JFE 技報 No. 51
（2023 年 2 月）p. 6–10

プライベートクラウド基盤の構築

Building Private Cloud Foundation

小林　健一　KOBAYASHI Kenichi　　JFE スチール　IT 改革推進部　主任部員（課長）
久米　正洋　KUME Masahiro　　JFE スチール　IT 改革推進部　主任部員（課長）

要旨

JFE スチールでは DX 戦略の一環である IT 構造改革を断行するため，本社基幹システムをはじめとするオープン化プロジェクトを推進している。これらを完遂するには，従来のホストコンピュータに替わる全社的なシステム基盤が必要不可欠である。基幹システムを支える高い可用性・継続性，コスト最適性などを備える基盤として，JFE グループ向けのオープン・プラットフォーム（プライベートクラウド※）を構築した。本稿では，プライベートクラウドに採用した主要技術の概要・狙い，および実運用を踏まえた今後の展望について論ずる。
※プライベートクラウド：企業が自社内でクラウド環境を構築し，社内部署やグループ会社に提供するサービス形態

Abstract:

As part of DX strategy, JFE Steel has been promoting a project to open, including the core systems of its headquarters, to carry out IT structural reforms. To complete these projects, a company-wide system infrastructure to replace the conventional host computer is essential. We built an open platform (private cloud*) for the JFE Group as a foundation with high availability, continuity, and cost optimality to support core systems. In this paper, we discuss the outline and aim of the key technologies adopted for the private cloud and the future prospects based on the actual operation.

*Private cloud: A form of service in which a company builds its own cloud environment and provides it to internal organizations and group companies

1.　はじめに

JFE スチールの基幹システムは，高い信頼性を持つホストコンピュータ上で稼働してきた。ホストコンピュータは各メーカーの独自規格で構築されるため，システム構成の選択肢や機能の拡張性に乏しく先端デジタル技術の活用が難しいなど，データ資産活用における課題を抱えていた。DX を推進して「変化に強い柔軟な IT 構造」を実現するためには，老朽化したレガシーシステム群の統合，オープン・プラットフォームへの移行のための基盤構築が肝要であった。構築する基盤は，技術動向・システム特性などを考慮し，下記要件を満たすプライベートクラウド（以下 J-OSCloud）とした。

① 標準化・ベンダロックイン回避による価格合理化
② 自動化による新規サービス提供リードタイムの短縮
③ 可用性・事業継続性の担保

本稿では J-OSCloud 構築で採用した下記 2 点の主要技術について述べる。

① OpenStack 技術
OSS（オープンソースソフトウェア）である OpenStack 採用によるベンダロックイン回避およびサーバ構築自動化
② ネットワーク仮想化技術
東西のデータセンターを論理的に 1 つに活用かつ冗長構成とすることによる可用性・事業継続性の担保

2022 年 9 月 13 日受付

2.　プライベートクラウド概要

J-OSCloud のシステム概要を**図 1** に示す。東日本エリアのデータセンター（以下，東 DC）および西日本エリアのデータセンター（以下，西 DC）へ各種機器を設置している。東西それぞれで，OpenStack 技術により仮想サーバを構成し，プライベートクラウドを構築している。すべての機器（サーバ，ストレージ，ネットワーク（以下 NW）スイッチなど）は，有事に備えそれぞれのデータセンター内で冗長化されている。また，東西のデータセンター間は仮想 NW 技術により，同一 NW セグメントで管理されている。データセンター自体でも，Tier4 要件相当を目指し，24 時間 365 日の基幹システム運用を支える整備を進めている。

J-OSCloud 上は社内の個別サーバ（オンプレミス）との連携のみならず，パブリッククラウドと連携したハイブリットクラウドとしての活用も可能とした。

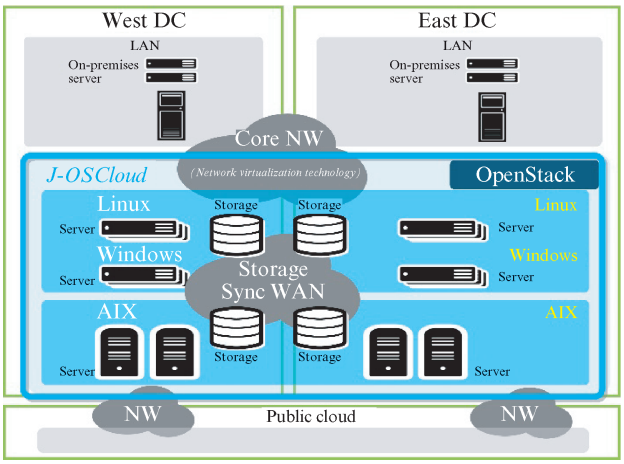

図 1　J-OSCloud システム概要

Fig. 1　J-OSCloud system overview

3.　採用技術の概要と狙い

3.1　OpenStack 技術の概要および狙い

3.1.1　OpenStack とは

OpenStack とは 2010 年に Rackspace 社と NASA によって始められたオープンソースソフトウェアの IaaS（Infrastructure as a Service）クラウド基盤管理スタックである。KVM や Xen，VMware ESX，Hyper-V 等のハイパーバイザ（仮想サーバ構築のソフトウェア）と組み合わせて IaaS やストレージサービスを提供するための管理機能を提供している。開発やライセンスの管理は 2012 年に全て非営利団体である OpenStack Foundation に移管され，特定ベンダーの技術に偏らないオープンな開発が可能となった。**図 2** に OpenStack の歴史を示す。当初は小規模からはじまり，年々機能の高度化，細分化を実施し進化している。J-OSCloud は Kilo を採用した。

表 1 に OpenStack を構成するコンポーネントを記す。J-OSCloud 内では，Nova, Glance, Cinder, Keystone, Horizon を使用し，J-OSCloud を構築している。Neutron（仮想 NW 管理）も，一部機能のみ使用している。

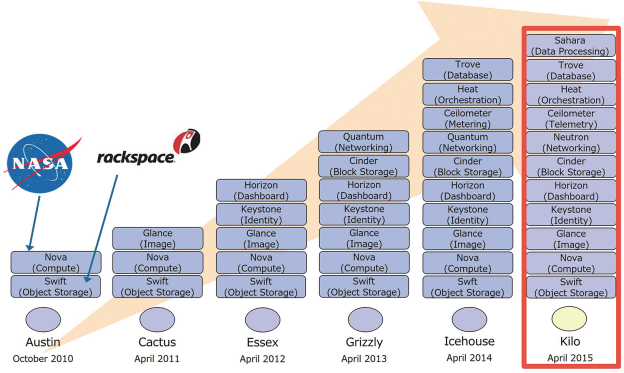

図 2　OpenStack の歴史

Fig. 2　History of OpenStack

表 1　OpenStack を構成するコンポーネント

Table 1　Components that make up OpenStack

Project	Service	Overview
Nova	Compute	Determining virtual machine placement, starting and stopping
Swift	Object storage	Object storage
Glance	Image	Managing virtual machine images
Neutron	Networking	Virtual network management
Cinder	Block storage	Providing block volumes
Keystone	Identity	Integrated authentication
Horizon	Dashboard	Web user interface
Heat	Orchestration	Orchestration (AWS cloud formation compatible)
Ceilometer	Telemetry	Collecting information on which to charge
Trove	Database	Database as a service
Sahara	Data processing	Hadoop duster provisioning

3.1.2　OpenStack によるサーバ構築自動化

OpenStack によりサーバ構築を自動化したことで，従来数週間～1 か月かかっていた構築作業を，約 3 日間に短縮した。構築自動化の前準備として，サーバ構成を標準化した。具体的には，OS は 3 つ（Windows，Linux，AIX）に限定し，OS のディスクサイズ，OS のファイル構成などを統一し，セキュリティ設定をテンプレート化した。

図 3 にサーバ構築自動化の流れを示す。表 1 にある OpenStack の各コンポーネント，およびテンプレート毎に準備したスクリプトを駆使して，構築依頼内容（CPU, MEMORY などのスペック要求，オプション設定）に応じたサーバ構築を実現している。

3.2　ネットワーク仮想化技術の概要および狙い

3.2.1　ネットワーク仮想化技術の概要および狙い

東西 DC を論理的に一つの DC として活用する場合，拠点を跨いだシステム冗長構成（拠点間ライブマイグレーション）を実現させる必要がある。実現方法はいくつかあるが，柔軟性があり，運用負荷が高くない方式を検討した。具体的には拠点間ライブマイグレーション時に，IP アドレスの変更作業や DNS サーバの登録変更操作を不要とする，ネットワーク仮想化技術の適用により解決を図った。当時，日本では当該技術の適用事例がなく，世界的に見ても事例が僅少で導入難易度が高かったが，将来の IT インフラ構想の実現に向け多くの技術検証を重ねて導入に至った。以下に，具体的な技術要素を紹介する。

3.2.2　拠点間ネットワークセグメント延伸技術

拠点間ライブマイグレーションとは，稼働中の仮想サーバを，OS やアプリケーションを停止させることなく，そのまま別拠点の物理サーバへ移動させることである。ネット

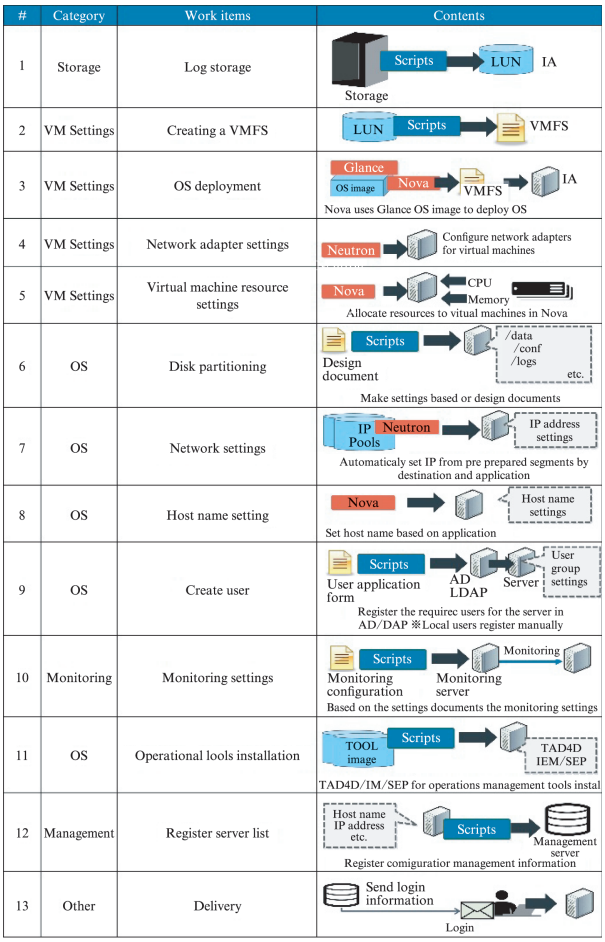

図 3　サーバ構築自動化の流れ

Fig. 3　Flow of server construction automation

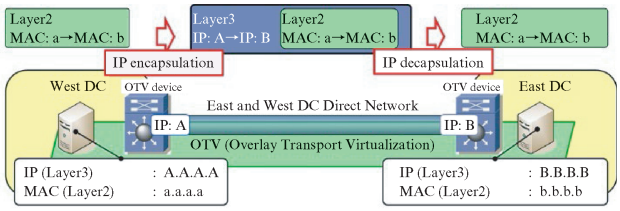

図 4　拠点間ネットワークセグメント延伸技術

Fig. 4　Inter-site network segment extension technology

	IP address (ID)			
	Network address (Locator)			Host
・IP address	1 .	1 .	1 .	1
・Subnet mask	255 .	255 .	255 .	0

図 5　IP アドレスの形式

Fig. 5　IP address format

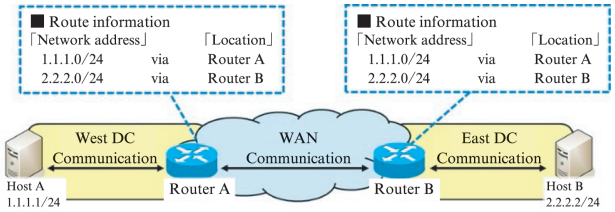

図 6　従来のネットワーク経路制御技術

Fig. 6　Conventional network routing technology

ワークの瞬断は発生するものの，仮想サーバの利用者（クライアント）は，仮想サーバの移動を意識することなく利用継続できる。

DNS サーバの登録変更操作を行わず，拠点間ライブマイグレーションを行うためには，移動先の物理サーバが移動前の物理サーバと同一の Layer2 ネットワークに接続されることが前提となる。Layer2 ネットワークとは，OSI 参照モデルの第 2 層における「データリンク層」を指す。データリンク層での通信は，接続されている機器の MAC アドレス情報を学習することで行われる。異なった Layer2 ネットワークに接続された物理サーバへ仮想サーバが移動すると，ネットワークアドレスが異なるため，仮想サーバ側の IP アドレスを変更して起動する形となる。その結果，利用者（クライアント）側からの通信を継続できなくなる。

拠点間ネットワークセグメント延伸技術として，Overlay Transport Virtualization（OTV）を採用した。OTV を利用することで，同一ネットワークアドレスを持つ，別拠点のシステムと通信できるようになる。同一のネットワークアドレスを保持するシステム同士の通信は，Layer2 通信を行う。一方で，別拠点の相手と通信を行う場合は，Layer3 の経路制御が必要となる。OTV では，**図 4** に示すとおり Layer2 のフレームを Layer3 の IP でカプセル化により隠蔽することで，通信を実現させている。

3.2.3　網内通信経路制御

経路制御技術に関する説明の前に，IP アドレスの形式を**図 5** に示す。IP アドレスは，32 bit 全体で識別情報（ID）＝ホストアドレス情報を意味し，ネットワークアドレス部が位置情報（Locator）を意味する。Locator が ID の一部であるという点が重要である。

OTV により，拠点間ネットワークセグメントを延伸できたが，延伸された広いネットワーク上で適切に通信相手先を特定しなければ通信は成立しない。通常のネットワーク基盤の制約として，同一ネットワークアドレスが複数拠点に存在するという状態を構成させることはできない。

図 6 に示すとおり，各ルータは，宛先のネットワークアドレスへはどのルータへ届ければ良いかを，テーブル情報として保持している。各ルータが経路情報としてネットワーク部単位（ID の一部）で位置情報（Locator）を管理している。その結果，同一のネットワークアドレスのままでは，拠点間の移動が不可となる。

本課題の解決にあたり，LISP[1)]（Locator/ID Separation Protocol）を適用した。LISP とは，IP アドレスが持つ“位置情報（Locator）”と“識別情報（ID）”を分離するルー

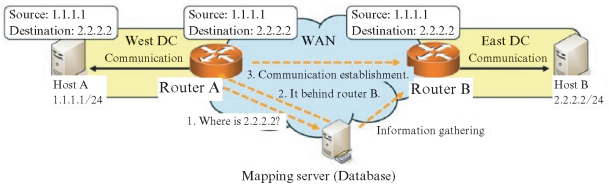

図 7　仮想ネットワーク（LISP）の経路制御技術
Fig. 7　Virtual network (LISP) routing technology

ティングアーキテクチャーである。LISP では，IP アドレスに対応する位置情報（Locator）をマッピングサーバと言われるデータベースに保持する。

LISP を利用した場合の経路制御は，ネットワークアドレスに対応するルータとの紐づけ管理は行わない。**図 7** に示すとおり，ルータは通信先情報（IP アドレス）をキーにデータベースに問い合わせ，データベースが適切なルータ情報を返却することで経路制御を実現させている。

適用技術要素をまとめると次のとおりである。東 DC〜西 DC 間へ Layer2 延伸技術である OTV を適用し，東西 DC 間の仮想サーバの移動を LISP VM Mobility 機能で検知させる。利用者（クライアント）側から各システムへの通信では，LISP を適用することで，ホストアドレス単位の経路制御を実現している。

4.　継続的な機能強化

J-OSCloud は 2016 年に構築し，2022 年 6 月時点では本社基幹システムサーバといった重要システムを含む，東西合わせて数百台のサーバが稼働している。この間，J-OSCloud は継続的に改善しており，その中で代表的なものを紹介する。

4.1　災害対策サイト立上げ自動化

J-OSCloud では，数百台のサーバが稼働しており，有事の際の災害対策サイトでのシステム立上げには，相当な時間がかかることが想定された。また，順次立上げる場合のサーバ毎の優先順位も考慮する必要があった。そこで，ストレージ，OS，ミドルウェア（MW）の立上げ時間短縮のため自動化を試みた。**図 8** が自動化範囲であり，ここでも標準化を意識してオープンソースソフトウェアである Ansible を採用した。Ansible は，プログラミングレス，エージェントレス（インストール不要）として広く普及しているツールであり，J-OSCloud でも，運用管理の自動化など複数ケースで積極的に活用した。NW，管理機能といった J-OSCloud の機能ごとの起動順，およびサーバごとの起動順を考慮して設計し，各構成要素の立上げをスクリプトで記述することで自動立上げを実現した。

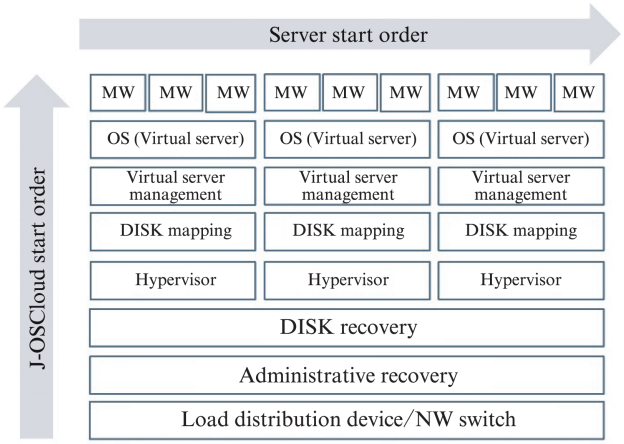

図 8　自動化構築範囲
Fig. 8　Scope of automation building

4.2　ストレージ高可用性対応

近年，ハードウェア機器不具合に伴う障害事案が発生していることから，JFE スチールでも各機器を再点検し，物理障害に対する対障害性を高める施策を実施した。一例として，ストレージ機器の完全冗長化および冗長切替の自動化がある。**図 9** にストレージ高可用性対応として検討した内容を記す。それぞれのシステムにおける障害時の復旧時間，データ鮮度（復旧ポイント）に応じ，対応可能な機能を構築した。

5.　おわりに

以上のとおり，OpenStack 技術およびネットワーク仮想化技術を採用したプライベートクラウドを構築し，当社の重要システムの稼働基盤を整備した。その後も継続して機能強

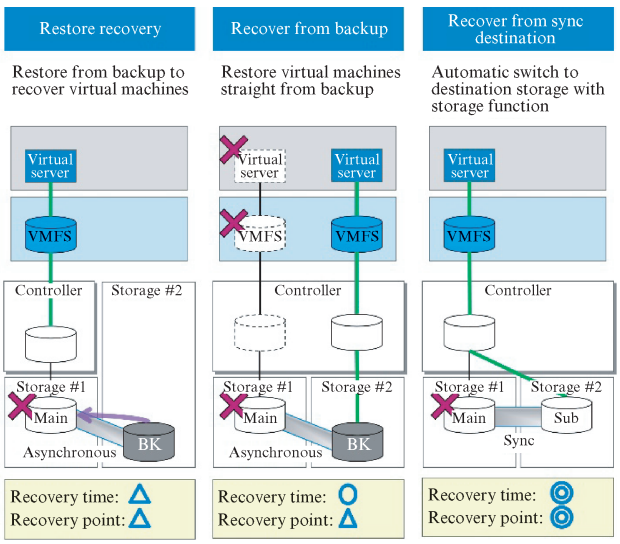

図 9　ストレージ高可用性対応
Fig. 9　Enabled storage high availability

化を重ねており，可用性，事業継続性といったさらなるサービスレベル向上に努めている。

一方，当基盤構築の2016年当時と比較して，AWS（Amazon Web Service）をはじめとするパブリッククラウド※の技術進化は著しく，金融業，製造業をはじめとして重要システムの稼働事例も増えつつある。JFEスチールのシステム特性を十分に考慮した上で，パブリッククラウドを含めた連携強化や，プライベートクラウドへの応用を検討していきたい。

※パブリッククラウド：広く一般のユーザーや企業向けにクラウド環境をインターネット経由で提供するサービス形態

参考文献

1）Farinacci, D.; Fuller, V.; Meyer, D.; Lewis D. 2013. The Locator/ID Separation Protocol（LISP）. RFC 6830. http://www.rfc-editor.org/rfc/rfc6830.txt