロイメント・プロセスに追加される各手順（パッチの適用やアプリケーション・スクリプトの提供、ユーザー用の環境の設定など）により、開発/テスト・システムや本番システムのデプロイメントに伴う全体的な生来のリスクが増します。Dockerは、すべての手順を単一のワークフローにまとめることを目的としたシンプルなツールセットを提供することで、これらの懸念に対処します。

Oracle RACは、最新のパッチやリリースを使用することで最適に使用できる、エンタープライズ・クラスのデータベース製品です。Oracle RACをDockerと組み合わせることで、これらの最新のパッチやリリースを使用して高速なプロビジョニングと既製のイメージを提供できます。これらのイメージは、開発環境やテスト環境で簡単に使用し、開発ジョブやテスト・ジョブが完了したときに簡単に破棄できるため、あらゆる組織の開発サイクルやテスト・サイクルを促進できます。

最後に、現在のITが直面しているもっとも重大な課題の1つは、完全なアプリケーション開発ライフサイクルをサポートするために必要なテスト環境、開発環境、QA環境、本番環境を効率的に管理することです。経済的な現実のため、これらの環境を統合型システム上でホストすることで、コスト効率を維持する必要があります。GitHubにOracle RACとDockerリポジトリ・ファイルが導入されたことで、これらすべての課題に対してシンプルなソリューションを使用できるようになりました。これにより、環境のシンプルかつ迅速なデプロイメント、ストレージ、移植性や、統合型環境に必要な独立性が実現されます。

詳細情報

- Oracle RAC on Docker - リリースされたバージョンと既知の問題（Doc ID 2488326.1）
- Oracle® Linux Oracle Container Runtime for Docker User's Guide
  https://docs.oracle.com/en/operating-systems/oracle-linux/docker/
- オペレーティング・システムのドキュメント：https://docs.oracle.com/en/operating-systems/index.html

CONNECT WITH US

+1.800.ORACLE1までご連絡いただくか、oracle.comをご覧ください。
北米以外の地域では、oracle.com/contactで最寄りの営業所をご確認いただけます。

blogs.oracle.com facebook.com/oracle twitter.com/oracle

Oracle RAC on Dockerのデプロイメントのベスト・プラクティス
2021年11月
著者：Sanjay Singh、Paramdeep Saini、Racpackチーム、Bob Thome、Markus Michalewicz
共著者：Avi Miller、Gerald Venzl、Monica Riccelli