コンテナおよび Kubernetes セキュリティへの階層型アプローチ

ビルド、デプロイ、実行までコンテナをセキュリティ保護

目次

はじめに 2

コンテナおよび Kubernetes の包括的なセキュリティ：レイヤーとライフサイクル 2

アプリケーションへのセキュリティの組み込み 4

デプロイ：デプロイの構成、セキュリティ、コンプライアンスの管理 8

実行中のアプリケーションの保護 11

強固なエコシステムによるセキュリティの拡張 15

まとめ 15

fb.com/RedHatJapan
twitter.com/RedHatJapan
linkedin.com/company/red-hat

はじめに

コンテナには幅広い魅力があります。たとえば、コンテナを使用することで、ユーザーはアプリケーションとそのすべての依存関係を単一のイメージにパッケージ化することができます。そしてそのイメージは、開発からテスト、プロダクションへとプロモートすることが可能です。コンテナを使用すると、物理サーバー、仮想マシン (VM)、プライベートクラウドやパブリッククラウドといったさまざまな環境やデプロイ先で、一貫性を保つことが容易になります。コンテナにより、アプリケーションの開発と管理がさらに簡単になり、ビジネスのアジリティを実現できます。

- **アプリケーション：**コンテナにより、開発者がアプリケーションとその依存関係を 1 つのユニットとしてビルドし、提供することが容易になります。コンテナは瞬時にデプロイできます。コンテナ化された環境において、ソフトウェアのビルドプロセスとは、アプリケーション・コードが必要なランタイムライブラリと統合されるライフサイクルの段階です。
- **インフラストラクチャ：**コンテナは共有 Linux® OS カーネル上のサンドボックス・アプリケーション・プロセスを表わします。仮想マシンよりもコンパクトかつ軽量で、複雑性が低く、オンプレミスからパブリッククラウド・プラットフォームまで、各種の環境への可搬性を備えています。

Kubernetes は企業向けのコンテナ・オーケストレーション・プラットフォームです。多数の企業が基本サービスをコンテナ上で実行するようになるとともに、コンテナのセキュリティ維持は一層重要になってきています。本書では、コンテナ化アプリケーションのセキュリティの基本要素を説明します。

コンテナおよび Kubernetes の包括的なセキュリティ：レイヤーとライフサイクル

コンテナのセキュリティ保護は、実行中の Linux プロセスをセキュリティ保護することと似ています。まず、コンテナをデプロイして実行する前に、ソリューション・スタックのレイヤー全体にわたるセキュリティについて考える必要がありますまた、アプリケーションとコンテナのライフサイクル全体におけるセキュリティについても考える必要があります。重要なのは、セキュリティは継続的プロセスであり、やはり IT ライフサイクル全体を通じて統合され、新たな脅威やソリューションが出現したときには対処のために拡張できるようになっている必要があるということです。図 1 は、コンテナセキュリティへの包括的なアプローチを示しています。

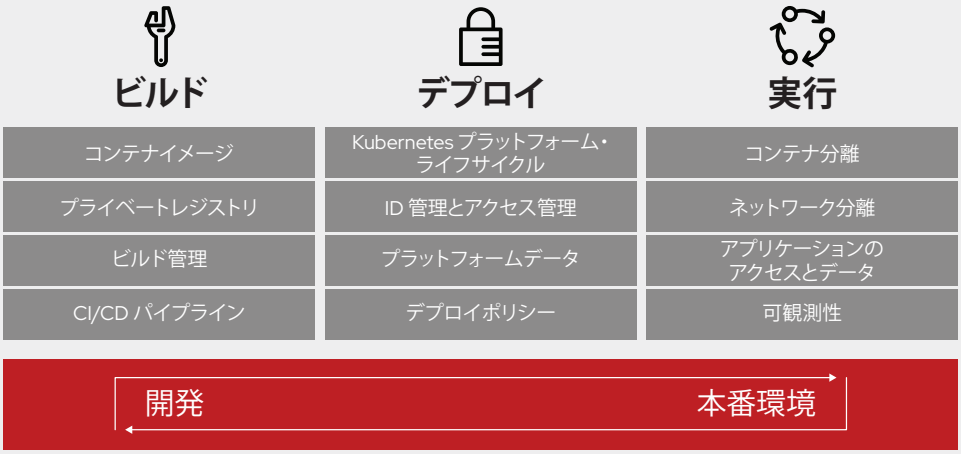

図 1. 包括的なコンテナセキュリティでは、ビルド、デプロイ、ランタイムをセキュリティ保護し、必要に応じて特殊ツールでセキュリティを拡張する

コンテナにより、開発者がアプリケーションとその依存関係を 1 つのユニットとしてビルドし、提供することが容易になります。また、コンテナは、共有ホストへのマルチテナント・アプリケーションのデプロイを可能にすることで、サーバーを最大限に活用できるようにします。1 つのホストに複数のアプリケーションを簡単にデプロイし、必要に応じて個々のコンテナを起動したりシャットダウンしたりできます。従来の仮想化とは異なり、ハイパーバイザーは必要ありません。各 VM のゲスト・オペレーティングシステムの管理も不要です。コンテナは、ハードウェアではなくアプリケーション・プロセスを仮想化します。

アプリケーションが単一のコンテナで提供されることは稀です。単純なアプリケーションにも、通常はフロントエンド、バックエンド、およびデータベースがあります。また、コンテナ化された最新のマイクロサービス・ベースのアプリケーションをデプロイすることは、複数のコンテナをデプロイすることを意味します。図 2 が示すように、デプロイ先は同じホストである場合も、複数のホストやノードに分散されている場合もあります。

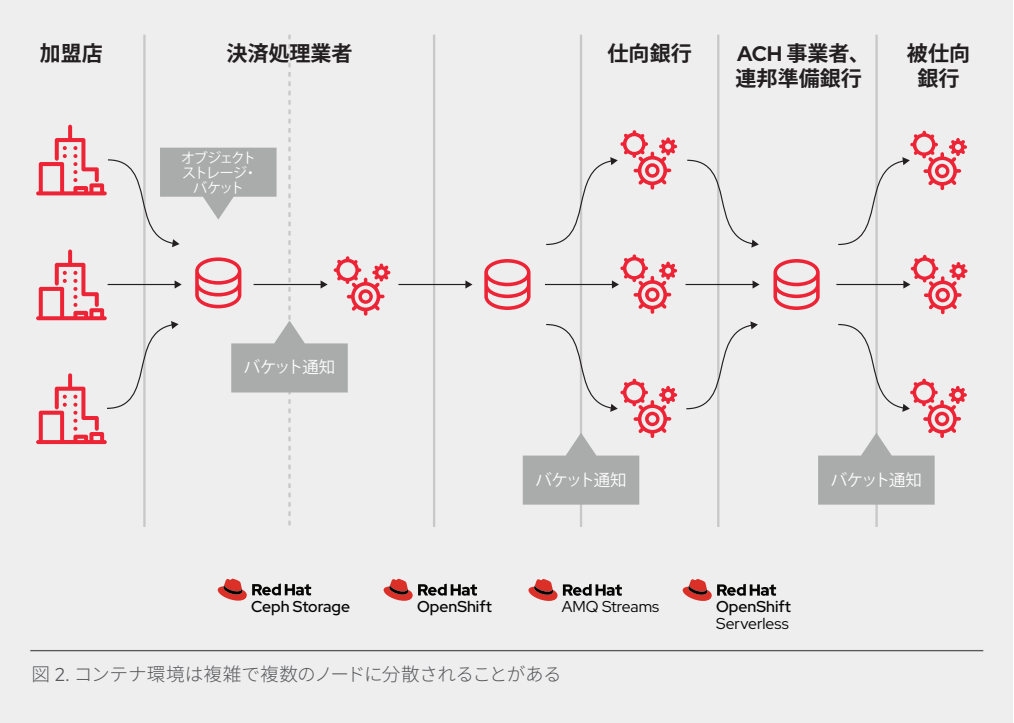

図 2. コンテナ環境は複雑で複数のノードに分散されることがある

大規模環境においてコンテナのデプロイメントを管理する場合、次の点を考慮する必要があります。

- どのホストにどのコンテナをデプロイするか
- どのホストの容量が大きいか
- どのコンテナが互いにアクセスする必要があるか、互いをどのように認識するか
- ネットワークやストレージなどの共有リソースへのアクセスとその管理をどのように制御するか
- コンテナの正常性をどのように監視するか
- 需要に対応するため、アプリケーション容量の自動的なスケーリングをどのように行うか
- セキュリティ要件を満たしながら開発者向けのセルフサービス機能をどのように有効化するか

Red Hat

独自のコンテナ管理環境を構築することはできますが、個々のコンポーネントの統合と管理に長い時間がかかります。または、組み込みの管理機能とセキュリティ機能を使ってコンテナ・プラットフォームをデプロイすることもできます。このアプローチでは、インフラストラクチャの再設計に手間をかけずに、ビジネス価値を実現するアプリケーションの構築に注力できます。

Red Hat® OpenShift® Container Platform は、コンテナ化アプリケーションの構築とスケーリングを行う、一貫したエンタープライズ向けハイブリッドクラウド Kubernetes プラットフォームです。組織全体で Kubernetes を使用するには、セキュリティ機能の増強が必要です。この機能により、セキュリティをアプリケーションに組み込み、コンテナ・デプロイ・セキュリティを管理するポリシーとコンテナランタイムを保護する機能を自動化します。

アプリケーションへのセキュリティの組み込み

アプリケーションへのセキュリティの組み込みは、クラウドネイティブ・デプロイにとってきわめて重要です。コンテナ化アプリケーションをセキュリティ保護するには、以下のことが必要です。

1. 信頼できるコンテナコンテンツを使用する
2. エンタープライズ向けコンテナレジストリを使用する
3. コンテナのビルドを制御し、自動化する
4. セキュリティをアプリケーション・パイプラインに統合する

1. 信頼できるコンテナコンテンツを使用する

セキュリティを管理する際、コンテナの中身が重要です。ある程度の期間、アプリケーションとインフラストラクチャの構成には簡単に利用可能なコンポーネントが利用されてきました。これらの多くは、Linux オペレーティングシステム、Apache Web Server、Red Hat JBoss® Enterprise Application Platform、PostgreSQL、Node.js などのオープンソース・パッケージです。これらのパッケージのコンテナ化されたバージョンも利用できるため、独自のパッケージをビルドする必要はありません。しかし、外部ソースからダウンロードするあらゆるコードと同様に、パッケージの提供元、作成者、そして悪質なコードが含まれていないかを確認する必要があります。以下の事項を確認してください。

- コンテナの内容物は、使用中のインフラストラクチャのセキュリティを脅かさないか？
- アプリケーション・レイヤーに既知の脆弱性が存在しないか？
- ランタイムレイヤーとコンテナの OS レイヤーは最新のものか？
- コンテナの更新頻度はどの程度で、更新されたことをどのような方法で知ることができるか？

Red Hat は長年にわたり、Red Hat Enterprise Linux と Red Hat のポートフォリオ全体において、信頼ある Linux コンテンツをパッケージ化して提供してきました。そして現在は、同様の信頼あるコンテンツを Linux コンテナとしてパッケージ化して提供しています。Red Hat Universal Base Images が登場し、Open Container Initiative (OCI) 準拠の Linux コンテナが実行されるあらゆる場面で、Red Hat コンテナイメージの信頼性、セキュリティ、パフォーマンスが向上しました。つまり、Red Hat Universal Base Image 上でコンテナ化アプリケーションをビルドして、任意のコンテナレジストリに push して共有できることになります。

Red Hat ではさらに、Red Hat Ecosystem Catalog を通じて、さまざまな言語ランタイム、ミドルウェア、データベースといった多数の認定済みイメージを提供しています。Red Hat 認定コンテナは、ベアメタルから VM、クラウドに至るまで、Red Hat Enterprise Linux が動作するあらゆる場所で実行でき、Red Hat およびパートナーによってサポートされます。

Red Hat では、提供するイメージの正常性を継続的に監視します。Container Health Index で各コンテナイメージの「グレード」を公開し、プロダクション環境におけるシステムのニーズを満たすためにコンテナイメージを選定、使用、評価する方法を詳しく説明しています。コンテナのグレードの評価基準の一部には、未適用のセキュリティエラータの古さと、それがコンテナの全コンポーネントに及ぼす影響の度合いが含まれています。ここでは、コンテナの安全性に関する総合的な評価が、セキュリティの専門家にもそれ以外の人にも理解できる形式で提供されます。

Red Hat がセキュリティアップデート (たとえば runc CVE-2019-5736、MDS CVE-2019-11091、VHOST-NET CVE-2019-14835 への修正プログラムなど) をリリースする際には、コンテナイメージも再構築してパブリックレジストリにプッシュします。Red Hat セキュリティ・アドバイザリーは、認定コンテナイメージで新しく発見された問題を警告し、最新のイメージにユーザーを誘導します。それによりユーザーは、そのイメージを使用するアプリケーションを更新することができます。

Red Hat が提供していないコンテンツが必要なときもあるでしょう。他のソースからのコンテナイメージを使用する際には、継続的に更新される脆弱性データベースを利用するコンテナ・スキャン・ツールを使用して、既知の脆弱性に関する最新情報を常に取得することをお勧めします。既知の脆弱性に関するリストは絶えず更新されるため、コンテナイメージをダウンロードした時点でコンテナイメージの中身をチェックし、さらに 、Red Hat が Red Hat コンテナイメージに関して行っているのと同じように、承認されデプロイされたすべてのイメージの脆弱性ステータスを追跡し続ける必要があります。

2. エンタープライズ向けコンテナレジストリを使用してコンテナイメージへのアクセスの安全性を強化する

コンテナを構築する際、ダウンロードしたパブリック・コンテナ・イメージの上にコンテンツレイヤーを重ねることは一般的です。他の種類のバイナリを管理するのと同様に、ダウンロードしたコンテナイメージと社内でビルドしたイメージへのアクセスと提供も管理する必要があります。コンテナイメージの保存をサポートするさまざまなプライベート・レジストリが存在します。プライベート・レジストリ選定の際は、レジストリに保存されているコンテナイメージについて、使用ポリシーを自動化できるものを選ぶことをお勧めします。

Red Hat OpenShift には、コンテナイメージを管理するための基本機能を提供するプライベートレジストリが含まれています。Red Hat OpenShift レジストリではロールベースのアクセス制御 (RBAC) が提供され、誰が特定のコンテナイメージの pull や push を実行可能かを管理できます。また、Red Hat OpenShift は、JFrog's Artifactory や Sonatype Nexus など、すでに使用している他のプライベート・レジストリとの統合もサポートしています。

スタンドアローンのエンタープライズ・レジストリとしては、Red Hat Quay が提供されています。Red Hat Quay には Geo レプリケーションやイメージトリガーのビルドなどの多数のエンタープライズ機能が追加されています。

Clair プロジェクトは、Red Hat Quay セキュリティスキャナーを作動して、Red Hat Quay 内のすべてのイメージから脆弱性を検出する、オープンソースエンジンです。Red Hat OpenShift Container Security Operator と Red Hat Quay と組み合わせて、OpenShift コンソールで、デプロイしたイメージにある既知の脆弱性をクラスタ全体で把握できます。

3. コンテナイメージのビルドを制御し、自動化する

このビルドプロセスの管理が、ソフトウェアスタックを保護するための鍵となります。「一度のビルドで、あらゆる場所にデプロイ」という思想に則れば、ビルドプロセスの成果物と完全に同一のものがプロダクション環境にデプロイされます。コンテナの不変性を維持するためにも重要です。すなわち、実行中のコンテナにパッチを適用するのではなく、再構築して再デプロイするのです。

Red Hat OpenShift には、カスタムイメージのセキュリティを向上させる手段として、外部イベントに基づいてビルドを自動化する多数の機能が含まれています。

- Red Hat Quay トリガーには、GitHub プッシュ、BitBucket プッシュ、GitLab プッシュ、webhook などの外部イベントから Dockerfile のリポジトリ再構築を発生させるメカニズムがあります。
- Source-to-image (S2I) は、ソースコードとベースイメージを結合するための、オープンソースのフレームワークです (図 3)。S2I が提供する再現可能なビルド環境上では、開発チームと運用チームが簡単にコラボレーションすることができます。開発者が git でコードをコミットすると、S2I により Red Hat OpenShift で以下のことが可能になります。
  - 利用可能なアーティファクト (S2I ベースイメージ、新たにコミットされたコードなど) から新しいイメージの自動ビルドをトリガーできる (コードリポジトリ上の webhook または他の自動化された CI プロセスを介して)
  - 新たにビルドされたイメージをテスト用に自動的にデプロイする
  - テストされたイメージをプロダクション・ステータスにプロモーションさせ、継続的インテグレーション/継続的デプロイ (CI) プロセスを通じて新しいイメージを自動的にデプロイする

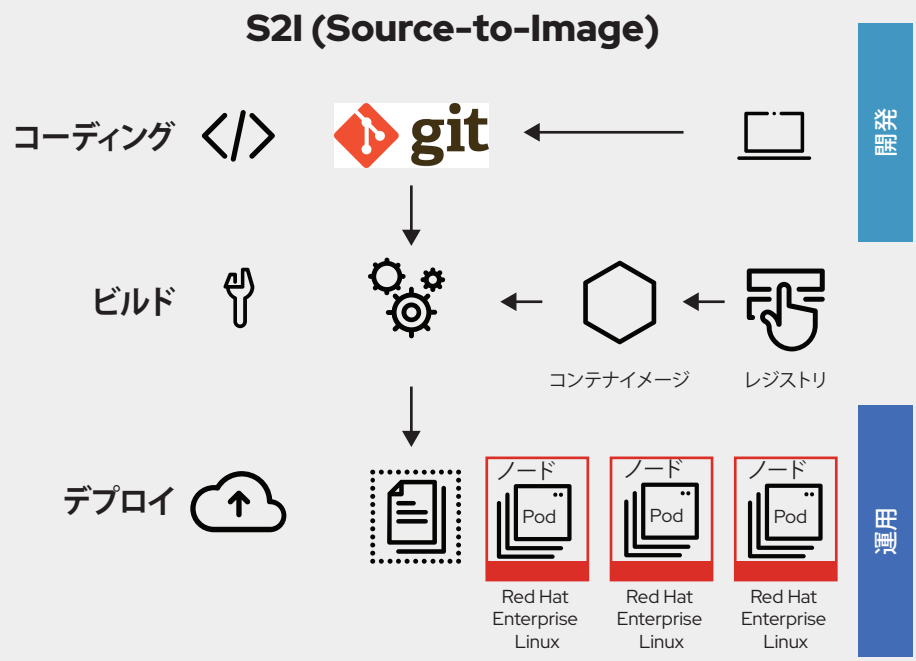

図 3. S2I は、再生可能で実行可能なコンテナイメージをソースコードから構築するツールキットおよびワークフロー

- Red Hat OpenShift イメージストリームを使用して、クラスタにデプロイされた外部イメージへの変更を監視できます。イメージストリームは、ビルドやデプロイ、ジョブ、レプリケーション・コントローラー、レプリカセットなど、Red Hat OpenShift にあるすべてのネイティブリソースと連携します。イメージストリームを監視すると、新しいイメージが追加または修正されたときにビルドやデプロイで通知を受信でき、その応答として、それぞれ自動的にビルドやデプロイを自動的に起動できます。

  例として、コア・レイヤー、ミドルウェア・レイヤー、およびアプリケーション・レイヤーという 3 つのコンテナ・イメージ・レイヤーを使用してビルドされたアプリケーションがあるとします。ベースイメージに問題が検出されました。このイメージは Red Hat で再ビルドされ、Red Hat エコシステム・カタログに push されます。イメージストリームが有効になっている場合、Red Hat OpenShift はイメージが変更されたことを検出できます。このイメージに依存し、トリガーが定義されているビルドがあれば、Red Hat OpenShift は修正されたベースイメージを組み込んで、アプリケーション・イメージを自動的に再ビルドします。

  ビルドが完了すると、更新されたカスタムイメージは Red Hat OpenShift の内部レジストリに push されます。Red Hat OpenShift は、内部レジストリ内のイメージへの変更を即座に検出し、トリガーが定義されているアプリケーションがあれば、更新されたイメージを自動的にデプロイし、プロダクションで実行されているコードが常に最新のイメージと同一であることを保証します。これらの機能すべてが連携して、CI/CD プロセスとパイプラインにセキュリティ機能が組み込まれます。

4. セキュリティをアプリケーション・パイプラインに統合する

Red Hat OpenShift には、CI および Tekton 向けの Jenkins インスタンスが統合されています。これはコンテナ (サーバーレスを含む) で機能する次世代 Kubernetes CI/CD パイプラインです。Red Hat OpenShift には機能豊富な RESTful API も含まれており、独自のビルドツールや CI/CD ツール、およびプライベート・イメージ・レジストリの統合に利用できます。

アプリケーション・セキュリティのベストプラクティスは、自動化されたセキュリティテストをレジストリ、統合開発環境 (IDE)、CI/CD ツールを含むパイプラインに統合することです。

レジストリ：コンテナイメージをプライベート・コンテナ・レジストリにスキャンでき、またそうする必要があります。Red Hat Quay を Clair セキュリティスキャナーと併用して、脆弱性が発見されたときに開発者に通知できます。OpenShift Container Security Operator と Red Hat Quay と組み合わせて、OpenShift コンソールで、デプロイしたイメージにある既知の脆弱性をクラスタ全体で把握できます。または、サードパーティが認定した複数のコンテナスキャナーを Red Hat エコシステム・カタログから見つけられます。

IDE：Red Hat Dependency Analytics 統合開発環境 (IDE) プラグインは、コードを初めて IDE に投入したとき、脆弱性の警告とプロジェクト依存関係の修復アドバイスを提供します。

CI/CD：スキャナーを CI と統合して、既知の脆弱性に照らしてリアルタイムでチェックできます。コンテナ内のオープンソース・パッケージをカタログ化し、既知の脆弱性を通知し、過去にスキャンされたパッケージで新たな脆弱性が発見されたときに最新情報を知らせます。

さらに、CI プロセスには、セキュリティスキャンで発見された問題があるビルドにフラグを立てるポリシーが含まれているべきです。これにより、チームは問題に対処するための適切な措置を後回しにするのではなくすぐに講じることができます。

最後の点として、ビルドとデプロイの間にコンテナが改竄されていないことを確認できるよう、カスタムビルドのコンテナには署名することをお勧めします。

デプロイ：デプロイの構成、セキュリティ、コンプライアンスの管理

デプロイの効果的なセキュリティには、Kubernetes プラットフォームの保護とデプロイポリシーの自動化が含まれます。Red Hat OpenShift では以下の機能をすぐに利用できます。

1. プラットフォーム構成とライフサイクル管理
2. ID 管理とアクセス管理
3. プラットフォームのデータと接続されたストレージの保護
4. デプロイポリシー

5. プラットフォーム構成とライフサイクル管理

2019 年夏に公開された「Cloud Native Computing Foundation (CNCF) Kubernetes Security Audit」では、Kubernetes に対する最大のセキュリティ脅威は、Kubernetes コンポーネントの構成と強化の複雑性であると結論づけられています。Red Hat OpenShift は Kubernetes Operators を使用してこの問題に対処しています。

Operator は、Kubernetes ネイティブのアプリケーションをパッケージ、デプロイ、管理する手段です。Operator はカスタムコントローラーとして機能し、Kubernetes アプリケーション・プログラミング・インタフェースを、アプリケーションの管理に必要なアプリケーション固有のロジックで拡張できます。Red Hat OpenShift プラットフォームの各コンポーネントは Operator にラッピングされ、OpenShift の自動化された構成、監視、管理を提供します。個々の Operator は API サーバーやソフトウェア・デファインド・ネットワーク (SDN) などのコンポーネントを直接構成し、クラスタバージョンの Operator はプラットフォーム上の複数の Operator を管理します。Operator により、カーネルからスタック内の上位サービスまで、アップデートを含めたクラスタ管理を自動化できます。

コンテナ・プラットフォームの重要な価値の 1 つは、開発者用のセルフサービス機能を実現することです。開発チームは承認されたレイヤー上でビルドされたアプリケーションを簡単かつ迅速に提供できるようになります。セルフサービス・ポータルによってチームは、セキュリティを実現しながらコラボレーションを促進するために必要となる、十分な制御性を得られます。Operator Lifecycle Manager (OLM) は Red Hat OpenShift クラスタユーザー向けのフレームワークを提供し、Operator を見つけて、アプリケーションの有効化に必要なサービスをデプロイするために使用します。OLM を使用すると、ロールベースのアクセス制御をインストールおよびアップグレードし、使用できる Operator に割り当てることができます。

コンプライアンスへの準拠を支援するため、Red Hat OpenShift には Compliance Operator が用意されており、コンプライアンス・フレームワークに必要な技術的制御でプラットフォームのコンプライアンスを自動化します。Compliance Operator によって Red Hat OpenShift 管理者は目的とするクラスタのコンプライアンス状態を記述し、ギャップとその修復方法の概要を取得できます。Compliance Operator はクラスタを実行するノードを含めて、すべてのプラットフォーム・レイヤーのコンプライアンスを評価します。クラスタノード上で定期的にファイル整合性チェックを実行する File Integrity Operator もあります。

6. ID 管理とアクセス管理

Kubernetes には開発者向けと管理者向けの豊富な機能が用意されているので、強力な ID 管理と RBAC はコンテナ・プラットフォームにとって欠かせない要素です。Kubernetes API は、大規模なコンテナ管理を自動化するための鍵となります。API はたとえば、Pod やサービスの構成およびデプロイなどのリクエストの開始や検証に使用されます。

API 認証と認可は、コンテナ・プラットフォームを保護するために重要です。API サーバーはアクセスの中心点となるので、最高レベルのセキュリティ監視を受ける必要があります。Red Hat OpenShift コントロールプレーンには Cluster Authentication Operator を通じてビルトインの認証が含まれます。開発者、管理者、サービスアカウントは、OAuth アクセストークンを取得して API に認証されます。管理者は任意の ID プロバイダーをクラスタに設定して、トークンを受信する前にユーザーが認証されるようにすることができます。Lightweight Directory Access Protocol (LDAP) ディレクトリを含めて、9 つの ID プロバイダーをサポートしています。

Red Hat OpenShift では、きめ細かな RBAC がデフォルトで有効になっています。RBAC オブジェクトは、クラスタ内で所定のアクションの実行がユーザーに許可されているかどうかを判定します。クラスタ管理者はクラスタロールとバインディングを使用して、OpenShift クラスタおよびクラスタ内のプロジェクトへのアクセスレベルを制御できます。

7. プラットフォームデータの保護

Red Hat OpenShift はデフォルトで移動中のデータを保護して、Kubernetes をハードニングします。保管されているデータを保護するオプションもあります。

Red Hat OpenShift では、移動中のプラットフォームデータを以下の方法で保護します。

- 相互に通信するすべてのコンテナ・プラットフォーム・コンポーネントで、移動中のデータを https を使用して暗号化する
- コントロールプレーンとのすべての通信を Transport Layer Security (TLS) を通じて送信する
- API サーバーへのアクセスは、X.509 証明書またはトークンベースにする
- 不正なトークンが与えるダメージを抑えるためにプロジェクトクォータを使用する
- 固有の認証局 (CA) と証明書で etcd を設定する(Kubernetes では、状態のマスターデータが etcd に永続的に保管されます。他のコンポーネントは etcd を参照し、変更を検出したらその新しい状態へと遷移します)
- プラットフォーム証明書を自動的にローテーションする

Red Hat OpenShift では、以下の方法で保管中のプラットフォームデータを保護します。

- Red Hat Enterprise Linux CoreOS ディスクと etcd データストアを暗号化して、セキュリティを強化する (オプション)
- Red Hat OpenShift を Federal Information Processing Standards (FIPS) 対応にする。FIPS 140-2 は米国政府のセキュリティ規格で、暗号化モジュールの承認に使用されます。Red Hat Enterprise Linux CoreOS を FIPS モードで起動すると、Red Hat OpenShift プラットフォーム・コンポーネントは Red Hat Enterprise Linux 暗号化モジュールを呼び出します。

コンテナは、ステートレス・アプリケーションとステートフル・アプリケーションの両方において役立ちます。Red Hat OpenShift では一時的ストレージと永続的ストレージの両方をサポートしています。接続されたストレージの保護は、ステートフル・サービスのセキュリティ保護において重要な要素です。Red Hat OpenShift は、ネットワークファイルシステム (NFS)、Amazon Web Services (AWS) Elastic Block Stores (EBS)、Google Compute Engine (GCE) Persistent Disk、Azure Disk、iSCSI、Cinder など、複数のストレージタイプをサポートしています。

さらに、Red Hat OpenShift® Container Storage は、Red Hat OpenShift Container Platform に統合され、最適化された、ソフトウェア・デファインドの永続ストレージです。OpenShift Container Storage は、ハイブリッドマルチクラウド上で暗号、レプリケーション、可用性などの機能を必要とするクラウドネイティブ・アプリケーションにとって、拡張性の高い永続的ストレージとなります。

- **永続ボリューム (PV)** は、リソースプロバイダーがサポートしている方法であればどの方法でもホストにマウントできます。プロバイダーはそれぞれ異なる機能を持ち、各 PV のアクセスモードはその特定のボリュームでサポートされている特定のモードに設定されます。たとえば、NFS は複数の読み取り/書き込みクライアントをサポートできますが、特定の NFS PV はサーバー上で読み取り専用でエクスポートされることがあります。各 PV は、特定の PV の機能を説明するアクセスモードのセットを取得します。たとえば、ReadWriteOnce、ReadOnlyMany、ReadWriteMany などです。
- **共有ストレージ** (NFS、Ceph、Gluster など) の場合、共有ストレージの永続的ボリューム PV にそのグループ ID (gid) を PV リソースのアノテーションとして登録させます。PV が Pod により要求されると、アノテーションとして登録された gid が Pod の従属グループに追加され、その Pod に共有ストレージのコンテンツへのアクセス権が与えられます。
- **ブロックストレージ** (EBS、GCE Persistent Disks、iSCSI など) の場合、コンテナ・プラットフォームは SELinux 機能を使用して非特権 Pod 用にマウントされたボリュームの root を保護し、マウントされているボリュームは関連付けられたコンテナに所有され、そのコンテナからのみ見えるようにします。

選択したストレージ・ソリューションで利用可能なセキュリティ機能も併せて活用しましょう。

8. ポリシーベースのデプロイの自動化

強力なセキュリティには自動化されたポリシーが含まれます。これらのポリシーを使用して、セキュリティの観点からコンテナとクラスタのデプロイメントを管理することができます。

- ポリシーベースのコンテナデプロイ

Red Hat OpenShift クラスタを、特定のイメージレジストリからのイメージの pull を許可または拒否できるように設定できます。プロダクションクラスタについては、プライベートレジストリからのイメージのみデプロイを許可するのがベストプラクティスです。

Red Hat OpenShift のセキュリティ・コンテキストによる制約 (SCC) アドミッション・コントローラー・プラグインは、システムに受け入れられるために Pod が従う必要のある一連の条件を定義します。**セキュリティ・コンテキストによる制約**により、デフォルトで権限を制限できます。これは重要であると同時にベストプラクティスでもあります。Red Hat OpenShift のセキュリティ・コンテキストによる制約 (SCC) によって、特権コンテナが OpenShift ワーカーノードで実行されないようにできます。ホストネットワークおよびホストプロセス ID へのアクセスはデフォルトで拒否されます。

必要な権限を有するユーザーは、希望する場合にはデフォルトの SCC ポリシーを調整して制約を緩めることができます。

Red Had Advanced Cluster Management for Kubernetes が提供する**高度なアプリケーション・ライフサイクル管理**はオープンスタンダードを使用して、既存の CI/CD パイプラインおよびガバナンス統制へ統合される配置ポリシーを使用してアプリケーションをデプロイします。

- ポリシーベースのマルチクラスタ管理

複数クラスタをデプロイすると、複数のアベイラビリティー・ゾーンにわたるアプリケーションの高可用性や、 Amazon Web Services (AWS)、Google Cloud、Azure などの複数のクラウドプロバイダーにまたがるデプロイや移行の共通管理機能が可能になります。複数のクラスタを管理する際は、使用しているオーケストレーション・ツールが、デプロイされた異なるインスタンス全体に必要なセキュリティを提供できる必要があります。ここでも、構成、認証、認可が重要な要素となります。加えて、実行されている場所に関わらず、アプリケーションにデータを安全に渡し、クラスタ全体でアプリケーションのポリシーを管理できる能力も重要です。Red Had Advanced Cluster Management for Kubernetes には、以下の機能があります。

- **マルチクラスタ・ライフサイクル管理：**Kubernetes クラスタを確実に、一貫性を保ちながら、大規模に作成、更新、破棄できます。
- **ポリシー駆動型ガバナンスリスクとコンプライアンス：**ポリシーを利用して、業界の企業標準に従ってセキュリティ統制の一貫性を設定し、維持します。コンプライアンスポリシーを指定して、1 つ以上のマネージドクラスタに適用することもできます。

実行中のアプリケーションの保護

インフラストラクチャの他に、アプリケーション・セキュリティの維持は重要です。コンテナ化アプリケーションをセキュリティ保護するには、以下のことが必要です。

1. コンテナの分離
2. アプリケーションとネットワークの分離
3. アプリケーションアクセスのセキュリティ保護
4. 可観測性

9. コンテナの分離

コンテナというパッケージ化とオーケストレーションのテクノロジーを最大限に活用するために、運用チームにはコンテナを実行するための適切な環境が必要です。運用チームに必要なのは、コンテナを境界で保護できるオペレーティングシステム (OS) です。これは、コンテナから脱出できないようホストカーネルを保護し、コンテナ同士を互いから保護します。

コンテナとは、分離性とリソース制限機構を備えた Linux プロセスであり、共有ホストカーネル上でサンドボックス化されたアプリケーションの実行を可能にします。コンテナのセキュリティ保護は、Linux の実行プロセスをセキュリティ保護する場合と同じアプローチで行われます。

NIST Special Publication 800-190 では、コンテナ向けに最適化された OS を使用してセキュリティを強化することを推奨しています。Red Hat OpenShift のオペレーティングシステム・ベースである Red Hat Enterprise Linux CoreOS は、ホスト環境を最小限に抑えながらコンテナ向けにチューニングすることで、攻撃対象の領域を縮小します。Red Hat Enterprise Linux CoreOS には Red Hat OpenShift の実行に必要なパッケージのみが含まれ、ユーザー空間は読み取り専用です。プラットフォームは Red Hat OpenShift と一緒にテスト、バージョン管理、リリースされ、クラスタによって管理されます。Red Hat Enterprise Linux CoreOS のインストールと更新は自動化され、クラスタとの対応が常時維持されます。また、ユーザーが選択したインフラストラクチャをサポートし、Red Hat Enterprise Linux エコシステムの大半を継承します。

Red Hat OpenShift プラットフォーム上で動作するどの Linux コンテナも、Red Hat OpenShift ノードに組み込まれた、強力な Red Hat Enterprise Linux セキュリティ機能で保護されます。Linux 名前空間、SELinux、Cgroups、ケーパビリティ (capabilities)、およびセキュア・コンピューティング・モード (seccomp) を使用して、Red Hat Enterprise Linux 上で動作するコンテナのセキュリティを保護します。

- Linux 名前空間は、コンテナの分離の基礎となります。名前空間を使用すると、名前空間内のプロセスにはグローバルリソースに独自のインスタンスがあるように見えます。名前空間はコンテナ内で専用のオペレーティングシステムを実行しているかのように見せる抽象化を提供します。
- SELinux は、コンテナを他のコンテナやホストから隔離した状態に保つための、追加のセキュリティレイヤーを提供します。SELinux では、管理者はすべてのユーザー、アプリケーション、プロセス、およびファイルに対して強制アクセス制御 (MAC) を適用できます。たとえるならば、SELinux は、(偶然または意図的に) 名前空間による抽象化をすり抜けてしまった場合に壁となるものです。SELinux はコンテナランタイムの脆弱性を緩和し、SELinux をうまく構成すると、コンテナプロセスのコンテナ外部への脱出を防止できます。

- Cgroups (コントロールグループ) は、一連のプロセスのリソース使用量 (CPU、メモリー、ディスク I/O、ネットワークなど) を制限し、集計し、分離します。Cgroups を使用することで、同じホスト上の別のコンテナによってコンテナリソースがつぶされないようにできます。また、Cgroup は、一般的な攻撃ベクトルである擬似デバイスを制御するためにも使用できます。
- Linux の各種 Capabilityを使用すると、コンテナの特権をロックダウンすることができます。Capability とは、個別に有効化または無効化できるよう分離された部分的な root 権限です。Capability を使用することで、raw インターネットプロトコル (IP) パケットの送信や、1024 未満のポートへのバインドなどを行うことができます。コンテナを実行する際、複数の Capability を無効化しても、大半のコンテナ化アプリケーションには影響がありません。
- 最後に、セキュア・コンピューティング・モード (seccomp) プロファイルをコンテナに関連付けることにより、使用可能なシステムコールを制限できます。

10. アプリケーションとネットワークの分離

Kubernetes を企業全体で使用する際には、マルチテナントセキュリティが重要となります。マルチテナンシーにより、さまざまなチームに同じクラスタを共有させつつ、相互の環境への不正アクセスを防止できます。Red Hat OpenShift は、カーネルの名前空間、SELinux、RBAC、Kubernetes (プロジェクト) 名前空間、ネットワークポリシーを組み合わせて、マルチテナンシーをサポートします。

- **Red Hat OpenShift プロジェクト**は、SELinux アノテーションを持つ Kubernetes 名前空間です。プロジェクトによって、チーム、グループ、部門間でアプリケーションが切り分けられます。ローカルロールとバインディングを使用して、個々のプロジェクトにアクセスできるユーザーを制御します。
- **セキュリティ・コンテキストによる制約**により、デフォルトで権限を制限できます。これは重要であると同時にベストプラクティスでもあります。Red Hat OpenShift のセキュリティ・コンテキストによる制約 (SCC) によって、特権コンテナが OpenShift ワーカーノードで実行されないようにできます。ホストネットワークおよびホストプロセス ID へのアクセスはデフォルトで拒否されます。

コンテナ化された最新のマイクロサービス・ベースのアプリケーションをデプロイすることは、しばしば複数のノードに複数のコンテナを分散させてデプロイすることを意味します。これらのマイクロサービスは相互に検出して通信する必要があります。ネットワーク防御を念頭におくと、単一クラスタでトラフィックを分割し、そのクラスタ内の異なるユーザー、チーム、アプリケーション、および環境を分離できるコンテナ・プラットフォームが必要です。クラスタへの外部アクセスと、クラスタサービスから外部コンポーネントへのアクセスを管理するツールも必要です。ネットワークを分離するには、以下の重要な要素が必要です。

- **Ingress トラフィック制御：**Red Hat OpenShift に含まれる CoreDNS は Pod に名前解決サービスを提供します。Red Hat OpenShift ルーター (HAProxy) は Ingress とルートをサポートし、クラスタ上で動作するサービスへの外部アクセスを提供します。両方とも再暗号化とパススルーのポリシーをサポートします。“reencrypt” は HTTP トラフィックを復号して転送時に再度暗号化し、“passthrough” は TLS を終了せずにトラフィックを通過させます。
- **ネットワーク名前空間：**ネットワーク防御の最前線にあるのがネットワーク名前空間です。コンテナの各集合体 (「Pod」) は、バインド先となる独自の IP とポート範囲を取得し、それによりノード上の Pod ネットワークは互いから分離されます。Pod の IP アドレスは Red Hat OpenShift ノードの接続先である物理ネットワークに依存しません。

- **ネットワークポリシー：**Red Hat OpenShift SDN はネットワークポリシーを使用して、Pod 間のきめ細かな通信制御を行います。ネットワークトラフィックは、任意の Pod 間での、特定のポート上で特定の方向の送受信を制御できます。ネットワークポリシーをマルチテナントモードで設定すると、各プロジェクトには固有の仮想ネットワーク ID が割り振られ、プロジェクトネットワークはノード上で相互に分離されます。マルチテナント・モード (デフォルト) では、プロジェクト内の Pod は相互に通信できますが、別の名前空間の Pod と別のプロジェクトの Pod またはサービスとの間でパケットを送受信できません。
- **Egress トラフィック制御：**Red Hat OpenShift には、ルーターまたはファイアウォールを使用して、クラスタ上で実行されるサービスからの Egress トラフィックを制御する機能もあります。たとえば、IP ホワイトリストを使用して外部データベースへのアクセスを付与できます。

11. アプリケーションアクセスのセキュリティ保護

アプリケーションのセキュリティ保護には、アプリケーションユーザーの管理に加えて、API 認証と認可が含まれます。

- **ユーザーアクセスの制御**

Web シングルサインオン (SSO) 機能は、先進的なアプリケーションにとって欠かせない部分です。コンテナ・プラットフォームには、開発者がアプリケーションを構築するときに使用できる多数のコンテナ化されたサービスが存在することがあります。Red Hat Single Sign-On は、完全にサポートされたすぐに使える Security Assertion Markup Language (SAML) 2.0 または OpenID Connect ベースの認証、Web シングルサインオン、アップストリームの Keycloak プロジェクトに基づくフェデレーション・サービスです。Red Hat Single Sign-On には、Red Hat Fuse および Red Hat JBoss Enterprise Application Platform 用のクライアントアダプターが搭載されています。Red Hat Single Sign-On によって、Node.js アプリケーションの認証と Web シングルサインオンが可能になり、Microsoft Active Directory や Red Hat Enterprise Linux Identity Management などの LDAP ベースのディレクトリサービスと統合することができます。Red Hat Single Sign-On は、Facebook、Google、Twitter などのソーシャル・ログイン・プロバイダーとも統合されています。

- **API アクセスの制御**

API は、マイクロサービスで構成されているアプリケーションにとって重要です。これらのアプリケーションには複数の独立した API サービスがあるので、サービス・エンドポイントの急増につながります。そのため、ガバナンスのための追加ツールが必要になります。API 管理ツールを使用することをお勧めします。Red Hat 3Scale API Management は、API 認証とセキュリティに関するさまざまな標準オプションを提供します。これらのオプションを単独または組み合わせて使用することで、資格情報の発行やアクセス制御を行えます。

Red Hat 3scale API Management のアクセス制御機能は、基本的なセキュリティと認証以上のものを提供します。アプリケーション・プランとアカウントプランでは、特定のエンドポイント、メソッド、およびサービスへのアクセスを制限し、ユーザーグループにアクセスポリシーを適用できます。アプリケーション・プランでは、API 使用率のレート制限を設定し、開発者グループのトラフィックフローを制御できます。インフラストラクチャを保護し、トラフィックがスムーズに流れるようにするためには、期間あたりの受信 API コールを制限します。また、レート制限に達するか超過したアプリケーションの超過アラートを自動的にトリガーし、制限を超過したアプリケーションに対する動作を定義できます。

アプリケーショントラフィックのセキュリティ保護

クラスタの Ingress および Egress オプションによるアプリケーショントラフィックのセキュリティ保護については、本書のセクション 10 で説明しています。マイクロサービスベースのアプリケーションでは、クラスタ上のサービス間のトラフィックに対するセキュリティも同じく重要です。この管理レイヤーは、サービスメッシュを使って構築できます。「サービスメッシュ」という用語は、分散マイクロサービス・アーキテクチャでアプリケーションを構成するマイクロサービスのネットワークと、これらのマイクロサービス間のインタラクションを指します。

オープンソース・プロジェクトの Istio をベースとする Red Hat OpenShift Service Mesh は既存の分散アプリケーションに透過的なレイヤーを追加し、サービス間通信を管理します。サービスコードに変更を加える必要はありません。Red Hat OpenShift Service Mesh はマルチテナント Operator を使用してコントロールプレーンのライフサイクルを管理するので、OpenShift Service Mesh をプロジェクトベースで使用できるようになります。さらに、OpenShift Service Mesh ではクラスタをスコープとする RBAC リソースは不要です。

Red Hat OpenShift Service Mesh は、ディスカバリー、負荷分散、そしてセキュリティの鍵となるサービス間認証と暗号化、障害復旧、メトリクス、監視を提供します。

3scale Istio Adapter は Red Hat OpenShift Service Mesh 内で実行するサービスにラベル付けできる、オプションのアダプターです。

12. 可観測性

Red Hat OpenShift クラスタを監視および監査する能力は、クラスタとクラスタユーザーを不正使用から保護するために重要な要素です。Red Hat OpenShift には組み込みの監視と監査が搭載されており、オプションでロギングスタックも利用できます。

OpenShift Container Platform サービスは、Prometheus とそのエコシステムで構成されたビルトインの監視ソリューションに接続します。アラートダッシュボードを利用できます。クラスタ管理者はユーザー定義のプロジェクトについて監視を有効にすることもできます。Red Hat OpenShift にデプロイされたアプリケーションを、クラスタ監視コンポーネントを利用できるように構成できます。

イベントの監査はセキュリティのベストプラクティスで、一般に規制フレームワークへの準拠に必要です。中核である Red Hat OpenShift 監査はクラウドネイティブ・アプローチを使用して設計されたもので、一元化とレジリエンシー (回復力) を提供します。Red Hat OpenShift では、すべてのノードでホスト監査とイベント監査がデフォルトで有効です。Red Hat OpenShift は、管理と監査データへのアクセスを、非常に高い柔軟性で設定できます。API サーバー監査ログに記録する情報量を、使用する監査ログポリシー・プロファイルを選択することで制御できます。

監視、監査、ログデータは RBAC で保護されます。プロジェクトデータはプロジェクト管理者が、クラスタデータはクラスタ管理者が使用できます。

ベストプラクティスとして、すべての監査とログイベントをセキュリティ情報およびイベント管理 (SIEM) システムに転送するようにクラスタを設定し、完全性管理、保持、分析を行います。クラスタ管理者はクラスタログをデプロイして、ホストおよび API の監査ログなどの Red Hat OpenShift クラスタからのすべてのログと、アプリケーション・コンテナ・ログやインフラストラクチャ・ログを集約できます。クラスタログはこれらのログをクラスタノード全体から集約し、デフォルトのログストアに保存します。ユーザーが選んだ SIEM にログを転送する方法は複数あります。

強固なエコシステムによるセキュリティの拡張

コンテナと Kubernetes セキュリティをさらに強化したり、既存のポリシーに適合させたりするには、サードパーティのセキュリティツールとの統合を検討することができます。Red Hat には認定パートナーからなる広範なエコシステムがあり、以下のようなソリューションを提供しています。

- 特権アクセス管理
- 外部認証局
- 外部 Vault および鍵管理ソリューション
- コンテナ・コンテンツ・スキャナーおよび脆弱性管理ツール
- コンテナランタイム分析ツール
- SIEM

まとめ

コンテナベースのアプリケーションとマイクロサービスのデプロイが目的とするのは、セキュリティだけではありません。コンテナ・プラットフォームは、開発チームや運用チームに適したエクスペリエンスを提供する必要があります。必要なのは、各チームが必要とする機能を損なうことなく、開発と運用の両者が運用効率とインフラストラクチャの利用効率を向上できる、セキュリティ重視でエンタープライズ・グレードのコンテナベースのアプリケーション・プラットフォームです。

Red Hat OpenShift は、次のようなセキュリティ機能を提供する、標準と可搬性のある Linux コンテナを基盤として構築されています。

- 統合されたビルドと CI/CD ツールによる安全な DevOps プラクティスの実現支援
- ビルトインのプラットフォーム構成、コンプライアンス、ライフサイクル管理を備えた、ハードニングされたエンタープライズ対応 Kubernetes
- エンタープライズ認証システムとの統合による強力な RBAC
- クラスタの Ingress と Egress を管理するオプション
- ネットワークのマイクロセグメンテーションのサポートによる SDN とサービスメッシュの統合
- リモート・ストレージ・ボリュームのセキュリティ保護のサポート
- 強力な分離によってコンテナの大規模な実行向けに最適化された、Red Hat Enterprise Linux CoreOS
- ランタイムセキュリティを自動化するデプロイポリシー
- 監視、監査、ロギングの統合

さらに Red Hat OpenShift は、幅広いプログラミング言語、フレームワーク、サービスをサポートしており (図 4)、Red Hat Advanced Cluster Management for Kubernetes は緊密に統合されたマルチクラスタ管理を提供します。

Red Hat OpenShift は OpenStack、VMware、ベアメタル、AWS、Google Cloud Platform (GCP)、Azure、IBM Cloud、および Red Hat Enterprise Linux をサポートするあらゆるプラットフォームで実行できます。Red Hat は、AWS および GCP 上でパブリッククラウド・サービスとして Red Hat OpenShift Dedicated を提供します。Azure Red Hat OpenShift の提供は Microsoft と Red Hat が共同で行っており、Red Hat OpenShift Service on AWS は Red Hat と Amazon が共同で提供しています。

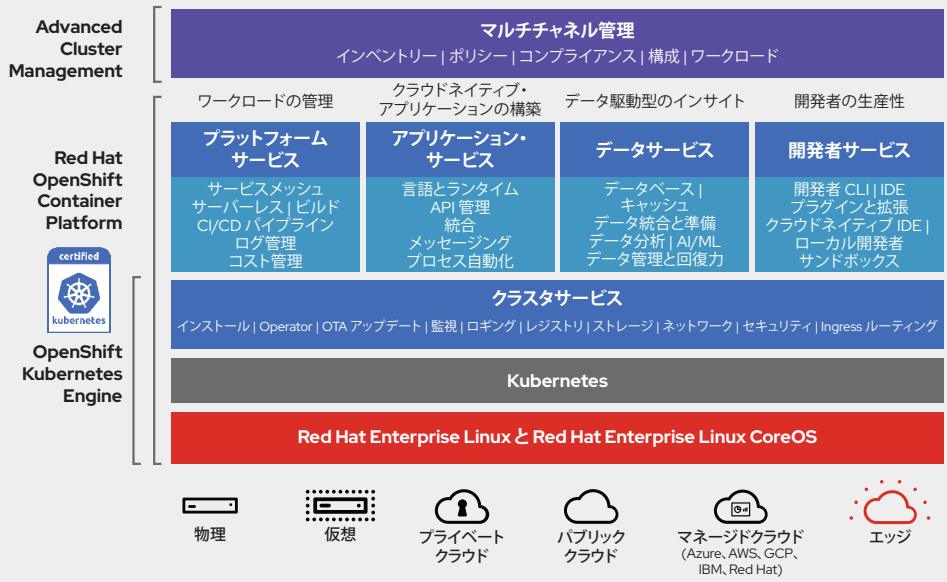

図 4. Red Hat OpenShift は機能を完備したテクノロジースタックを、物理、仮想、プライベートクラウドおよびパブリッククラウドに提供する

Red Hat は、世界有数のプロバイダーとして 20 年以上にわたり、優れた信頼性を誇るオープンソースのソリューションを法人のお客様に提供してきました。Red Hat OpenShift Container Platform や Red Hat Advanced Cluster Management for Kubernetes などのソリューション、そしてコンテナ対応の Red Hat 製品ポートフォリオを通じて、Red Hat は同水準の信頼性とセキュリティをコンテナに提供します。

RED HAT について

エンタープライズ・オープンソース・ソフトウェア・ソリューションのプロバイダーとして世界をリードする Red Hat は、コミュニティとの協業により高い信頼性と性能を備える Linux、ハイブリッドクラウド、コンテナ、および Kubernetes テクノロジーを提供しています。Red Hat は、新規および既存 IT アプリケーションの統合、クラウドネイティブ・アプリケーションの開発、Red Hat が提供する業界トップレベルのオペレーティングシステムへの標準化、複雑な環境の自動化、セキュリティ保護、運用管理を支援します。受賞歴のあるサポート、トレーニング、コンサルティングサービスを提供する Red Hat は、Fortune 500 企業に信頼されるアドバイザーです。クラウドプロバイダー、システムインテグレーター、アプリケーションベンダー、お客様、オープンソース・コミュニティの戦略的パートナーとして、Red Hat はデジタル化が進む将来に備える企業を支援します。

アジア太平洋
+65 6490 4200
apac@redhat.com

オーストラリア
1 800 733 428

インド
+91 22 3987 8888

インドネシア
001 803 440 224

日本
0120 266 086
03 5798 8510

韓国
080 708 0880

マレーシア
1 800 812 678

ニュージーランド
0800 450 503

シンガポール
800 448 1430

中国
800 810 2100

香港
800 901 222

台湾
0800 666 052

fb.com/RedHatJapan
twitter.com/RedHatJapan
linkedin.com/company/red-hat

jp.redhat.com
#F26463_1220