OSSを活用したクラウド運用管理の潮流

企業におけるクラウドの利用が一般的となり、複数環境に対する運用管理を低コストで効率的に行うことが求められる中、オープンソースソフトウェア（以下、OSS）によるクラウド運用管理が注目されています。Infrastructure as Codeをベースにした運用の自動化とインフラCI/CDの実現、クラウドの特性を考慮したセキュリティー対策の実装、クラウドネイティブ環境における監視やロギングの運用を、どのようにOSSで実現するのかについて解説します。

1. はじめに

デジタルトランスフォーメーションに邁進する企業が増える中、クラウドやコンテナといった技術の利用が広がり続けています[1][2]。ITシステム環境は、企業が提供するWebサービスやデジタル・サービスといった顧客サービスを支える根幹であり、サービス提供を継続するために、システム停止がますます許容されなくなってきています。また、ITシステムの運用管理を低コストで行うことが求められているにもかかわらず、運用管理に多大なワークロードが割かれています[3]。そのため、これまでのようにオンプレミスの環境のみを運用管理するだけではなく、クラウドを利用したシステムの運用管理を低コストで効率的に実現する必要があります。そこで、クラウドに対応できる可搬性を持ち、低コストで運用管理を行えるOSSへの注目が高まってきています。

OSSは、ベンダー・ロックインを好まない企業や、低コストでITシステム開発や運用を実現したい企業に多く利用されるようになってきています。多くのシステムで利用されているLinuxやWindowsといったOSに対応しているものが多く、クラウドやコンテナといった技術にも対応して利用できるものが多く登場してきています。こうしたOSSはオンプレミスでも利用できるものが多く、ハイブリッド・クラウドやマルチクラウドでの利用が可能です。OSSをITシステムの運用管理にも利用することで、安価で効率的な運用管理を、複数環境に対して行うことが可能です。また、クラウド・ベンダーが提供している運用サービスにも、多くのOSSが利用されています。

本稿では、クラウド運用管理の項目ごとに、OSSを自分たちで組み合わせて活用する事例や考慮点について解説します。

2. クラウド運用管理でOSSを使う場合の考慮点

クラウド運用管理でOSSを活用する場合、開発の現場で利用するのとは異なる考慮点が存在します。運用管理を行う場合、期間が限定される開発の場合よりも長期間の利用を前提として、サポートの手厚いOSSの利用を検討する必要があります。OSSの中には、古くからデファクト・スタンダードとして使われているものや、急速に利用が拡大し多くのシステムで利用されているものがあります。これらに共通するのは、コミュニティーの活発さやサポートが充実していることです。修正ファイルの開発やバージョン管理についても、それぞれのOSSのコミュニティーに委ねられているため、コミュニティーが活発に活動し手厚いサポートが受けられるOSSを利用することが、運用管理を考える上では重要となります。それまで商用ソフトウェアを利用していた運用管理をOSSに置き換える場合、商用ソフトウェアと同等程度のサポートを期待する企業が多いこともその理由です。

また、OSSによる運用管理を行う場合、これまでの商用ソフトウェアとは異なる社内のレビューや、契約書への記載、OSSライセンスの種類についても十分注意する必要があります。

ITシステムの運用管理には、さまざまな管理項目があ

ります。運用自動化、セキュリティー管理、監視・ログ収集といった項目ごとに、OSSをどのように活用してクラウド運用管理を行うべきかなのか、事例や考慮点を紹介します。

3. Infrastructure as CodeとインフラCI/CD

クラウド環境においては、Infrastructure as Codeの考え方に基づいてコードでシステムを運用管理する手法が一般的となってきています。システムはコードを実行した結果により構成されるものであり、コードを維持・管理していくことがシステムの維持・管理をしていくことになります。これによりシステムは常にあるべき姿を保ち続けることができます（べき等性を保った状態）。また、Infrastructure as Codeに基づいて運用することで、ソフトウェア開発のベスト・プラクティスである、CI（継続的インテグレーション）、CD（継続的デリバリー）がインフラにも適用できるというメリットも享受できます。自動化によりシステム運用のスピードや効率を上げるだけではなく、インフラCIやCDを実現するOSSの活用方法について考えます。

図1は、OSSと関連ツールを活用してInfrastructure as CodeとインフラCI/CDを実現する一般的なアーキテクチャーの例です。

Ansibleは構成管理ツールとして、業界のリーダーの位置付けで多くの企業で利用され、エージェントなしでの実装が可能であり、多様なエンドポイントに利用することができるOSSです。Ansibleにより、べき等性を保った構築・運用が可能となり、自動化により実現される効率化・迅速化などのメリットを得ることができます。コード管理のためのソフトウェア構成管理ツールとしてはGitLabを利用します。コードの更新を基にした実装までを行うCI、開発環境での実装から本番環境での実装までの一連を自動化するCDを実現するために、CIツールとして各ツール間を連携するJenkins、バーチャル・マシンの払い出しを行うTerraform、テスト自動化のためのServerspecを利用しています。JenkinsはOpenShiftなどのメジャーなクラウドネイティブ・プラットフォームでも利用されています。これらはそれぞれの用途でのデファクト・スタンダードとなっているツールでもあり、長期間の運用管理のために利用するのに適しています。また、OSSではありませんが、CI/CDを実現する上でのユーザーへの通知を行うためにSlackも活用しています。

4. OSSとクラウドにおけるセキュリティー管理

OSSの人気は、ユーザーや開発者の要求が無償か安価で提供される機能的な側面も大きいと考えられます。一方、セキュリティー観点では、そのソースコードが多くの人の目に触れることで洗練され、脆弱性の修正も早いことから広く受け入れられています。本章では、セキュリティー観点での昨今のOSS利用を中心に解説します[4]。特に、従来にはないパブリック・クラウド特有の考慮点やリスク[5]を述べ、関連OSSの利用パターンについて説明します（**図2**）。

4-1. ユーザーIDとアクセス権の管理

まず注意しなければならないのは、エンドユーザーがオンプレミスに加えてクラウド環境にもアクセスする点

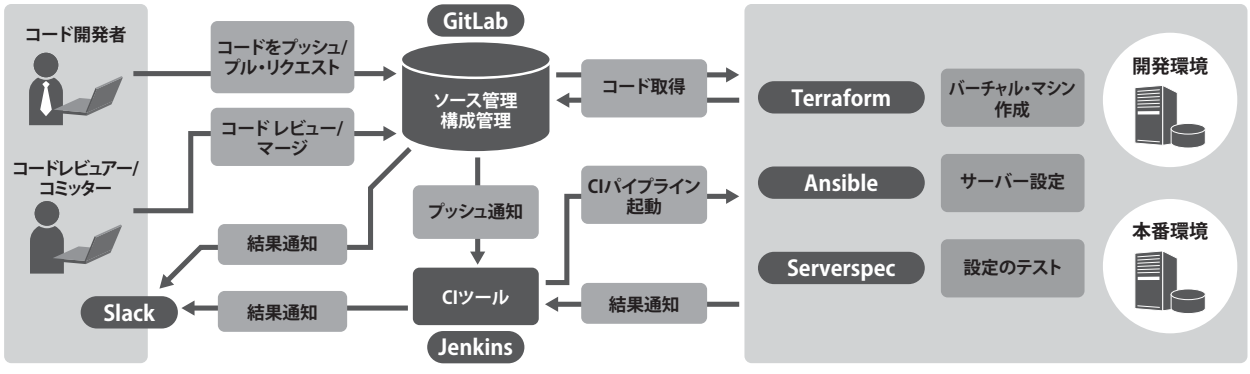

図1. Infrastructure as CodeとインフラCI/CDを実現するアーキテクチャー

です。その際、従来から企業内で利用してきたユーザーIDとは別に、クラウド・サービスごとにユーザーIDを持つと、利便性や管理工数の点で好ましくありません。さらに、付与されたパスワードの入力をサービスごとに毎回求めると、容易に推測可能なパスワードの使用やパスワードの使い回しが増え、セキュリティーのレベルも低下します。

こうした背景から、複数の組織体やサービス間で信頼関係を形成することでID体系を連携させ、一定時間、パスワードの再入力を求めないシングル・サイン・オンを実現するフェデレーション技術の利用が一般的です。ユーザーIDの登録、削除、パスワード設定などの基本機能を提供するOSSとしてOpenIDMがあり、それと連携してフェデレーション機能を提供するOSSとしてOpenAMがあります。これらはForgeRock社が開発、サポートし、フェデレーション未対応のアプリケーションに代理認証を提供するOpenIGなどと組み合わせて使用されています。

4-2. インターネットからの直接アクセスによるリスクへの対応

インターネットに接続できる環境であれば、エンドユーザーがどこからでもサービスを利用できることで、パブリック・クラウドの利便性は高くなります。テレワークや出張時に必要だったVPNによるイントラネットへの暗号化接続は不要ですが、ここには新たなリスクが生じます。

まず、エンドユーザーからクラウド・サービスへのアクセスのリスクを説明します。インターネットからの直接アクセスが可能になると、パスワードの推測・悪用の可能性が非常に高まります。そこで、ユーザーの認証時に、パスワードに加えて本人しか持ち得ない情報を要求する多要素認証の利用が加速しています。例えば、そのユーザーが保有するメールアドレスや、SMSにワンタイム・パスコードを送信し認証時にその入力を求める、といった手法です。前述のOpenAMはGoogle AuthenticatorなどのOSSとの連携によって多要素認証を実現できます。

次に、パブリック・クラウドで稼働するシステムに目を向けてみましょう。従来、ファイアウォールなどによる防御により、イントラネットとインターネットの境界は明確でした。インターネットに直接面した場所を除いては安全と考えられ、セキュリティー設定が甘いサーバーも多く見られました。しかし、パブリック・クラウドでは、すべてのシステムがリスクの高い場所にあると考える必要があります。

以上のことから、サーバー上で動くOSやミドルウェアのセキュリティー設定がより重要となります。この点では、Chef社が開発を主導するOSSを統合したChef Automate 2が利用でき、セキュリティー設定順守のチェックを自動化できます。Chef Automate 2は、米国の非営利のセキュリティー団体であるCIS(Center for Internet Security)が公開しているセキュリティー設定標準集であるCIS Benchmarks[6]をサポートしており、高い有用性を誇っています。

インターネットからの直接アクセスが可能になるため、攻撃からの防御もより重要になります。侵入の検知、保護の機能を持つOSSとして、非営利団体が開発している

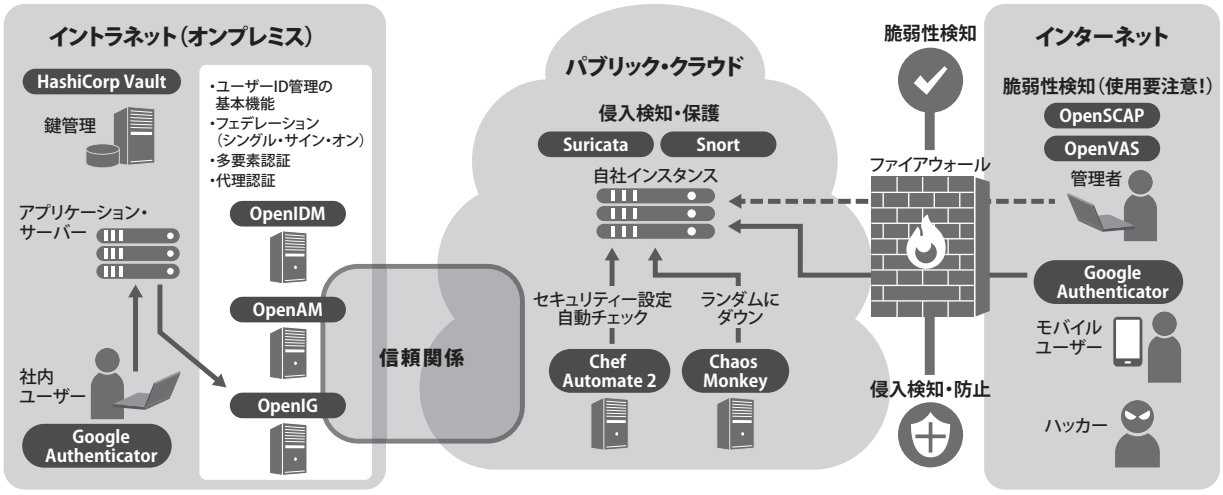

図2. クラウド環境におけるセキュリティー関連OSSの利用

Suricataや、Cisco社に買収され、商用製品への組み込みも多いSnortがあります。また、システムの脆弱性検知のOSSとしては、NIST(アメリカ国立標準技術研究所)の規格に基づいたOpenSCAPや、以前OSSだったNessusを源流とするOpenVASがあります。ただし、クラウドは基本的に他の顧客との共有リソースですので、他の顧客への影響を考慮する必要があります。従って、顧客自身によるこれらのOSSのインストールや使用は、クラウド・ベンダーとの契約上の許可や技術上の制約の確認が必要です。むしろ、クラウド・ベンダーがこれらのOSSと同等の機能を提供している場合が多いので、それらの利用を先に検討すべきでしょう。

また、異色なものとして、Netflix社が開発し、後にOSS化されたChaos Monkey[7]があります。これは自社システムをランダムにダウンさせることで、インターネットからの攻撃を模倣し、冗長機能を確証できます。

4-3. 暗号化の徹底

以前から、暗号化通信を提供するOSSとしてOpenSSL、OpenSSH、OpenVPNが利用されてきました。クラウド環境では、顧客が物理的にアクセスできない第三者にデータを預ける、ということを考慮しておく必要があります。そのため、データのライフサイクルを通じた暗号化の徹底が極めて重要です。暗号化で利用される鍵の管理にはHashiCorp Vaultが利用できます。

4-4. その他の考慮点

OSSだけでは、セキュリティー・リスクのすべてをカバーできません。不要となった機密データの確実な抹消や、昨今人気のコンテナ技術[8]、そこで稼働するアプリケーションで用いられるAPIなどは、それ特有のセキュリティーの考慮が必要です。また、顧客とクラウド・ベンダー間の責任分界点や、国や地域、業界特有のプライバシー関連の法令や監査なども考慮した契約締結が重要です。

5. コンテナ基盤の特性と監視

本章では、コンテナ基盤における監視について掘り下げます。

従来型のサーバー中心のシステムでは、監視対象となるノードをあらかじめ決定することが可能でした。監視対象にエージェント・プログラムを導入し、監視サーバー側ではエージェントからの通信を受信することで、監視を実現するのが一般的な構成です。しかし、コンテナ基盤の監視を考えるとき、主に以下の2つの特性が、従来型の監視の仕組みに適合しません。

1つは、コンテナが停止し消滅するとともにコンテナ内部に保持されているデータも消去されるという特性です。アプリケーションやミドルウェアのログも例外ではありません。どのような処理を行っていたのかをログから確認するためには、コンテナが停止する前に、継続的に外部へログを保管することが必要です(コンテナの異常停止による消失も考慮する)。

もう1つは、監視の対象となるコンテナの数と、そのコンテナが稼働するノード(サーバー)は、あらかじめ決定できないという特性です。コンテナ基盤上では、処理の負荷が高くなれば、起動されるコンテナの数を増加させるポリシーを定義するのが一般的です。また、リソースに空きのあるノードに自動的に配置されます。

これら2つのコンテナ基盤の環境特性に対応した、監視のためのOSSが登場しています。本稿では、モニタリングのためのOSSとしてPrometheus、Grafanaを用いた構成、ログ管理のためのOSSとしてElasticsearch、Fluentd、Kibanaを用いた構成を、それぞれ紹介します。

5-1. モニタリング

本節で取り上げるモニタリングとは、「CPUやメモリー等のリソースの利用状況の継続的な記録を行い、閾値を超えたときに発報する仕組み」のことを指します。PrometheusとGrafanaは採用事例が多く、事実上の標準となっているOSSです。**図3**のような構成にするのが一般的です[9][10]。

①Prometheus Server:監視サーバー本体
②監視対象:PrometheusやKubernetesが提供するAPIやユーザーAPIによる情報提供
③Alertmanager:アラート通知を行うPrometheusのコンポーネント
④Service Discovery:監視すべきサービスを発見・登録できる仕組み(DNS等を用いることもある)
⑤Grafana:監視結果をグラフィカルに表示するダッシュボード

従来型の監視と大きく異なる点として、2つ挙げることができます。

1つは、②の監視対象に対して、①のPrometheus Serverが、定期的かつ能動的にリソース状況の照会を行う点です。照会により得られた「現在のCPU利用率ー30%」のようなメトリクスと呼ばれるデータを、①内の時系列データベース内に蓄積します。これにより②の監視対象が消滅しても、モニタリング情報を保持することが可能になります。

もう1つは、④のService Discovery機能を用いて、監視すべきサービスを自動的に発見し、自らの監視対象に加える点です。エージェント型の場合、監視する対象はあらかじめサーバーに設定されるのが一般的です。一方Prometheusでは、定義された条件に基づいて、監視対象を動的に追加・削除します。

5-2. ログ管理

一般的なログ管理を考えるとき、以下の3つのステップを踏む必要があります。それぞれを実施するOSSを見てみましょう。

①ログを収集する

Fluentdは、データの収集を担うOSSです。さまざまな場所、さまざまな形式で発生するデータを、一元的なフォーマットで収集することが可能です。コンテナ基盤ではログの収集目的で利用され、多くの実績があります。プラグインを用いることで、さまざまなログの発生源、ログの送信先を指定可能です。

②ログを集積する

Elasticsearchは、オランダのElastic社が開発を主導している分散処理型の全文検索エンジンを提供するOSSです。Elasticsearchが提供する全文検索データベースに対して、Fluentdによって集約される多彩なログを格納します。コンテナ数が増えた場合、そのログは多岐にわたり、容量は膨大になります。Elasticsearchの検索機能を用いて参照・利用することで、必要なログ情報を抽出することが可能になります。

③ログの傾向を見やすく表示する（解析する）

Kibanaは、前述のElastic社が開発を主導しているOSSで、データの傾向を見やすく表示し、容易に解析を行う機能を提供します。中でもElasticsearchのデータを可視化することに長けており、ログの抽出だけでなく、グラフ化や定例的に利用するダッシュボード作成の機能を持っています。

5-3. Red Hat OpenShift Container Platformでの監視

統合的なコンテナ稼働環境を提供する基盤として、「Red Hat OpenShift Container Platform」（以下、OpenShift）が注目されています。OpenShiftの内部では、これまで説明してきたOSSを組み合わせて、システムの監視を行っています（**図4**）。

コンテナ基盤を導入する際には、既存の監視システム・監視運用との接続についても慎重に考える必要があります。OpenShiftの場合、例えば①Webhookと呼ばれる、http、TLS/SSLを用いた同期通信による既存システムへの通知、②Fluentdからのログのsyslog転送、などの方法が考えられます。既存システムとの接続性も考慮し

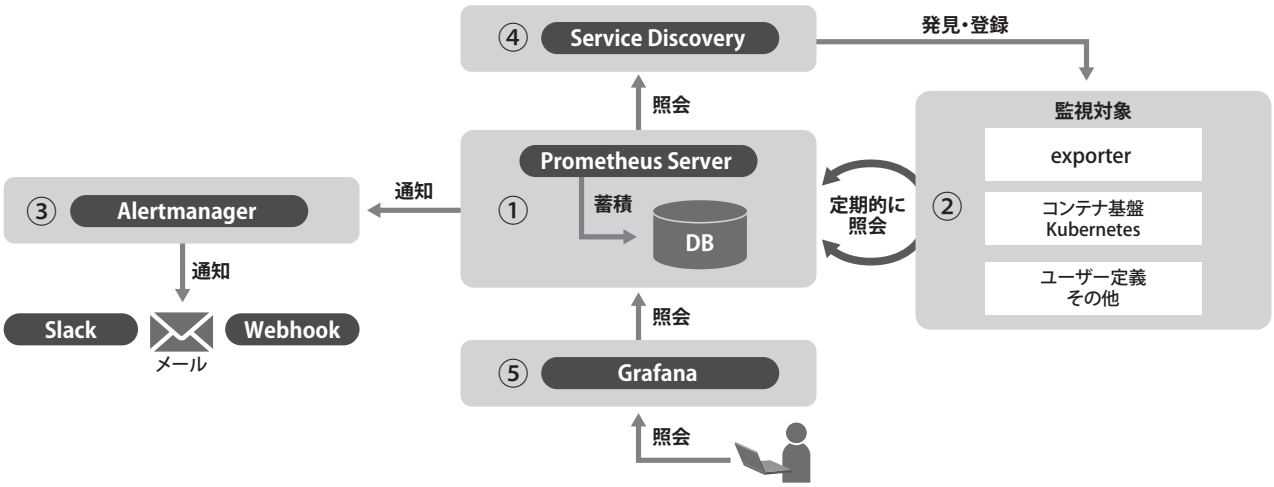

図3. Prometheusを利用したモニタリング・システムの構成例

た設計をする必要があります。

6. おわりに

OSSを活用したクラウドの運用管理として、インフラ運用の自動化、セキュリティー対策の実装、クラウドネイティブ環境の監視やロギングについて説明してきました。OSS利用による低コストや可搬性といったメリットを享受できる一方で、商用ソフトウェアと同等のサポートを受けられないといったデメリットが存在することも事実です。運用管理を検討する上でも、そのシステムで優先される要件に従って、適切なソフトウェアやツールを選択していくことが重要です。

[参考文献]

[1] IDC Japan：国内パブリッククラウドサービス市場 売上額予測、2018年～2023年 https://www.idc.com/getdoc.jsp?containerId=prJPJ44928319

[2] IDC Japan:2019年 国内Dockerコンテナ／Kubernetesに関するユーザー導入調査 https://www.idc.com/getdoc.jsp?containerId=prJPJ45328619

[3] 経済産業省：「DXレポート～ITシステム年の崖」克服とDXの本格的な展開～」http://www.meti.go.jp/shingikai/mono_info_service/digital_transformation/20180907_report.html

[4] 一般社団法人日本クラウドセキュリティアライアンス（CSA ジャパン）クラウドセキュリティワーキンググループ
：CSAガイダンスversion4.0を用いたクラウドセキュリティリファレンス（OSSマッピング2019），https://www.cloudsecurityalliance.jp/site/WG_PUB/cloudsecurity_WG/CSAguidance_mapping_20190226.pdf

[5] Ben Mailsow：CCSP(ISC)2 Certified Cloud Security Professional Official Study Guide(2019).

[6] Center for Internet Security : CIS Benchmarks , https://www.cisecurity.org/cis-benchmarks/

[7] Netflix, Inc. : Chaos Monkey , https://netflix.github.io/chaosmonkey/

[8] Murugiah Souppaya, John Morello, Karen Scarfone : NIST Special Publication 800-190 Application Container Security Guide by NIST (2017), https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.800-190.pdf

[9] Brian Brazil : An Introduction to Prometheus, Open Source Forum, The Linux Foundation(Nov.15,2016)
http://events17.linuxfoundation.org/events/archive/2016/open-source-forum

[10] Brian Brazil : Prometheus: Up & Running: Infrastructure and Application Performance Monitoring, O'Reilly Media(2018)

日本アイ・ビー・エム株式会社
グローバル・テクノロジー・サービス事業　クラウド・ネイティブ・プラットフォーム・サービス
エグゼクティブ・アーキテクト

青山 真巳
Aoyama Manami

1999年日本IBM入社。流通・製造業のアウトソーシングのお客様システムの設計・構築・運用を経て、クラウド・サービスとオートメーション・サービスの企画・開発・設計に従事。2019年よりクラウドネイティブのソリューション検討・提案を担当。

日本アイ・ビー・エム株式会社
グローバル・テクノロジー・サービス事業　IGAサービスマネジメント
シニア・アーキテクト

河合 淳一
Kawai Junichi

1995年日本IBM入社。同社社内システムの設計、保守に長く従事、社内で培われたノウハウや事例の紹介も行う。2010年頃からセキュリティーにフォーカス。セキュリティーの上位資格であるCISSPを保持。

日本アイ・ビー・エム株式会社
グローバル・テクノロジー・サービス事業　デリバリー&トランスフォーメーション
金融・保険サービス・デリバリー セクターテクニカル推進
アドバイザリー・アーキテクト

舘田 潤
Tateda Jun

2003年日本IBM入社。関西の中堅企業のお客様担当SEを歴任。2012年人材交流の一環でJR九州に出向。鉄道安全推進・サービス企画に従事。帰任後、OSSやコンテナ技術に関心を持ち、同分野のシステム設計・構築を担当。

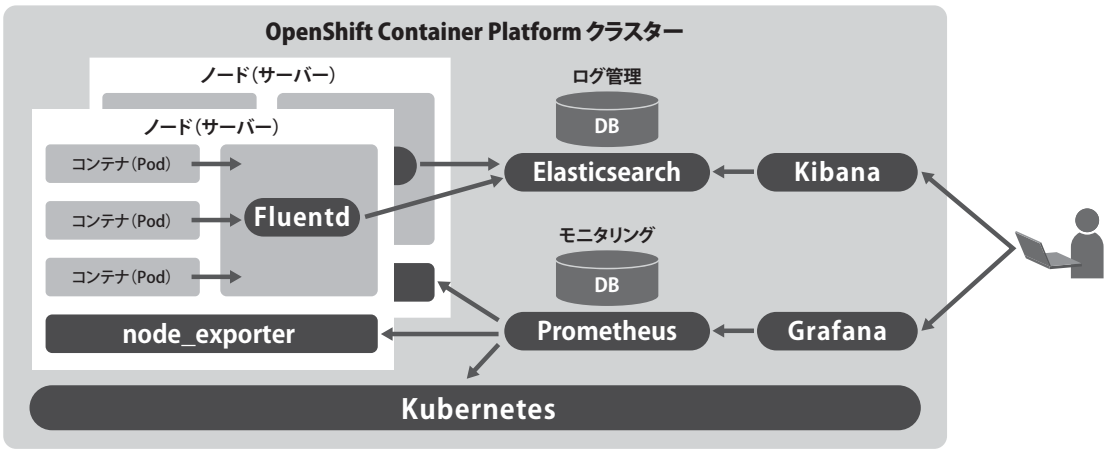

図4. OpenShiftにおけるシステム監視とOSS