OpenStackで実現する分散ストレージ「Swift」とプライベートクラウド

2013/02/22

日本ヒューレット・パッカード株式会社
テクノロジーコンサルティング統括本部
ソリューション開発本部　コアテクノロジー部
石田 精一郎

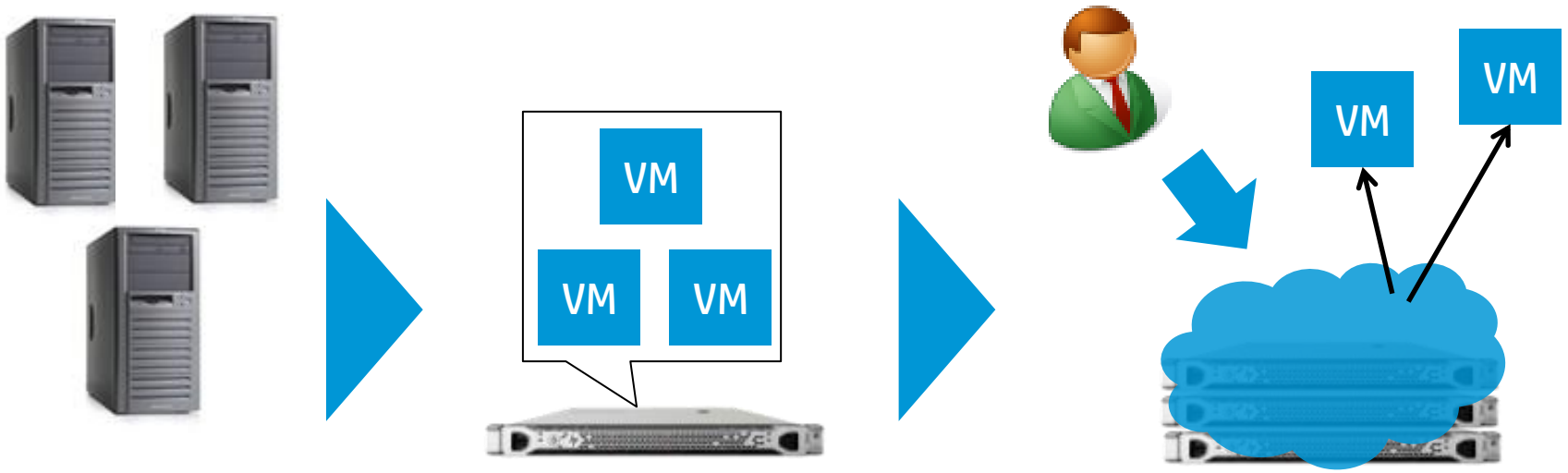
Agenda

- OpenStack概要
- Swift概要
- OpenStack/Swiftデモ
- Q & A

OpenStack概要

OpenStackが注目される背景
仮想化からクラウドへのIT基盤の進化

仮想化によるIT基盤統合 → IT基盤のクラウドサービス化



サイロ型IT基盤

IT基盤統合から標準化へ

IT基盤のクラウドサービス化

得られる効果

- リソース稼働率の向上
- 運用作業の標準化
- システムコストの最適化

- ITサービスを迅速に提供
- システム提供をサービスメニュー化
- セルフポータルの提供による管理業務の自動化

クラウドIT基盤を構築できるソフトウェアのニーズが拡大

クラウドIT基盤とは

- おおよそ「サービス化」＋「標準化」＋「自動化」
  - 実装手段として「仮想化」技術を利用することが多いが「仮想化」は必須ではない

サービス化
- 利用者はIT基盤の内部構造を意識しない
- 使いたいときに使いたい分を利用する
- 使い終わった後に資産、在庫として残らない

標準化
- 次のような条件を共通メニューとして揃える
  - ✓ マシンリソース要件(OSイメージ、CPU、メモリ、ストレージ、ネットワーク等)
  - ✓ 利用条件(SLA、セキュリティ等)
  - ✓ 申請方法、運用管理等のプロセス

自動化
- 利用申請やリソース払い出しなどの管理タスクをポータルやAPIで自動化

IT基盤の利用者のメリット

- オンデマンドで指定したスペックの仮想サーバやストレージをすぐに利用できる

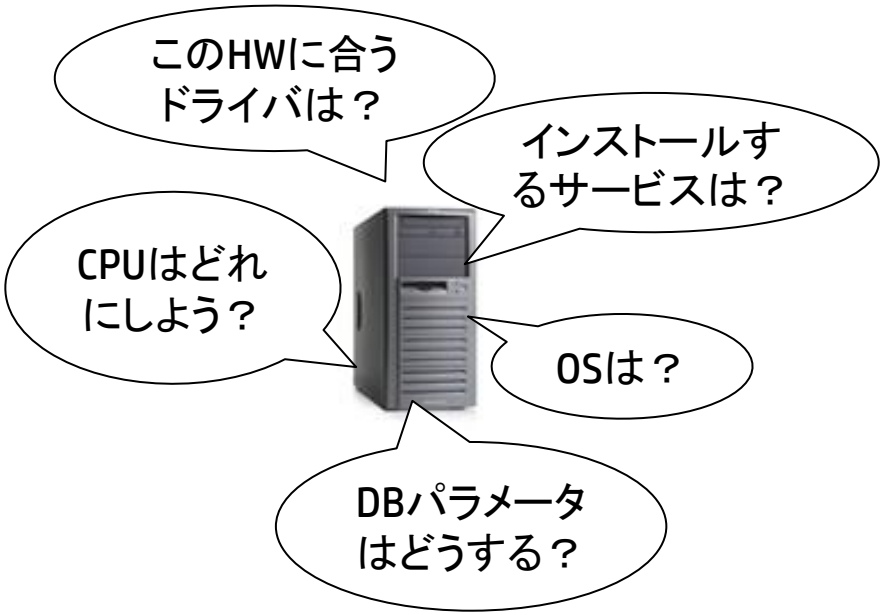
IT基盤の管理者のメリット

- 利用者ごとの個別対応が不要
- 運用の効率化と管理の向上
- ヘルプデスクの負荷軽減
- 統合によるコスト削減効果

オーダー・メイドからレディ・メイドへ

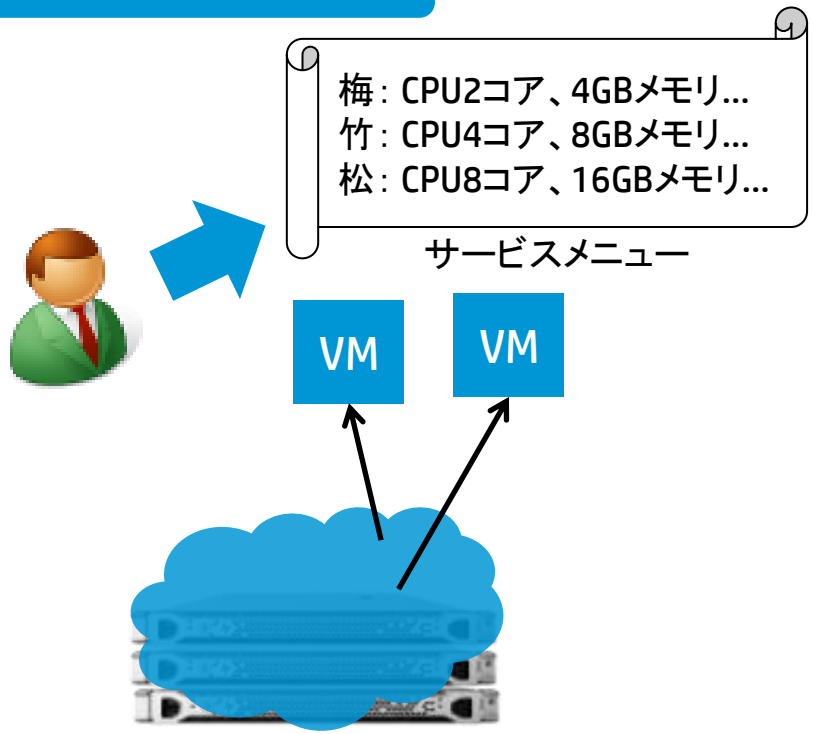
クラウドを利用するのは、「ユニクロ」で服を買うようなもの

HWから調達



HWのパーツからすべてをカスタマイズ可能。
自分のシステムにぴったり合うものを作れる。
ただし、コンポーネントごとの個別検討が必要。

クラウドサービス利用



事前に定められたメニューから構成を選択。
迅速にサーバを用意できるが、HWのカスタマイズの幅はあらかじめ決まっている。
サービスの裏には高度な自動化の仕組み。

OpenStackとは

OpenStackは誰でも使える、誰でも開発に参加できるクラウド基盤ソフトウェア

OpenStackとは

- クラウド基盤ソフトウェアを開発するOSSプロジェクト
  - 仕様は全てコミュニティの議論で決定される
- ITインフラのライフサイクルを管理
  - サーバ、ストレージ、ネットワークリソースの生成、割当、返却、再利用
  - APIによるハードウェアのソフトウェア化
  - ユーザ、グループの分離 (マルチテナント化)
- **クラウド基盤の標準を目指す**

運営体制

- 非営利団体であるOpenStack Foundationが運営
- HP、Redhat、SUSE、Canonical、AT&T、Cisco、IBM、DELL、RackSpace、NEC、Intel、VMware、EMC、Yahoo!などが参加
- Linux Foundationモデルに類似

OpenStack開発の経緯

NASA

Nebula
(IaaS基盤)

2009年
独自のクラウドプ
ラットフォームを
開発・運営

2010/7

OpenStack

RackSpace

Cloud Files
(ファイルホスティング)

2008年
独自のクラウドファイル
ホスティングサービスを
開発・運営

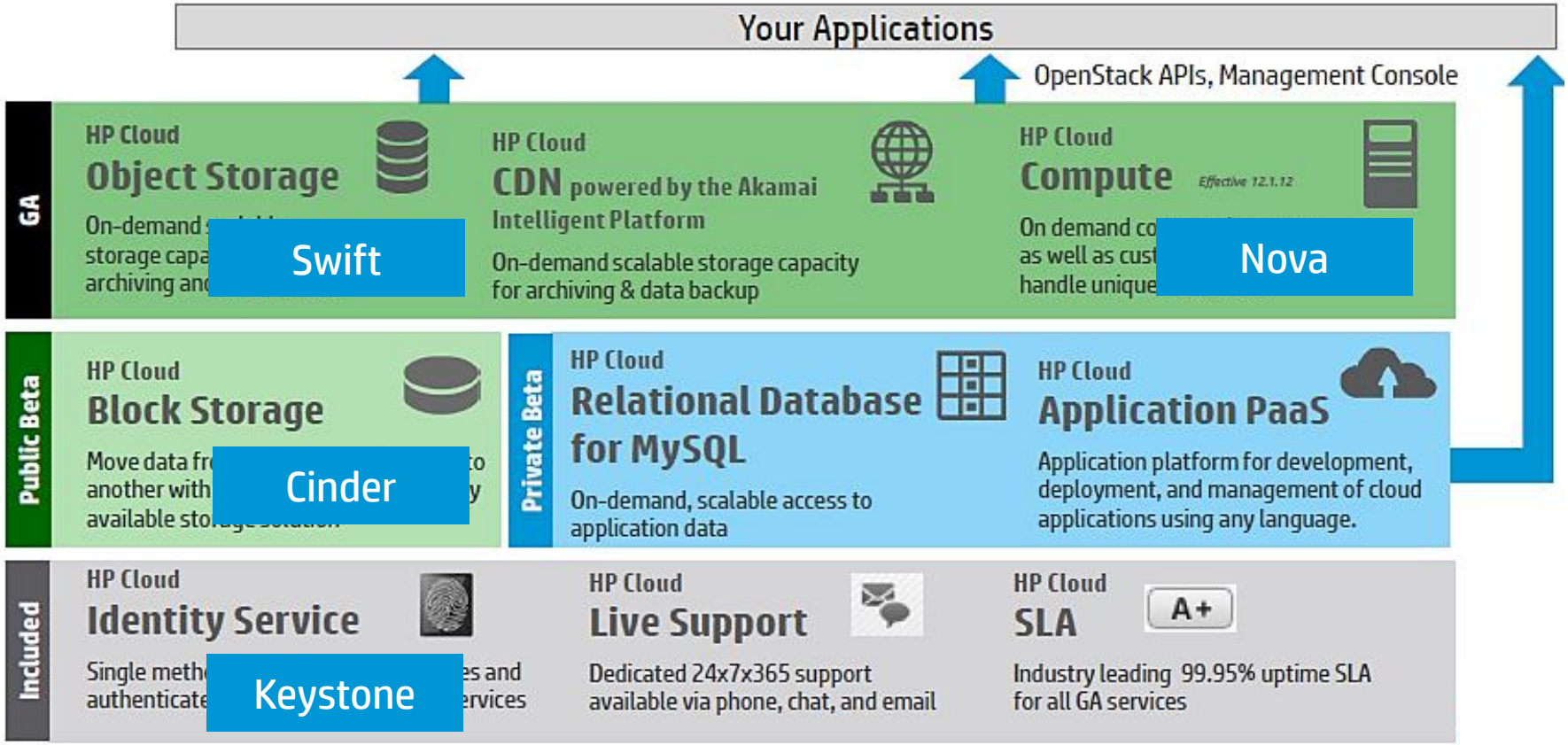
ロードマップ

Austin	Bexar	Cactus	Diablo	Essex	Folsom	Grizzly
2010/10	2011/2	2011/4	2011/10	2012/4	2012/9	2013/4

商用で活用されているOpenStack

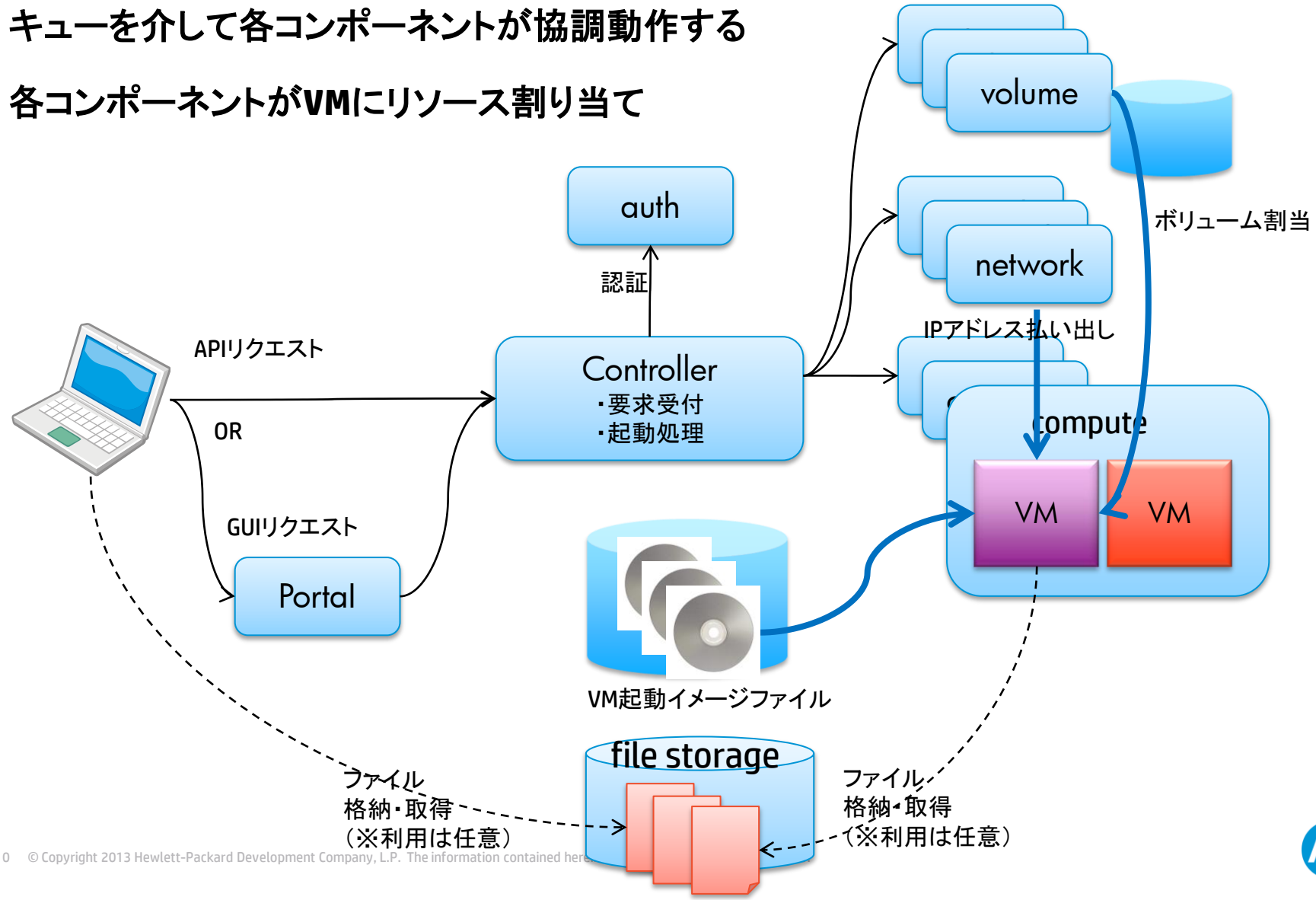
HPが提供しているパブリッククラウドのHP Cloud Services（www.hpcloud.com）にて利用

数千台の物理マシンとPbyteクラスストレージシステムが複数DCにて稼働



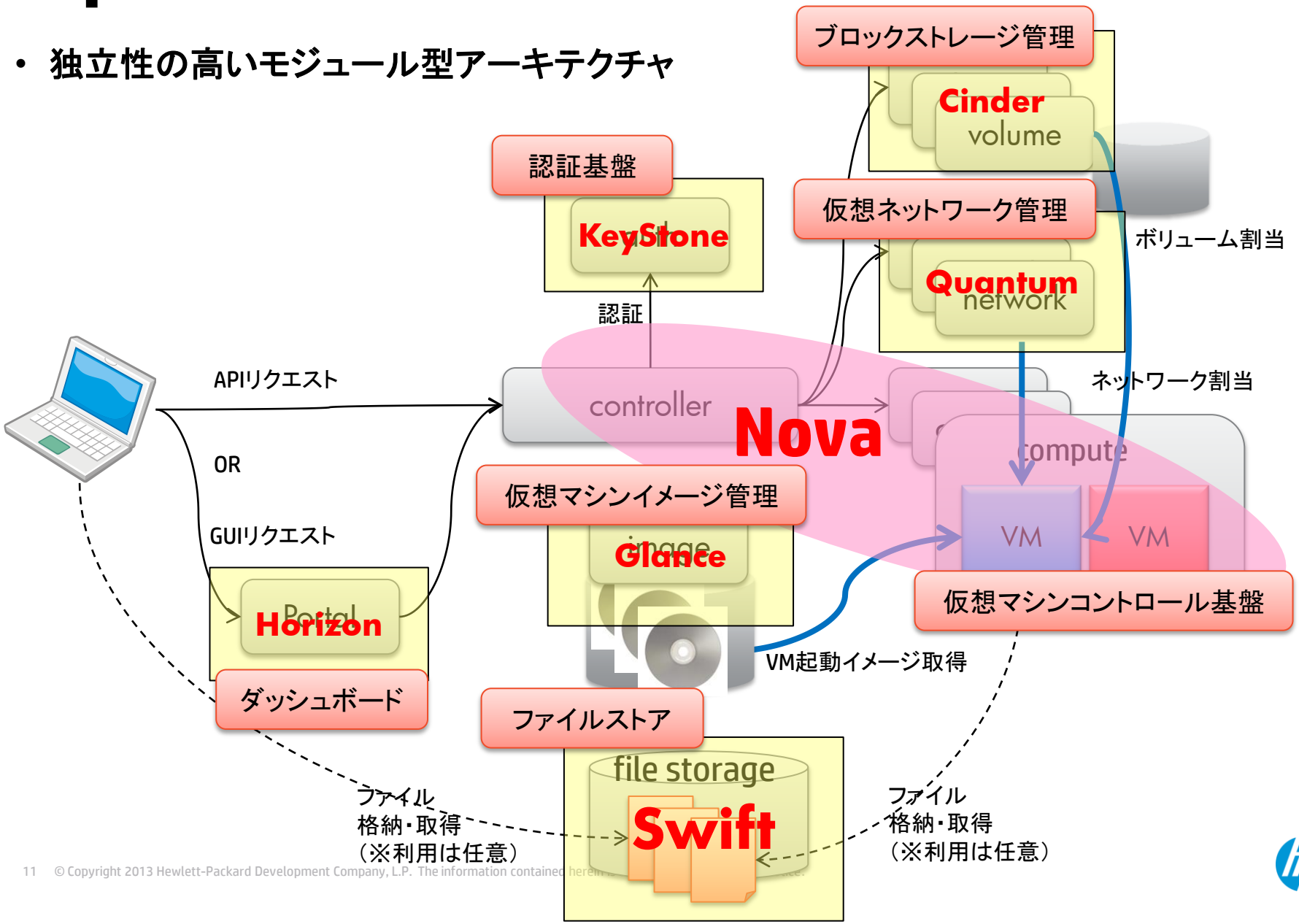
OpenStackのアーキテクチャ

- キューを介して各コンポーネントが協調動作する
- 各コンポーネントがVMにリソース割り当て



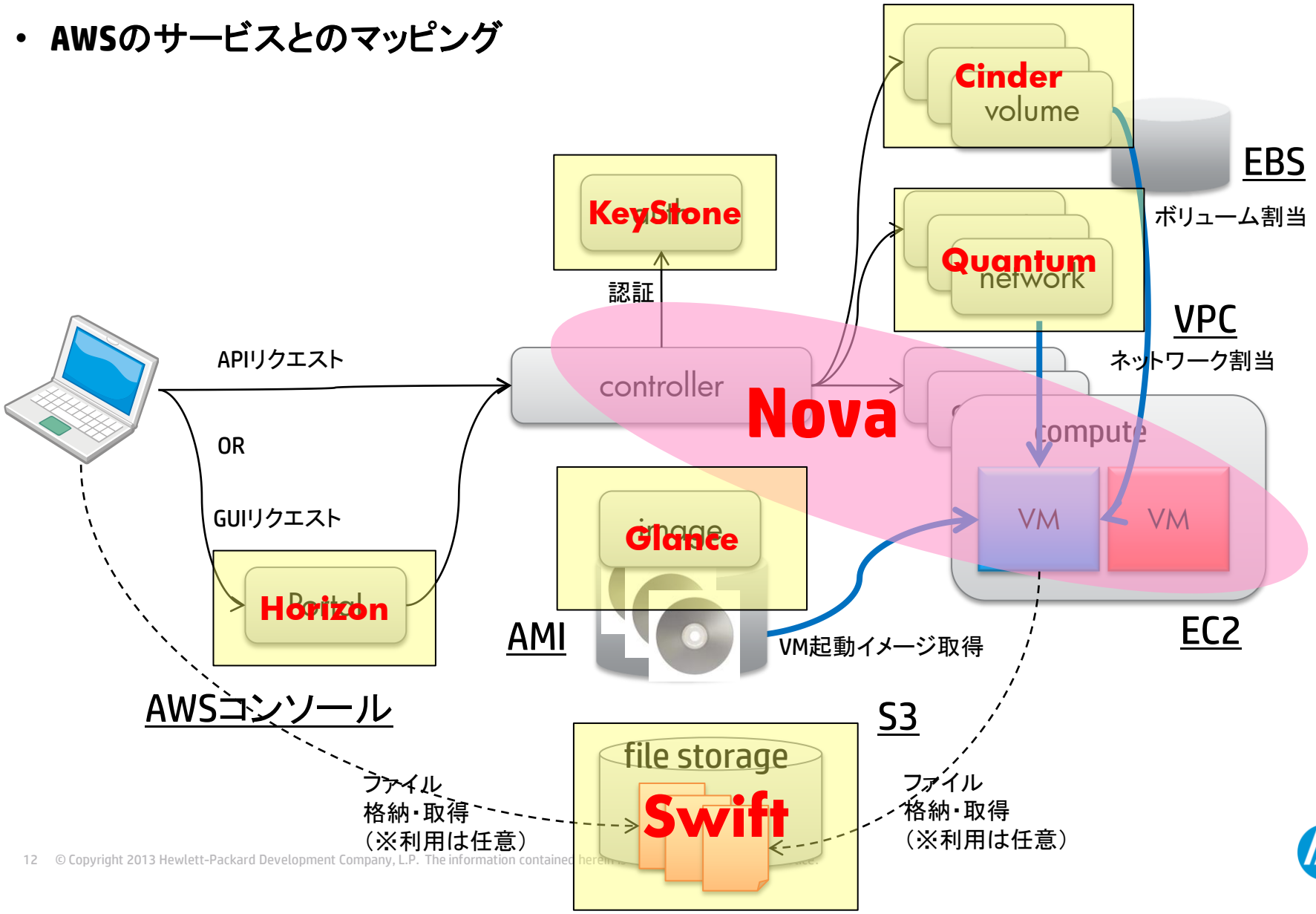
OpenStackのアーキテクチャ

- 独立性の高いモジュール型アーキテクチャ



Amazon Web Servicesとの対比

- AWSのサービスとのマッピング

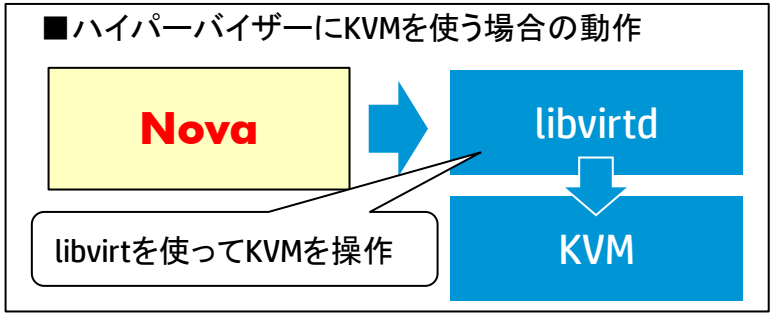


OpenStack コンポーネントの内部動作

OpenStackは、いろいろな素材を組み合わせてクラウド基盤を作っている

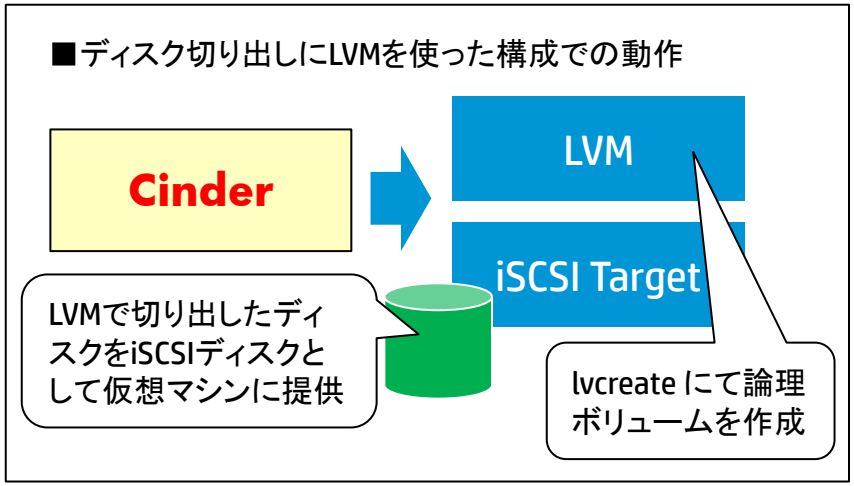
Nova

- APIを通じて、OpenStackの各コンポーネントを連携させる。
- KVM等のハイパーバイザーをコントロールし、仮想マシンを作成。
- KVMのコントロールには「libvirt」を使用。



Cinder

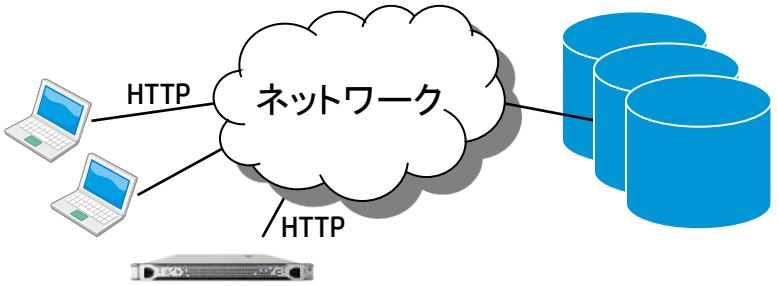
- iSCSIストレージにアクセスして、ボリュームを作成。
- 作成したボリュームをNovaが作成した仮想マシンに提供。
- LVMとiSCSI Target（tgt）を使ってiSCSIストレージを構築し、Cinderから利用することができる。



Swift概要

Swiftとは

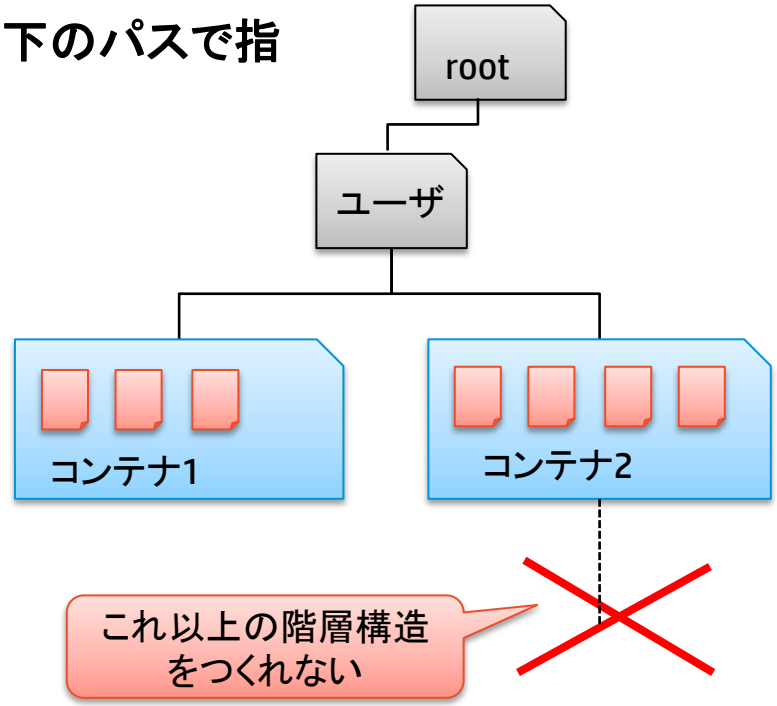
- **HTTPでアクセスする低コスト・高可用性の分散ファイルストレージ**
  - Amazon S3のような、ネットワーク上のファイル置き場
  - 仮想マシンイメージや写真、動画、バックアップファイル等の静的なファイルをオブジェクトとして管理・保存



- **Swiftの主な機能**
  - **REST API**
    - HTTP経由でファイル操作（アップロード、ダウンロード、削除、リストなど）を行います。
      - NFSやCIFSでのアクセスなど、NASのような使い方はできません。
  - **可用性**
    - 自動的にファイルの複製を3台のストレージ用ノード上に作成します。
  - **スケールアウト**
    - ストレージ用ノードの追加により、ストレージ容量を増やしていくことができます。

ファイル(オブジェクト)の論理配置の特徴

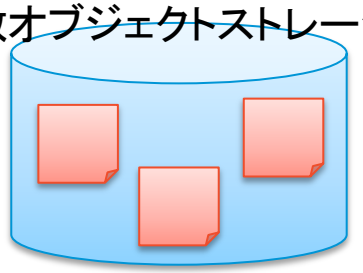
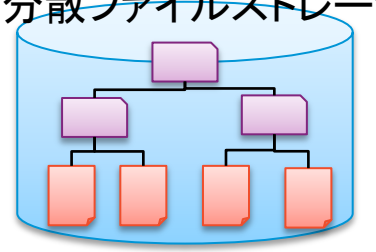
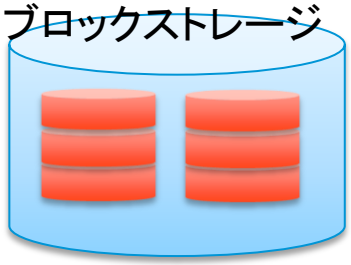
- **ファイル(オブジェクト)はユーザが作成する「コンテナ」の中に保存される。**
  - コンテナは階層構造にすることはできない。
- **保存するファイル(オブジェクト)の指定は以下のパスで指定する。**
  - /ユーザ/コンテナ名/オブジェクト名



※ このようなフラットな論理構造になっているのは、REST API でアクセスする際のURLを簡単にするための工夫

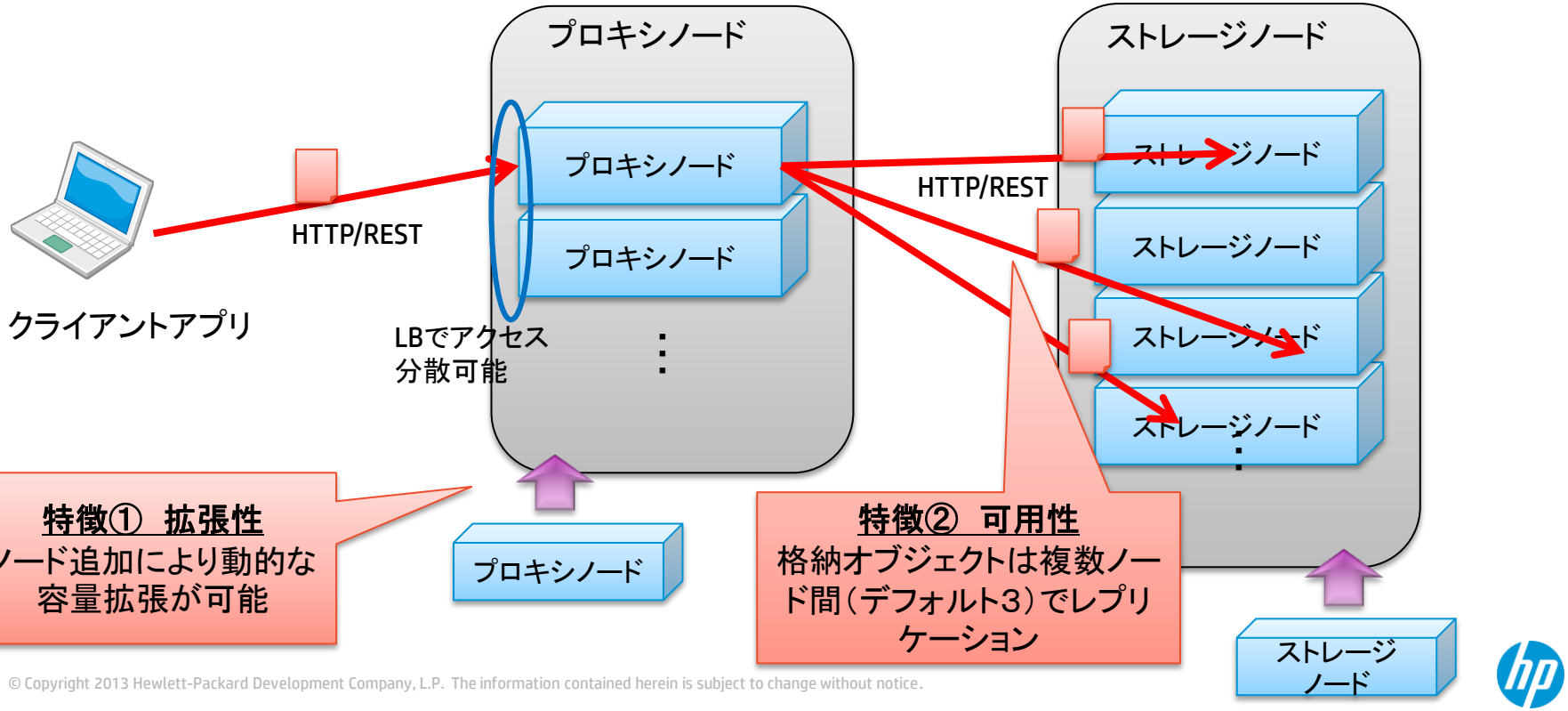
クラウドデータストア比較

- データストアの3つの方式の比較

	クライアント	アクセスプロトコル	保存・アクセス単位	特徴	製品
分散オブジェクトストレージ 	HTTP クライアント	HTTP/REST	ファイル	低コストで可用性が高いが、ファイル内の一部のデータ更新などランダムアクセス処理はできない	Swift(OpenStack) Amazon S3
分散ファイルストレージ 	OS	NFS/CIFS	ファイル	NASとして利用可能。ファイルシステム経由でPosix準拠のシステムコール(open(), read(),write()等)が実行可能	GlusterFS
ブロックストレージ 	OS	iSCSI	ブロック	クライアントとなるOSからローカルディスクをマウントするのと同じような操作が可能 ファイルシステムの種類はクライアント側で指定できる	Cinder(OpenStack)

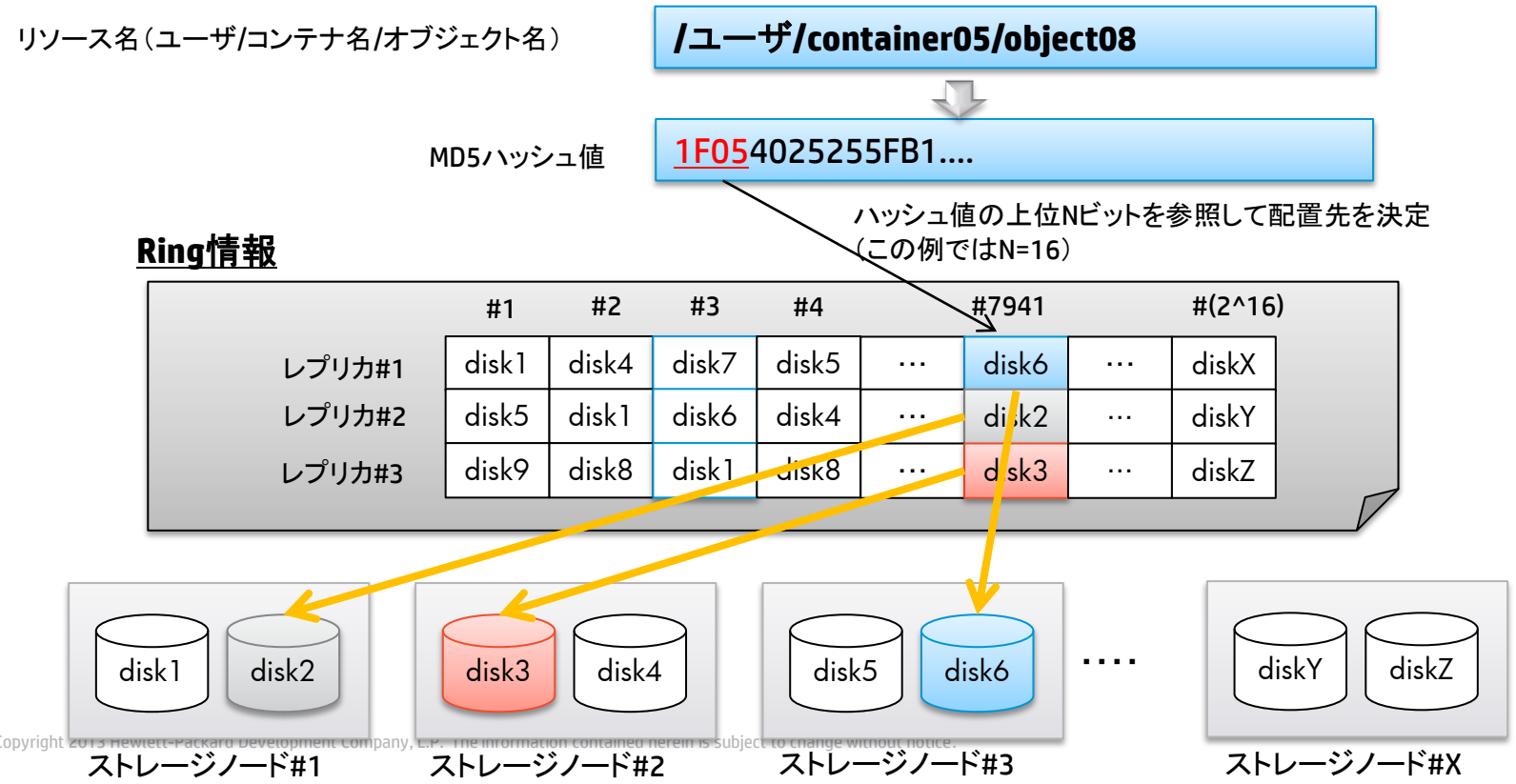
Swiftの構成と特徴

- プロキシノード
  - クライアント要求を後段のストレージノードに振り分ける
- ストレージノード
  - アカウント情報の管理、オブジェクト情報の管理、オブジェクト実体の管理を行う



Swiftのオブジェクト配置方式

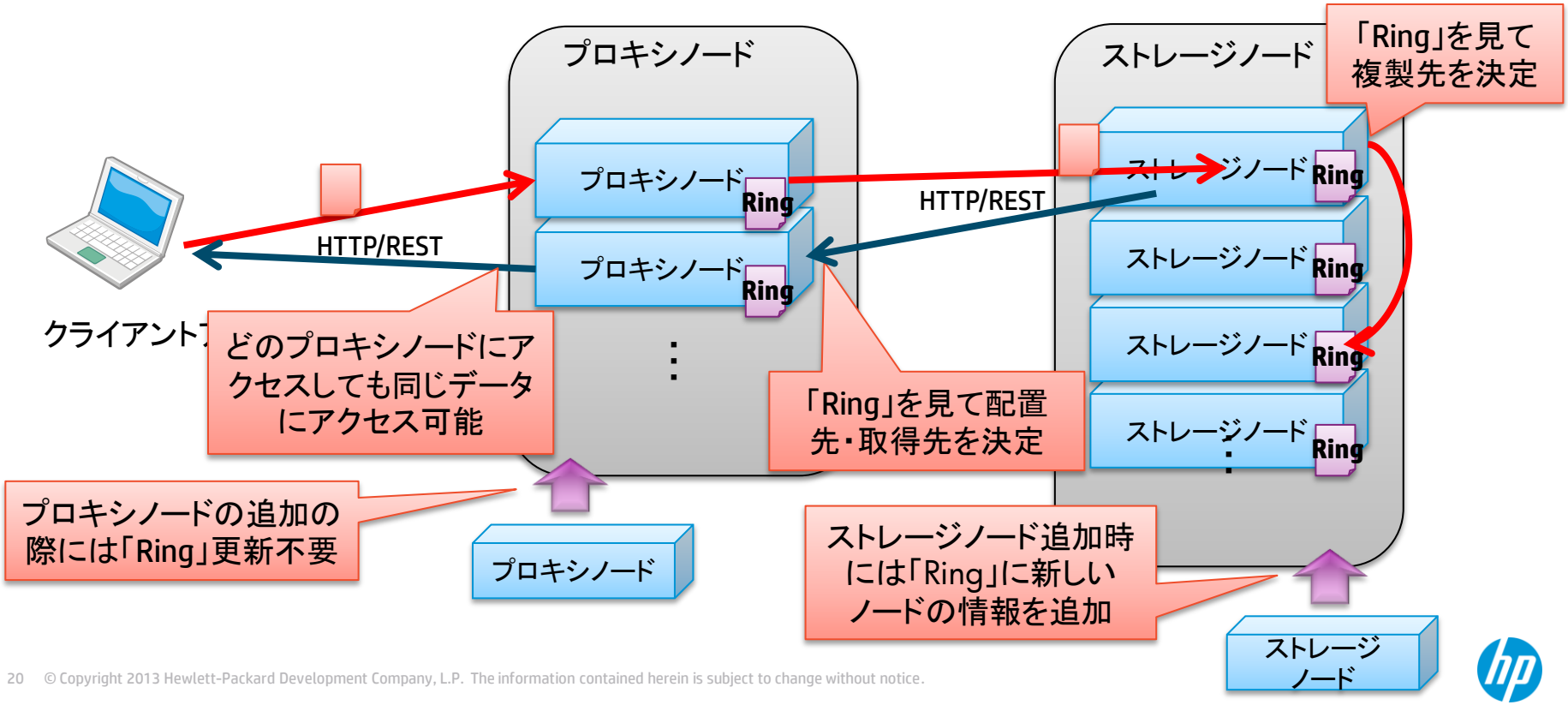
- オブジェクトのリソース名のハッシュ値から配置先のストレージノードのディスクを一意に決定
- 「Ring」と呼ばれるファイルが配置先の対応表の情報を保持



Swiftのデータ管理

SPOFをなくし、性能と容量のスケールアウトを実現

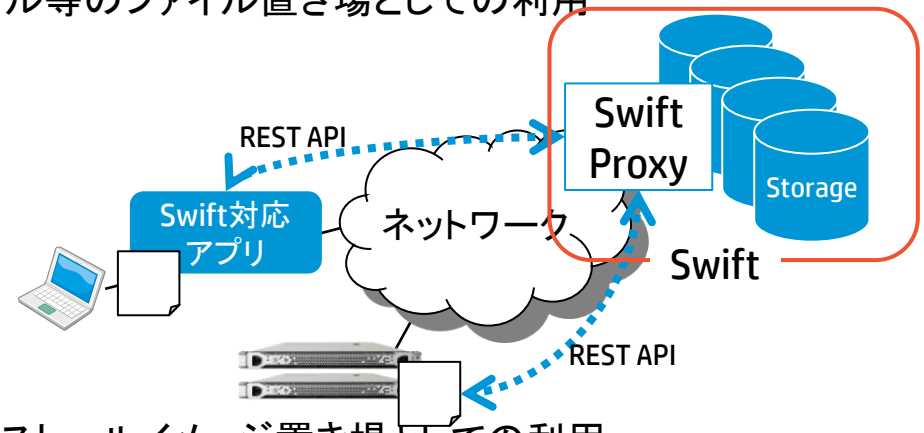
- 全てのプロキシノード、ストレージノードに同じ「Ring」データを配置
- プロキシノードの追加でスループットの向上が可能
- ストレージノードの追加でストレージ容量の向上が可能
  - ストレージノード追加時には全ノードが保持している「Ring」に新しいノードの情報を追加する必要あり。



Swiftのユースケース例

SWIFTのREST APIに対応したクライアントアプリケーションから利用する

- **プライベートDropbox**
  - 動画や写真など、個人用のファイル置き場としての利用
  - 社内の電子書類、ログファイル、イメージファイル等のファイル置き場としての利用
  - クライアント:
    - REST APIを利用したSwiftとの同期機能を持つクライアント
    - WebブラウザからSwiftと接続するWebアプリケーションにアクセス



- **OpenStackでのイメージファイル管理**
  - OpenStackにデプロイする仮想マシン用OSインストールイメージ置き場としての利用
  - 利用休止中の仮想マシンのディスクイメージ置き場としての利用

- **Swiftに向かない例**:
  - 頻繁に更新されるファイル(DBのデータファイルなど)の格納
  - NAS のようなNFSやCIFSを使ったアクセスが求められるシステム

OpenStack/Swiftデモ

デモ環境(ネットワーク構成)

Procurve2810-48G Gigabit Ethernet Switch

#01

Nova
Glance
Keystone
Cinder
Horizon

#02

Swift Proxy

#03

#04

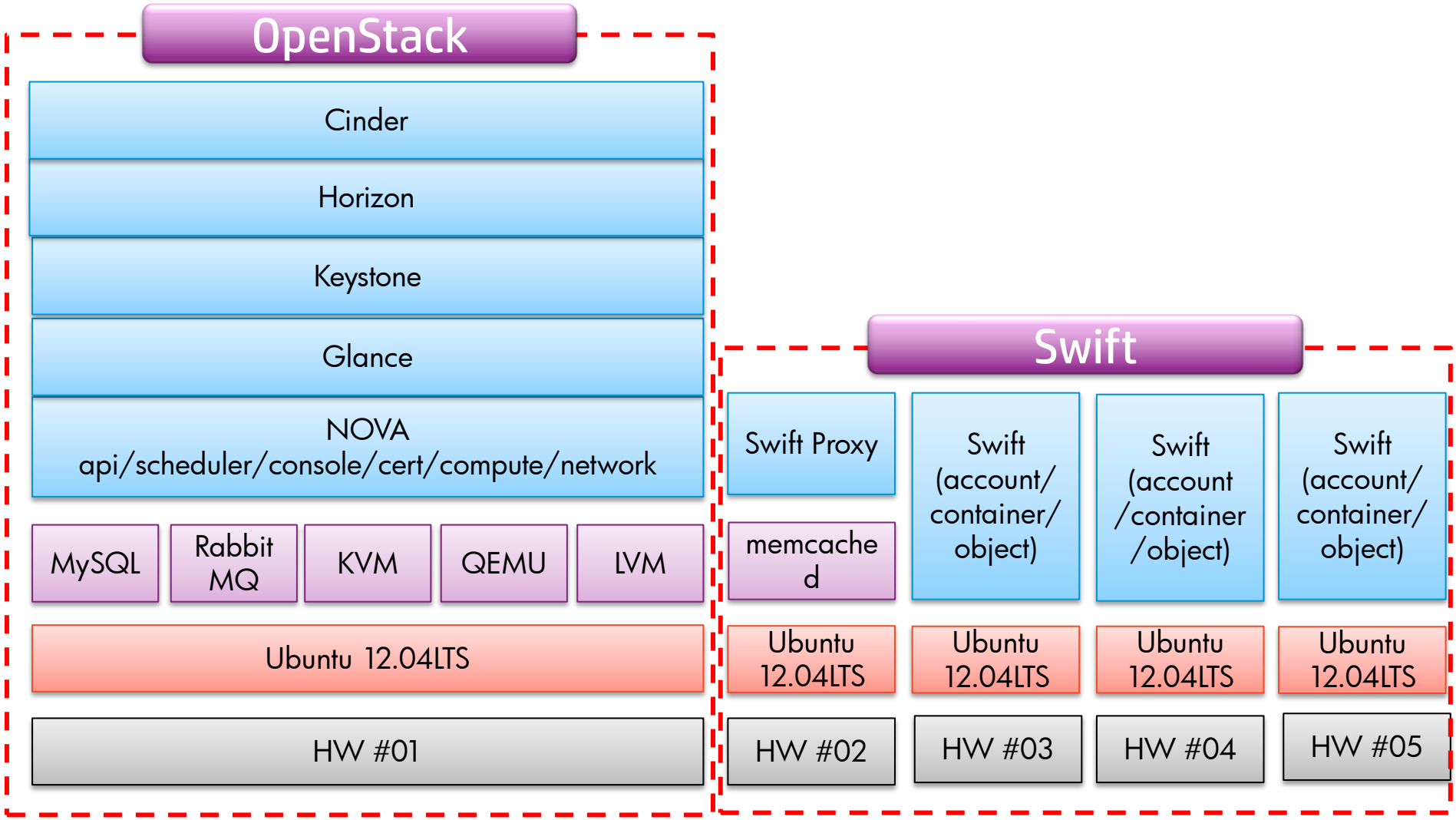
#05

Swift Object Server

■共通HW/OS情報
HW: ProLiant DL160G6
OS: Ubuntu12.04 LTS(x86_64)
CPU : Xeon 2.4GHz 2P12C
メモリ : 96GB
ディスク容量 : 2TB

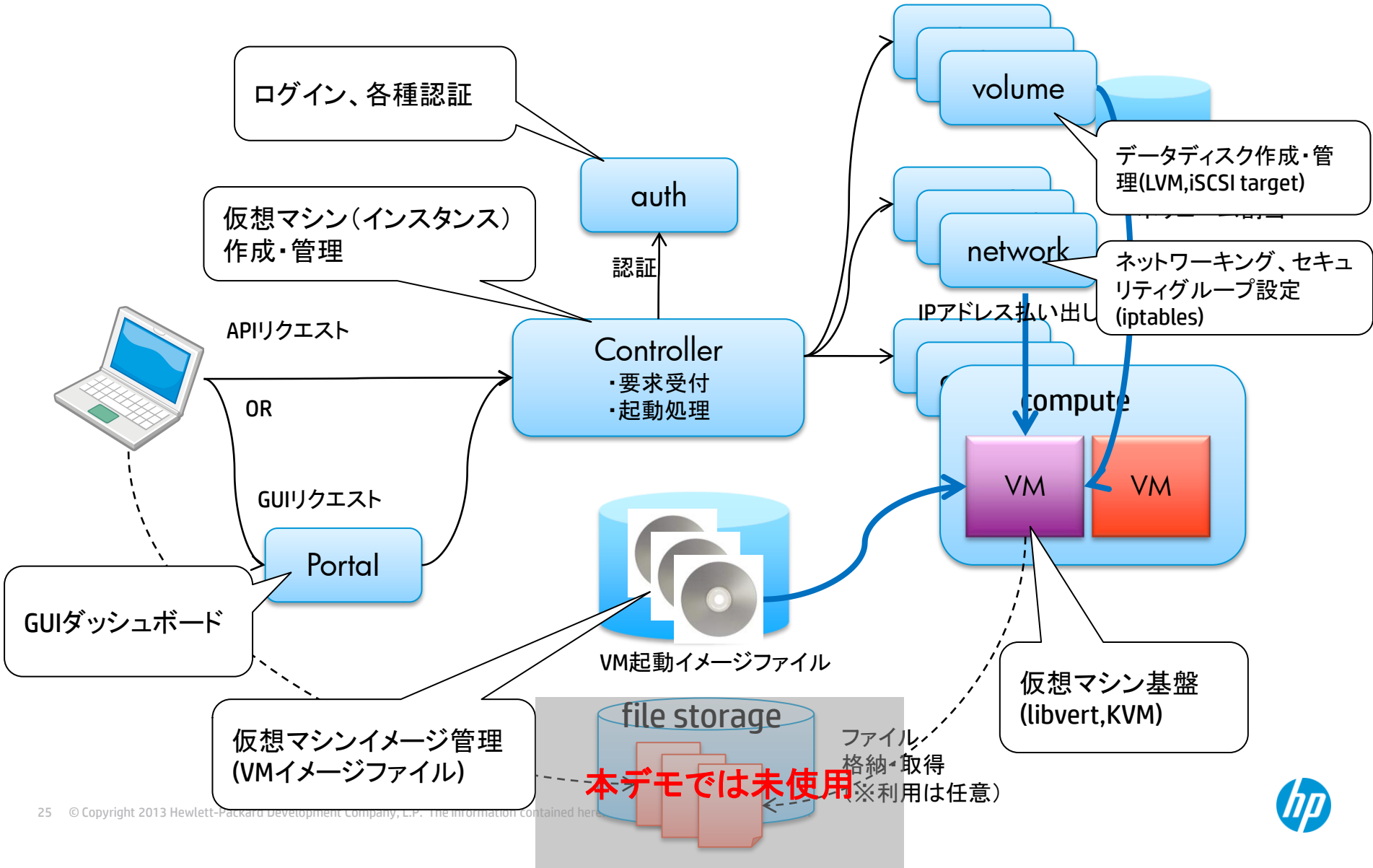
■OpenStackバージョン
Folsom(Canonical Folsomリポジトリ版)

デモ環境(コンポーネント構成)



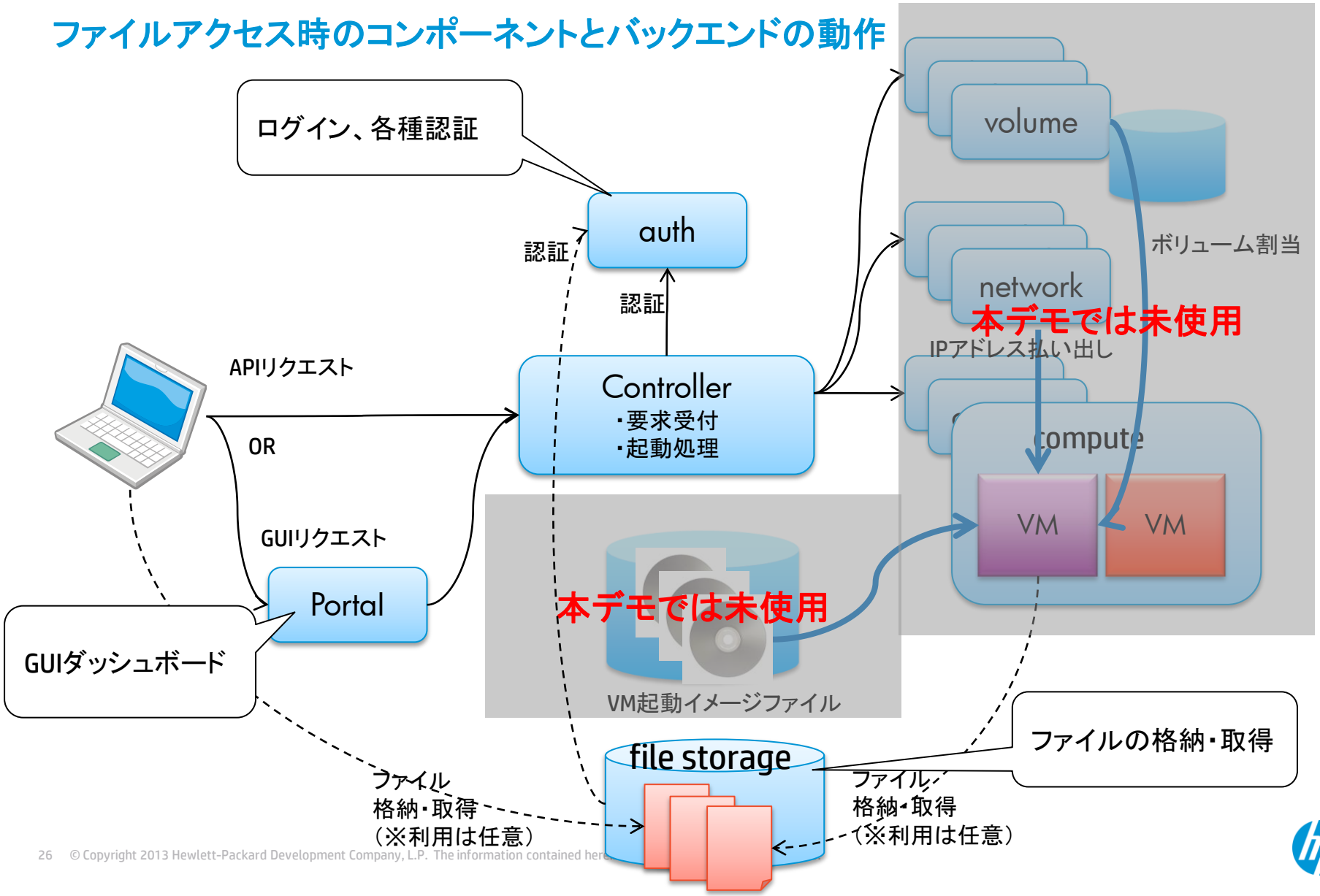
OpenStackデモ解説

仮想マシン作成時のコンポーネントとバックエンドの動作



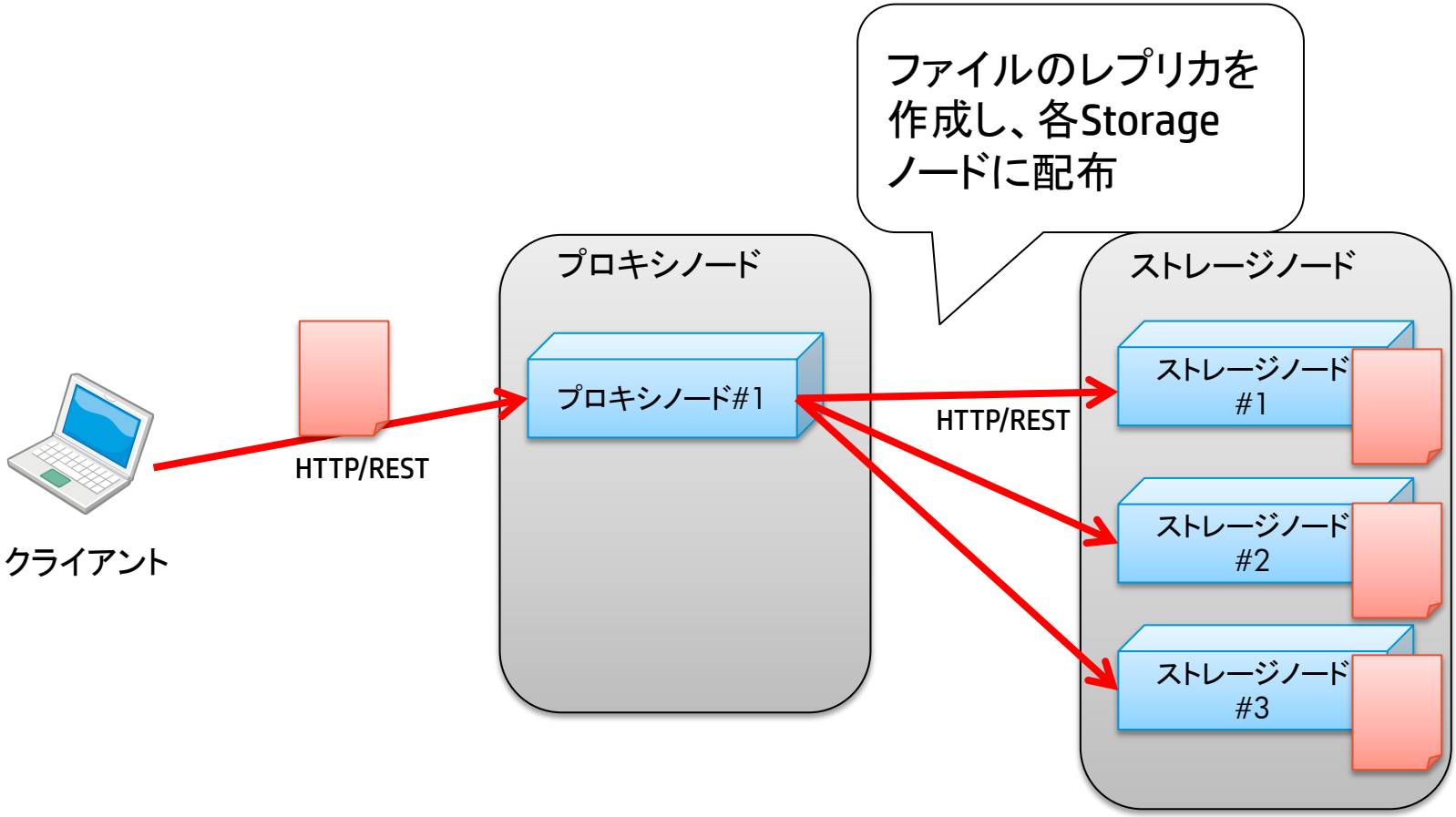
Swiftデモ解説

ファイルアクセス時のコンポーネントとバックエンドの動作



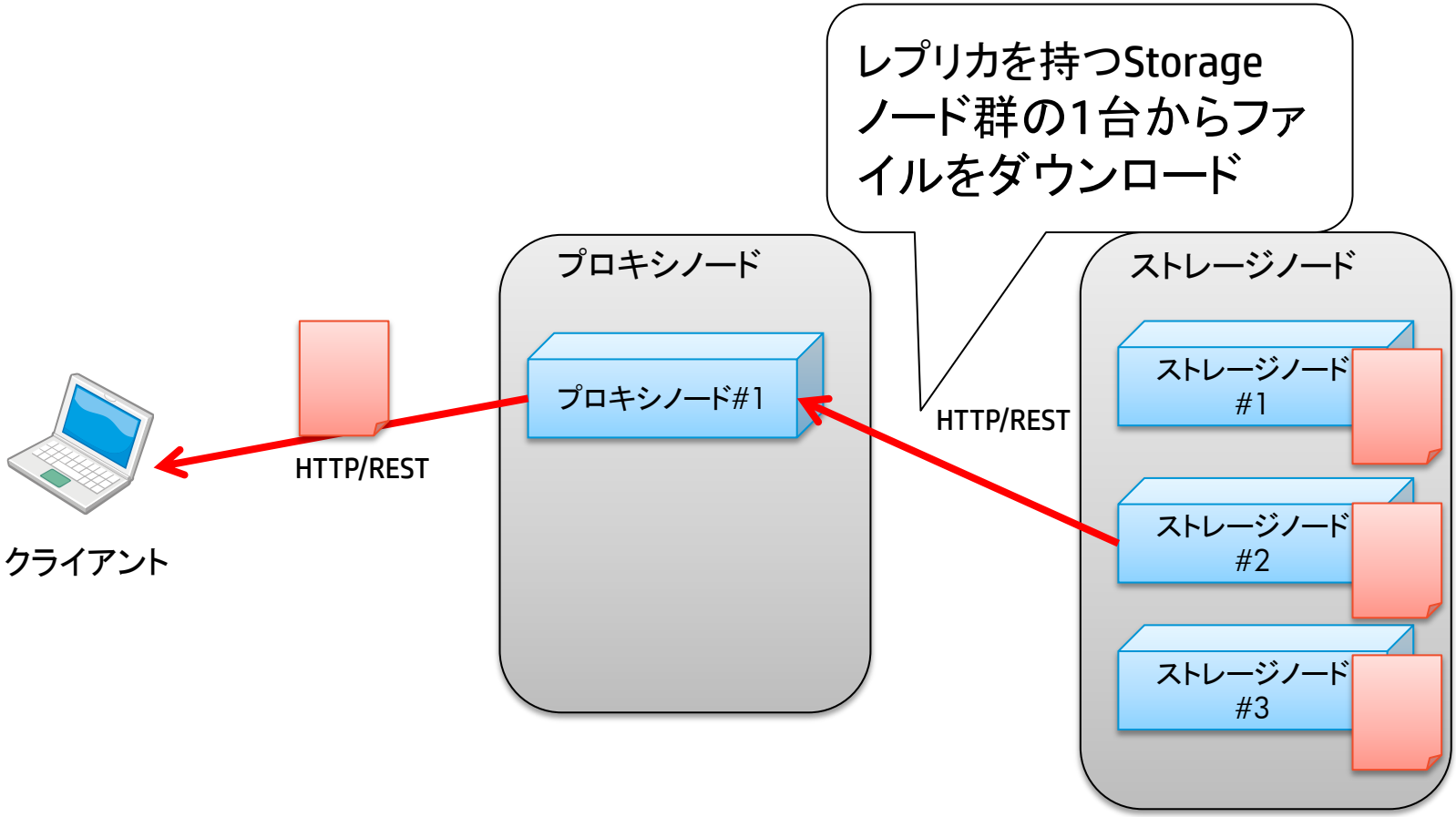
Swiftデモ解説

ファイルのアップロード時の動作



Swiftデモ解説

ファイルのダウンロード時の動作



Q & A

注：会場でいただいた質問への回答をセミナー資料に追記して掲載しています。

Q. Swiftにファイルを入れるときに、例えば、/usr/local/foo/bar.txtのようなフォルダを作って、そこにファイルを入れることはできるのか？

A. Swiftの「コンテナ」自体は1階層ですが、通常のフォルダ構造のような階層構造を作るために「疑似フォルダ」という機能があります。この機能を使った場合では「/usr/local/foo/bar.txt」という名前を持った「オブジェクト」を登録し、「オブジェクト」名の中の「/」をフォルダ区切りと見なして扱うようにすることで、フォルダ構造があるかのようなファイル操作を行うことができます。

Q. Swiftのファイル属性にはどのようなものがあるのか、バージョニング等の情報を入れることができるのか。

A. Swiftのファイル属性としては、アカウント情報、コンテナ情報、オブジェクト情報のほかACL、タイムスタンプ等があり、任意のものを追加可能です。最新のSwiftではオブジェクトのバージョニングも可能になっています。

Q. Swiftに接続できるようなクライアントにはどのようなものがあるのか。

A. 「Cloudberry」というクライアントがあり、これを使うとSwiftに接続しファイル転送をすることが出来ます。有償版もあるが、無償版もあります。（URL: http://www.cloudberrylab.com/free-openstack-storage-explorer.aspx）

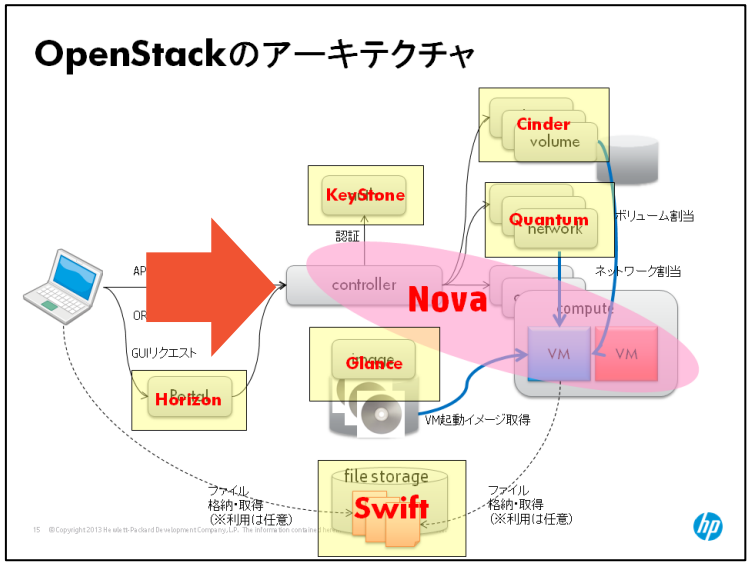
Q. Ringによってデータ配置が決まると言うことは、ノード追加後にRingの内容を更新して各ノードに配布した場合、それに基づいて元からあったファイルも移動するのか。

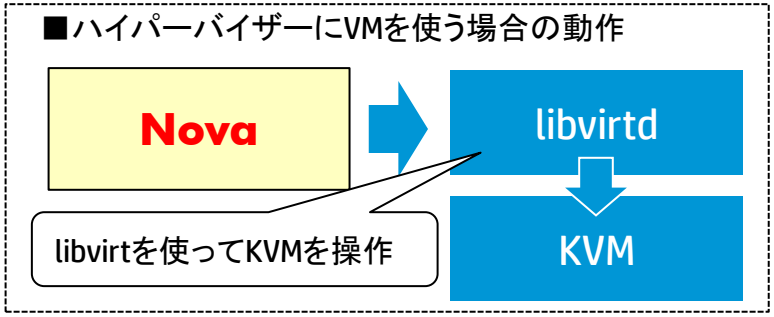
A. その通り。更新されたRingが配布されると、元からあったデータもそれに応じて再配置されます。ただし、徐々に配置変更を行い、性能面の影響が出ないようになっています。

補足資料

OpenStack 構成要素と特長

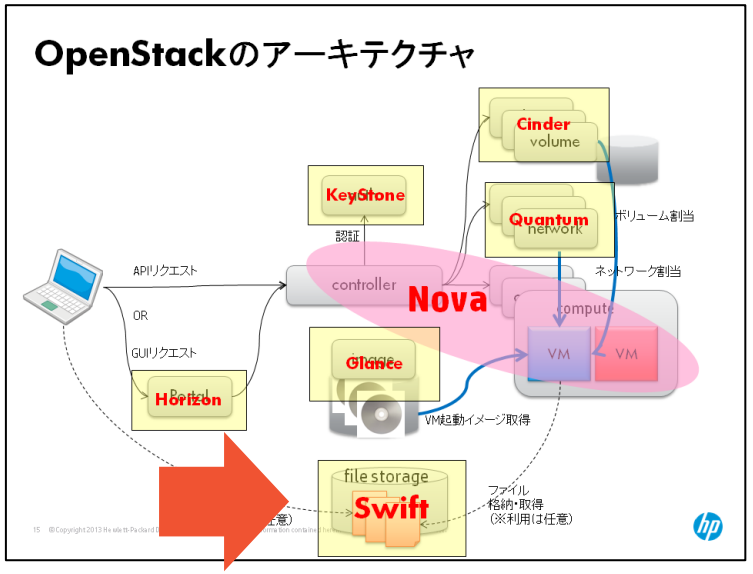
- Nova
  - 役割
    - 仮想マシンコントロール基盤
  - 動作概要
    - APIを通じて、OpenStackの各コンポーネントを連携させる。
    - KVM等のハイパーバイザーをコントロールし、仮想マシンを作成。
    - Glance、Cinder、Quantum等から取得したリソース（OSイメージ、ブロックデバイス、ネットワーク）を仮想マシンにアタッチ。





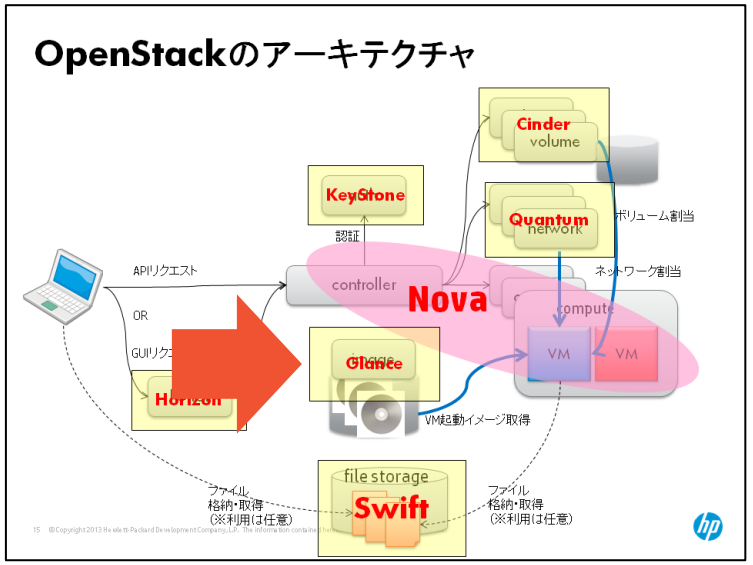
OpenStack 構成要素と特長

- Swift
  - 役割
    - 高可用性ファイルストア
  - 動作概要
    - HTTP経由でアクセスする分散ファイルストレージを提供。



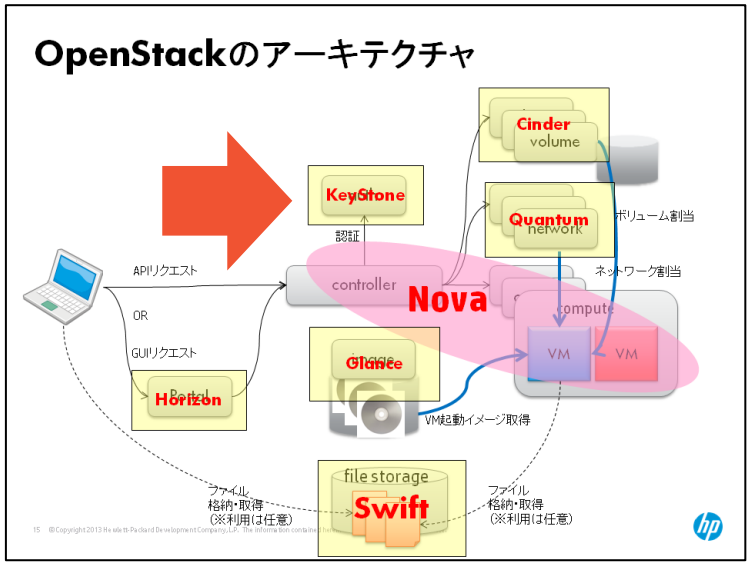
OpenStack 構成要素と特長

- Glance
  - 役割
    - 仮想マシンイメージ管理
  - 動作概要
    - Novaが作成した仮想マシンにブートイメージを提供する。
    - データ保存場所としてローカルファイルシステムに加え、Swiftを利用可能。



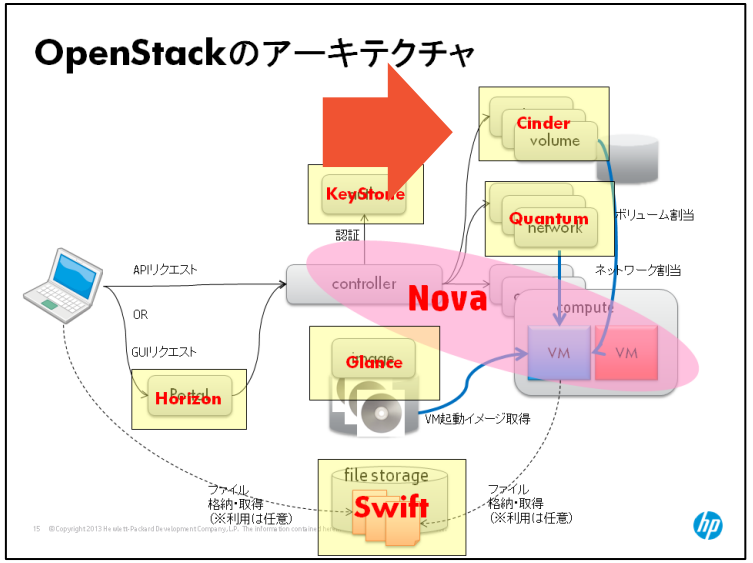
OpenStack 構成要素と特長

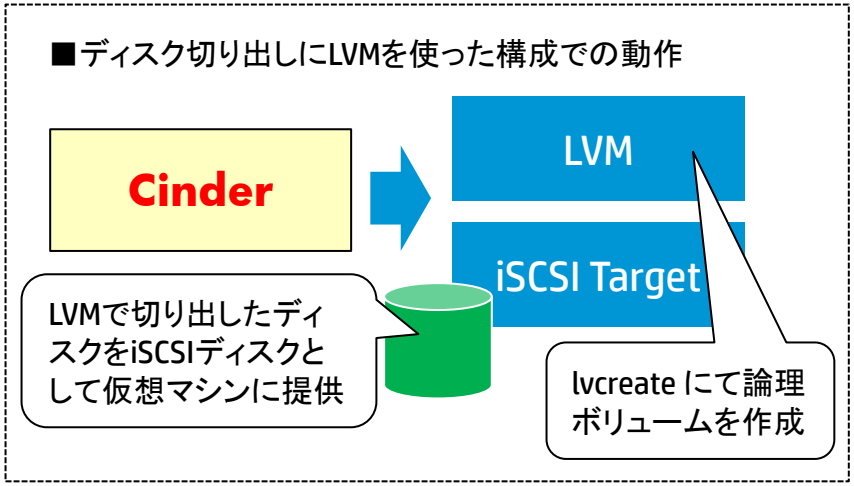
- Keystone
  - 役割
    - 統合認証基盤
  - 動作概要
    - OpenStackの各コンポーネントに共通の認証基盤を提供。
    - 各コンポーネントはKeystoneが発行したTokenを元にユーザ認証を行う。



OpenStack 構成要素と特長

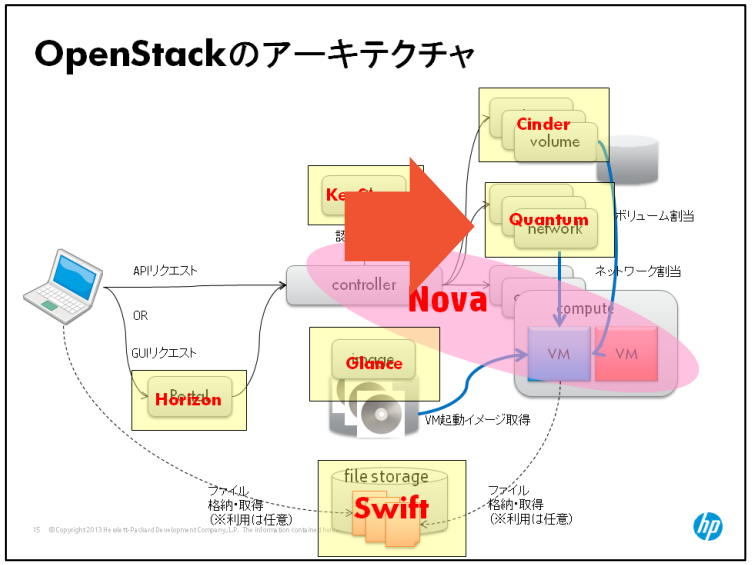
- Cinder
  - 役割
    - ブロックストレージ管理
  - 動作概要
    - iSCSIストレージにアクセスして、ボリュームを作成。
    - 作成したボリュームをNovaが作成した仮想マシンに提供する。





OpenStack 構成要素と特長

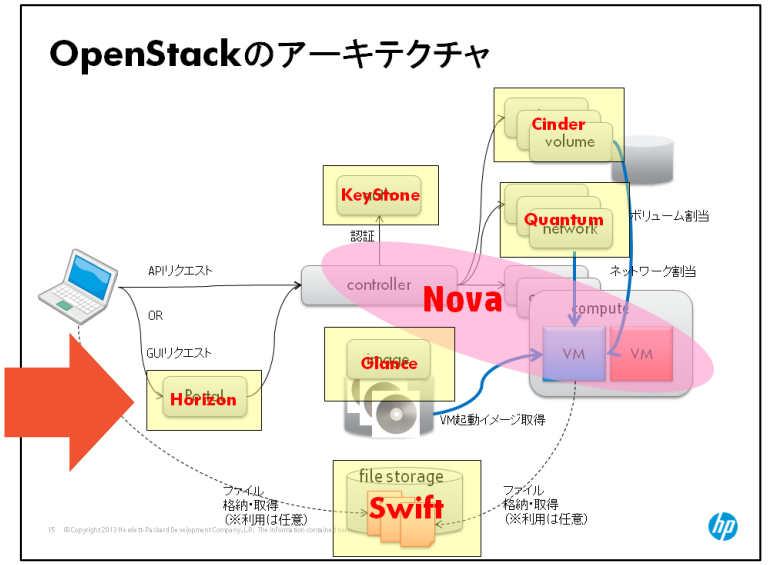
- Quantum
  - 役割
    - 仮想マシンネットワーク管理
  - 動作概要
    - iptables, Network Namespace 等を用いて仮想ネットワークを構築し、それを仮想マシンに割り当てる。



herein is subject to change without notice.

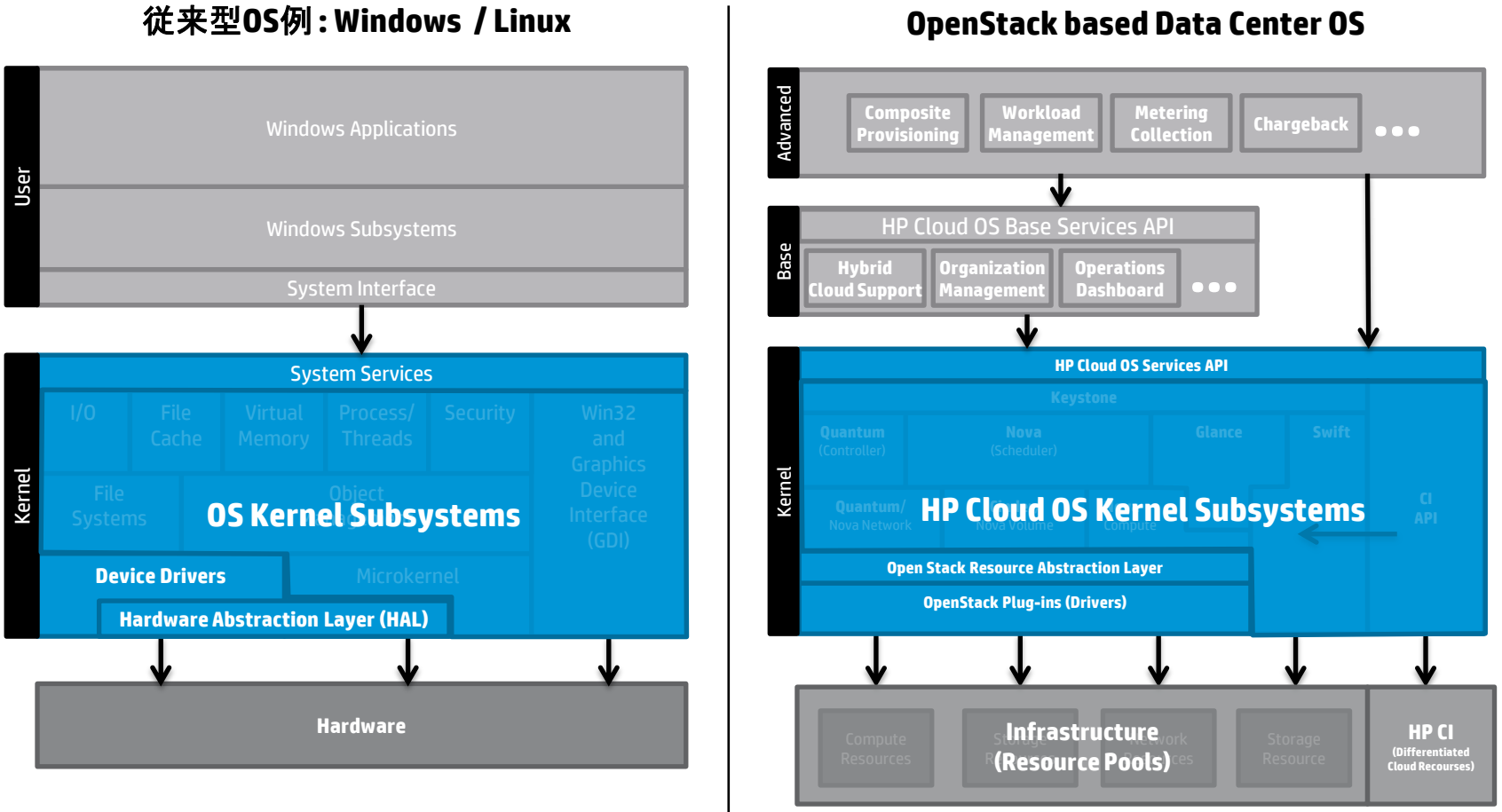
OpenStack 構成要素と特長

- Horizon
  - 役割
    - OpenStackの標準ダッシュボード
    - ブラウザからのOpenStack操作
  - 動作概要
    - 各コンポーネントの管理GUI提供
      - 仮想マシン
      - ブロックストレージ
      - オブジェクトストレージ
      - ネットワーク
      - ユーザ認証　など。
    - リソース使用状況の把握。



OpenStackを"OS"としたデータセンター

HPのCloud OS構想を例に



Thank you