**SoftLayerハンズオンセミナー**

WordPress編

**2016/2/222**

**日本アイ・ビー・エム株式会社**

**マーケティング＆コミュニケーションズ**

**改定履歴**

|  |  |  |  |
| --- | --- | --- | --- |
| 日時 | バージョン | コメント | 更新者 |
| 2015/7/14 | 初版 |  |  |
|  |  |  |  |
|  |  |  |  |

**目次**

[1. 事前準備 3](#_Toc425691117)

[1.1. 諸注意 3](#_Toc425691118)

[1.2. ハンズオンの概要 4](#_Toc425691119)

[1.3. 本ハンズオンで構築する環境 4](#_Toc425691120)

[1.4. 仮想インスタンスの準備 5](#_Toc425691121)

[2. SSH鍵の自動展開 6](#_Toc425691122)

[2.1. SSH鍵の作成 6](#_Toc425691123)

[2.2. SSH鍵のインスタンスへの登録方法 8](#_Toc425691124)

[3. Provisioning Script 9](#_Toc425691125)

[3.1. Provisioning Scriptの利用方法 9](#_Toc425691126)

[4. VLAN Spanninig 10](#_Toc425691127)

[5. スケーラブルWordPressシステムの展開 11](#_Toc425691128)

[5.1. Zabbixサーバのセットアップ 11](#_Toc425691129)

[5.2. バックエンドDBサーバ 14](#_Toc425691130)

[5.3. WordPressサーバ 16](#_Toc425691131)

[5.3.1. 一台目のセットアップ 16](#_Toc425691132)

[5.3.2. WordPressのセットアップ 18](#_Toc425691134)

[5.3.3. 二台目のセットアップ 20](#_Toc425691135)

[5.3.4. SoftLayer Object　Storageの設定 21](#_Toc425691136)

[5.4. Load Balancer 25](#_Toc425691137)

[5.4.1. Load Balancerの導入と設定 25](#_Toc425691138)

[5.5. Zabbixサーバへの登録 35](#_Toc425691139)

[6. 最後に 38](#_Toc425691140)

[6.1. 仮想インスタンスの削除 38](#_Toc425691141)

[6.2. Object Storage内のファイルの削除 38](#_Toc425691142)

[7. Appendix 39](#_Toc425691143)

[7.1. サポートチャットとチケット 39](#_Toc425691144)

[7.2. SolftLayerのネットワーク構成例 41](#_Toc425691145)

[7.3. Apacheを利用したWordPressサーバの構築 42](#_Toc425691146)

[7.4. Local Load Balancer 43](#_Toc425691147)

# 事前準備

## 諸注意

* SoftLayerのアカウントは、事前取得していることを前提にしています。次の「SoftLayer 無料トライアルのご案内」を参照してください。

|  |
| --- |
| 【SoftLayer無料トライアルのご案内】  <http://www.ibm.com/cloud-computing/jp/ja/softlayer_flow.html> |

* 本資料はSoftLayerハンズオン初級編を既に受講していることを前提にしています。

|  |
| --- |
| 【SoftLayerハンズオン初級編ガイド】  <http://www-06.ibm.com/ibm/jp/cloud/pdf/softlayer_handson_server_guide.pdf> |

* 当日は取得したアカウントを使用してハンズオンを行います。数十円程度の課金が発生します。
* SoftLayerのアカウントの解約について

全てのリソースがアカウントから削除されますと、お客様のアカウント環境はクローズされ、お手持ちのIDとパスワードでのログインが出来なくなり、事実上ご解約の扱いとなります。ただし、もし同じアカウントIDで再度アカウントを開通されたい場合にはSoftLayerの公式WEBサイト等のチャットにてお申し出頂く事で再度開通させて頂く事も可能です。

* IBMから貸与しているアカウントを利用している場合は、本資料に記載している以外のサービスを注文しないように注意してください。
* 講師の指定がない限り、本ハンズオンで作成するサーバーは、時間課金サーバー (**Virtual Server (public node) - Hourly**)を選択してください。

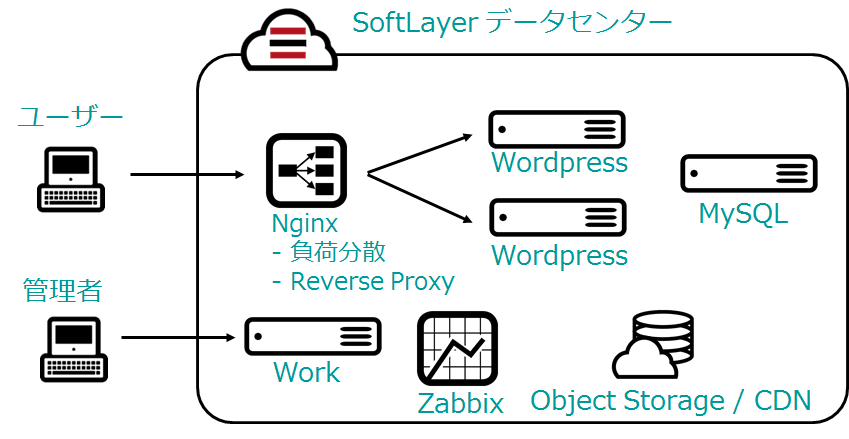
## ハンズオンの概要

本ハンズオンでは、よりSoftLayerを深くご理解いただくことを目的に、様々な機能や要素を用いてスケーラブルな環境を構築します。SoftLayerの提供する機能は、管理者の手間を省き迅速にサーバをデプロイする為に有用な機能や標準のセキュリティを向上させる機能など多岐にわたって存在しています。本ハンズオンを通じて、それらの機能の柔軟さと強力さに触れてみましょう。

## 本ハンズオンで構築する環境

本ハンズオンでは、下図に示すようなスケーラブルで高性能なWordPress環境をSoftLayer上に構築します。複数のWordPressサーバを自動で構築し、リバースプロキシサーバでコンテンツをキャッシュして通信負荷を低減します。また、ロードバランサーを導入して、外部からの要求を効率よく分散することで、大量のアクセスにも強いシステムとなります。

|  |  |
| --- | --- |
| **注意** | 本ハンズオンでは複数の仮想インスタンスを注文し利用します。既にSoftLayerをご利用で複数VLANセグメントをお持ちの場合は、注文の際に所属するVLANを明示的に指定して全てのVMが同じVLANに所属していることを確認して下さい。 |



## 仮想インスタンスの準備

SoftLayer上で下記スペックの時間課金仮想インスタンス（**Virtual Server (public node) - Hourly**）を構築し、SSHでの接続と操作が可能であることを確認してください。

|  |  |
| --- | --- |
| Data Center | San Jose 1 |
| Operating System | CentOS 6.x - Minimal Install (64 bit) |
| UPLINK PORT SPEEDS | 100 Mbps Public & Private Network Uplinks |
| Advanced Monitoring | None |
| Monitoring | Host Ping |
| Response | Automated Notification |
| Provisioning Scripts  Existing Script:  URL 1: | ブラウザで下記のURLを参照してスクリプトを確認してから、コピペで入力  (new URL)  https://raw.githubusercontent.com/kkitase/handson/master/tools/centos-base.sh |
| Host Name | ご自身のアカウントを利用している方: 任意  **IBMから貸与したアカウントを利用している方: アカウント名-work** |
| Domain Name | ご自身のアカウントを利用している方: 任意  **IBMから貸与したアカウントを利用している方: handson.jp** |

※他はデフォルト設定、講師から何か指定がある場合はそちらに従ってください。

仮想インスタンスの構築方法がわからない場合は、「SoftLayerハンズオン初級編ガイド」を参考にしてください。本ハンズオンではこの仮想インスタンスを利用しますので、必ず構築ください。以降、この仮想インスタンスのことを作業マシンと呼びます。本ハンズオンでは、特に指定がない限りは全て上記スペックの仮想インスタンスを用いるものとします。

Provisioning scriptが正しく実行されれば、ログイン時に下記のメッセージが出力されています。

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Provisioning was successful!!!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# SSH鍵の自動展開

SoftLayerでは、管理ポータルにSSHの公開鍵を事前に登録しておき、サーバのデプロイ時に指定した公開鍵を自動で組み込む機能が存在しています。後述のProvisioning Scriptと組み合わせることで、デプロイ時点でSSHのパスワード認証を禁止して鍵交換方式に限定することができるため、セキュリティが向上します。

## SSH鍵の作成

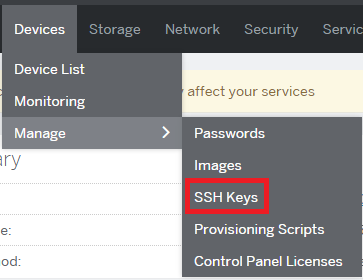
|  |
| --- |
| # ssh-keygen  Generating public/private rsa key pair.  Enter file in which to save the key (/home/shiroma/.ssh/id\_rsa):  Enter passphrase (empty for no passphrase):　[パスワードを入力/表示されません]  Enter same passphrase again: 　[同じパスワードを入力/表示されません]  Your identification has been saved in /root/.ssh/id\_rsa.  Your public key has been saved in /root /.ssh/id\_rsa.pub.  The key fingerprint is:  b5:40:b3:2e:de:95:df:7a:45:3c:8d:2f:d0:84:c6:31 root@workingvm  The key's randomart image is:  +--[ RSA 2048]----+  | o .Eo |  | . o +.. |  | o .. o o.|  | . o o. o.+|  | . S + . o.|  | . o . . .. o|  | . . . .o |  | .. |  | .. |  +-----------------+ |

まずは作業マシン上でSSHの鍵ペアを作成します。

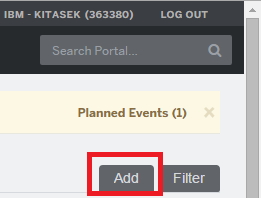
同じパスワードを適切に二回入力すると、秘密鍵(id\_rsa)と公開鍵(id\_rsa.pub)のペアが作成されます。それぞれの鍵ファイルは名前の通り、公開鍵は他人に公開する為の鍵、秘密鍵は厳重に管理し認証に用いる鍵となります。SoftLayerのポータルには公開鍵を登録します。作成した公開鍵を確認してみましょう。

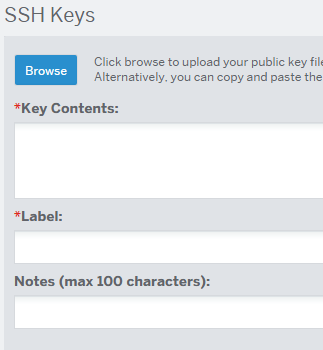
|  |
| --- |
| $ cat .ssh/id\_rsa.pub  ssh-rsa BBBB3NzaC1yc2EAAAABIwAAAQEAzdAGY+pYT5/74czK+Expt3I1xW2np+UMyM0baTPfTjSoYT  pus2o44Y8zR881bbbbbb9DmMF3K02r4qB0pdz4iioqCoYupbbbbRpBy/I7hIc9asnt8BWwTRjk1  pauIFQ882KFY3+b4t1UlgmBfGmV6QEHaA4j1JEbVW6M2Hv1JspH9f/H2L44bbbbbbbbbbbbbCb  THQgCKpIH4aYb34DHfH4T8bbbzQfef8X+iwvBtYz/JNN119Q9hbbbbVuIxvjHif9xPUsRlurLPJs45b  zT1xY68A1G33eBO9NyJKkRMXAW4WEO1a1Z+vDcTjQ== root@student1031-work.handson.jp |

公開鍵を確認したら、SoftLayerのポータルに公開鍵を登録します。ポータルのトップからDevice -> SSH Keysの順にクリックします。



SSH鍵の管理メニューが表示されたら、右上のAddをクリックします。





鍵の登録メニューが表示されたら、Key Contentsに公開鍵(id\_rsa.pub)の内容をコピー&ペーストし、Labelに識別用の名前（任意ですが、本ハンズオンではアカウント名）を入力します。入力が完了したら、右下のAddをクリックし、登録を完了します。

これでSSH公開鍵の登録は完了です。次の注文時から、マシンの注文時に指定することで登録した公開鍵を含んだ状態でデプロイすることが可能です。

## SSH鍵のインスタンスへの登録方法

物理、仮想問わず、SSH鍵のインスタンスへの登録は、注文時にしか行えません。上で登録した鍵は、インスタンス注文の最終段階においてSSH Keysという項目で選択可能です。



# Provisioning Script

SoftLayerには、Provisioning Scriptと呼ばれるデプロイ自動化の仕組みが存在しています。この仕組は、インスタンスの注文時のデプロイ最終段階において、ユーザーが指定したスクリプトを自動で実行するものです。

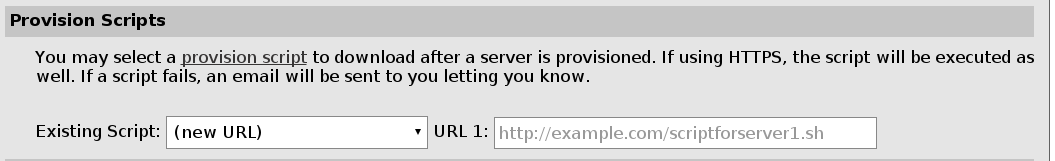
|  |  |
| --- | --- |
| **注意** | 注文したインスタンスでProvisioning Scriptを自動実行するためには、HTTPSでスクリプトを取得する必要があります。Provisioning ScriptにHTTPのURLを指定した場合、自動実行は行われずスクリプトのダウンロードに留まります。 |

Provisioning Scriptは基本的にはシェルスクリプトなので、手元の環境で動作を確認の上でProvisioning Scriptとして登録をすることができます。Provisioning Scriptはデプロイ中に自動で実行される都合上、利用者の入力を必要とする対話的なコマンドを用いることはできませんので、yumをyum -yに置き換えるなどの工夫が必要です。発展的な利用方法として、Provisioning Scriptの最初にRubyやPHP、Pythonといった他のスクリプト実行環境を導入し、wget等で外部からソースコードをダウンロードして実行することも可能です。

　本ハンズオンでは、4ページの図におけるReverse Proxy、WordPress Server、MySQL Server、そしてZabbix Monitoringの4つの役割に対して、それぞれ必要なソフトウェアの導入と設定の大部分を行うよう作成されたProvisioning Scriptを用いることでProvisioning Scriptの力強さを体験します。

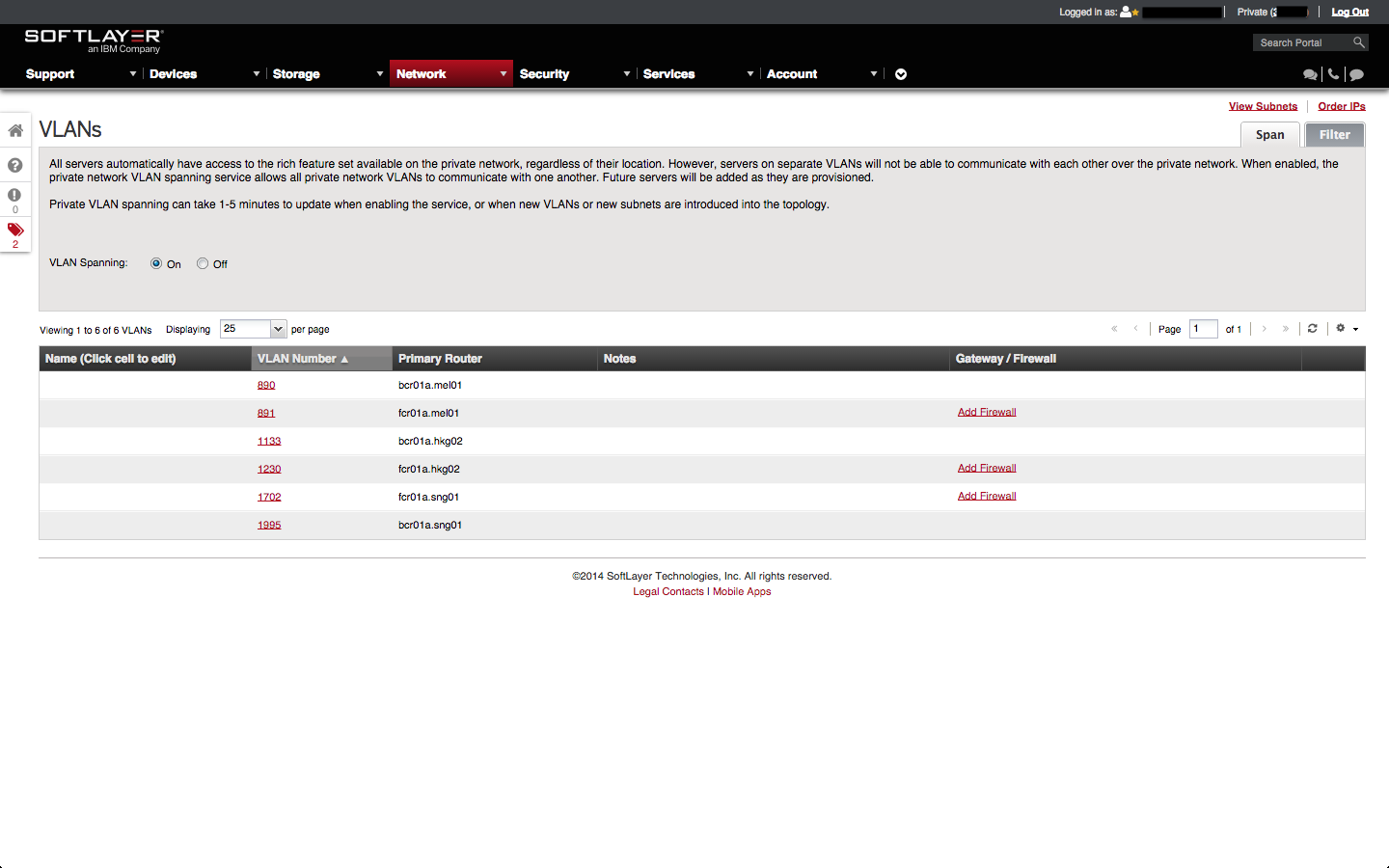
## Provisioning Scriptの利用方法

Provisioning Scriptは注文確定画面最終段階で設定可能です。URL1に自動で実行させたいProvisioning Scriptを指定することで、インストール時にそれらのスクリプトを自動で実行させることができます。



# VLAN Spanninig

複数のPrivate VLANとの通信を有効にする為に、VLAN Spanningの設定を行います。管理ポータルから、Network – IP Management – VLANsで、Spanタブをクリックし、VLAN Spanning: Onをチェックします。



# スケーラブルWordPressシステムの展開

本章では、実際にProvisioning Scriptを活用しながらスケーラブルなWordPressシステムを構築します。本ハンズオンで構成するスケーラブルWordPress環境は、以下の通り複数の段階を踏んで負荷分散を行っています。

1. 上位で負荷分散を行うロードバランサー
2. 途中でコンテンツをキャッシュするReverse Proxy
3. 複数の末端WordPressサーバ

これらの環境を構築するために、本ハンズオンでは以下の順番でマシンを注文し、設定します。

1. マシン群を監視するZabbixサーバ
2. バックエンドに設置されるDBサーバ
3. WordPressサーバ
4. Reverse Proxyサーバ
5. ロードバランサ

これから仮想サーバーを5台作成しますが、下記の要領でまとめてオーダーしたほうが便利です。

|  |  |
| --- | --- |
| ../../../../../Documents/Screenshot/%202016-02-23%2018.30.33.p | ../../../../../Documents/Screenshot/%202016-02-23%2018.31.23.p |

## Zabbixサーバのセットアップ

管理ポータルより、下記スペックの時間課金仮想インスタンス（**Virtual Server (public node) - Hourly**）を作成してください。

|  |  |
| --- | --- |
| Data Center | San Jose 1 |
| Operating System | CentOS 6.x - Minimal Install (64 bit) |
| UPLINK PORT SPEEDS | 100 Mbps Public & Private Network Uplinks |
| Advanced Monitoring | None |
| Monitoring | Host Ping |
| Provisioning Scripts  Existing Script:  URL 1: | (new URL)  https://raw.githubusercontent.com/kkitase/handson/master/tools/zabbix.sh |
| SSH Key | 登録したSSH keyのラベルを選択 例: student1031 |
| Host Name | ご自身のアカウントを利用している方: 任意  **IBMから貸与したアカウントを利用している方: アカウント名-zabbix** |
| Domain Name | ご自身のアカウントを利用している方: 任意  **IBMから貸与したアカウントを利用している方: handson.jp** |

※他はデフォルト設定、講師から何か指定がある場合はそちらに従ってください。

Provisioning scriptが正しく実行されれば、ログイン時に下記のメッセージが出力されています。

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Provisioning was successful!!!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

上記のProvisioning Scriptは、以下の内容を自動化します。

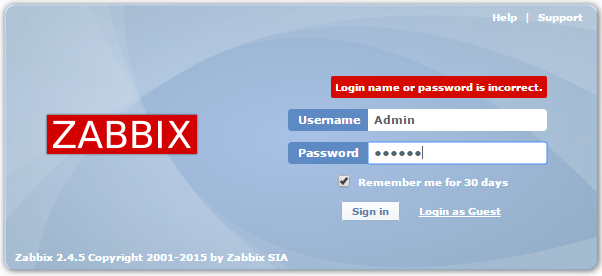
1. WebサーバとMySQLサーバのインストール、セットアップ
2. Zabbixサーバ、クライアントのインストール
3. Zabbixサーバのセットアップ

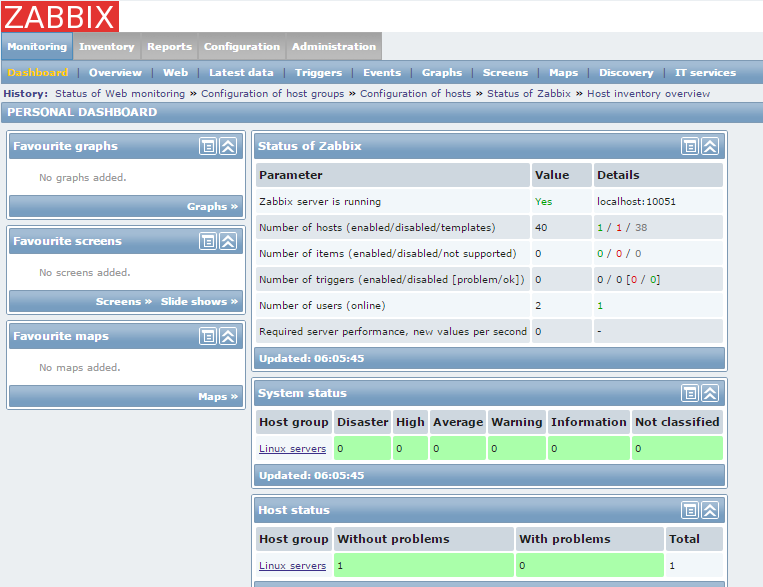
Provisioning Scriptでセットアップを行った場合は、すぐにZabbixを利用できる状態になっています。ブラウザからZabbixのセットアップを行って利用を開始しましょう。セットアップは以下のURLから開始できます。

|  |
| --- |
| <http://Zabbix>サーバのIPアドレス/zabbix/ |

インストールはウィザード形式になっています。「3: Configure DB connection」では、Database Name,User,Passwordの全てにzabbixと入力してください。それ以外は全て右下のNextをクリックし、ウィザードを進めてください。

インストールウィザードが終了したら、ログイン画面が表示されます。デフォルトのパスワードはAdmin/zabbixです。早速ログインしてみましょう。





Zabbixのセットアップは以上で完了です。引き続き他のノードをセットアップし、最後にそれらのノードをZabbixを追加し、自動で監視を行います。

## バックエンドDBサーバ

次はバックエンドのDBサーバを構築します。本来DBサーバは多くのトランザクションへの対応が予想されるため、仮想インスタンスではなくベアメタルインスタンスを利用する事が多いですが、本ハンズオンでは仮想インスタンスで代用します。ベアメタルインスタンスでも同様にSSH鍵の自動登録やProvisioning Scriptを活用することができます。

管理ポータルより、下記スペックの時間課金仮想インスタンス（**Virtual Server (public node) - Hourly**）を作成してください。

|  |  |
| --- | --- |
| Data Center | San Jose 1 |
| Operating System | CentOS 6.x - Minimal Install (64 bit) |
| UPLINK PORT SPEEDS | 100 Mbps Public & Private Network Uplinks |
| Advanced Monitoring | None |
| Monitoring | Host Ping |
| Response | Automated Notification |
| Provisioning Scripts  Existing Script:  URL 1: | ブラウザで下記のURLを参照してスクリプトを確認してから、コピペで入力  (new URL)  https://raw.githubusercontent.com/kkitase/handson/master/tools/db.sh |
| SSH Key | 登録したSSH keyのラベルを選択 例: student1031 |
| Host Name | ご自身のアカウントを利用している方: 任意  **IBMから貸与したアカウントを利用している方: アカウント名-db** |
| Domain Name | ご自身のアカウントを利用している方: 任意  **IBMから貸与したアカウントを利用している方: handson.jp** |

※他はデフォルト設定、講師から何か指定がある場合はそちらに従ってください。

※Provisioning scriptが正しく実行されれば、ログイン時に下記のメッセージが出力されています。

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Provisioning was successful!!!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Provisioning Scriptでは主に以下の3点を自動で行っています。

1. MySQLサーバの自動インストールと設定
2. Zabbixエージェントの自動インストール
3. WordPress用DBユーザーの自動追加

ログインしたら、以上の内容がセットアップされていることを確認しましょう。

|  |
| --- |
| [root@db ~]# yum list installed | grep -i -e mysql -e zabbix  mysql.x86\_64 5.1.73-5.el6\_6 @updates  mysql-libs.x86\_64 5.1.73-5.el6\_6 @updates  mysql-server.x86\_64 5.1.73-5.el6\_6 @updates  perl-DBD-MySQL.x86\_64 4.013-3.el6 @base  zabbix.x86\_64 2.4.5-1.el6 @zabbix  zabbix-agent.x86\_64 2.4.5-1.el6 @zabbix  zabbix-get.x86\_64 2.4.5-1.el6 @zabbix  zabbix-release.noarch 2.4-1.el6 installed  [root@db ~]# **mysql -u root**  Enter password: [Enter]  Welcome to the MySQL monitor. Commands end with ; or \g.  (中略)  mysql>**select Host, User, Password from mysql.user;**  +-----------------+----------------+-------------------------------------------+  | Host | User | Password |  +-----------------+----------------+-------------------------------------------+  (中略)  | db-bare.ibm.com | | |  | % | WordPress\_user | \*5098F389DDBE540AB35FE3EA85BF9CC4597208F6 |  +-----------------+----------------+-------------------------------------------+ |

## WordPressサーバ

次はWordPressサーバのデプロイです。一般にWordPressのインストールには非常に複雑な手順を要しますが、Provisioning Scriptによりこれらの多くは自動で行われます。まずは一台目をインストールしましょう。

### 一台目のセットアップ

管理ポータルより、下記スペックの時間課金仮想インスタンス（**Virtual Server (public node) - Hourly**）を作成してください。

|  |  |
| --- | --- |
| Data Center | San Jose 1 |
| Operating System | CentOS 6.x - Minimal Install (64 bit) |
| UPLINK PORT SPEEDS | 100 Mbps Public & Private Network Uplinks |
| Advanced Monitoring | None |
| Monitoring | Host Ping |
| Response | Automated Notification |
| Provisioning Scripts  Existing Script:  URL 1: | ブラウザで下記のURLを参照してスクリプトを確認してから、コピペで入力  (new URL)  https://raw.githubusercontent.com/kkitase/handson/master/tools/backweb\_nginx.sh |
| SSH Key | 登録したSSH keyのラベルを選択 例: student1031 |
| Host Name | ご自身のアカウントを利用している方: 任意  **IBMから貸与したアカウントを利用している方: アカウント名-wordpress1** |
| Domain Name | ご自身のアカウントを利用している方: 任意  **IBMから貸与したアカウントを利用している方: handson.jp** |

※他はデフォルト設定、講師から何か指定がある場合はそちらに従ってください。

※Provisioning scriptが正しく実行されれば、ログイン時に下記のメッセージが出力されています。

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Provisioning was successful!!!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Provisioning Scriptでは以下の設定を自動で行います。

1. Zabbixエージェントの自動インストール
2. WordPressの自動インストールとNginxの自動設定
3. WordPress用データベースの情報設定

WordPressサーバのProvisioning Scriptでは、WordPressを構成するために必要なほぼすべての設定が自動化されていますが、DBサーバの設定だけは自分で変更する必要があります。5.2で注文したバックエンドDBサーバのPrivate IP Addressを確認し、WordPressの設定ファイルに変更を加えましょう。

|  |
| --- |
| [root@backweb ~]# vi /etc/wordpress/wp-config.php  /\*\* MySQL hostname \*/  define('DB\_HOST', 'DBサーバのPrivate IP address'); |

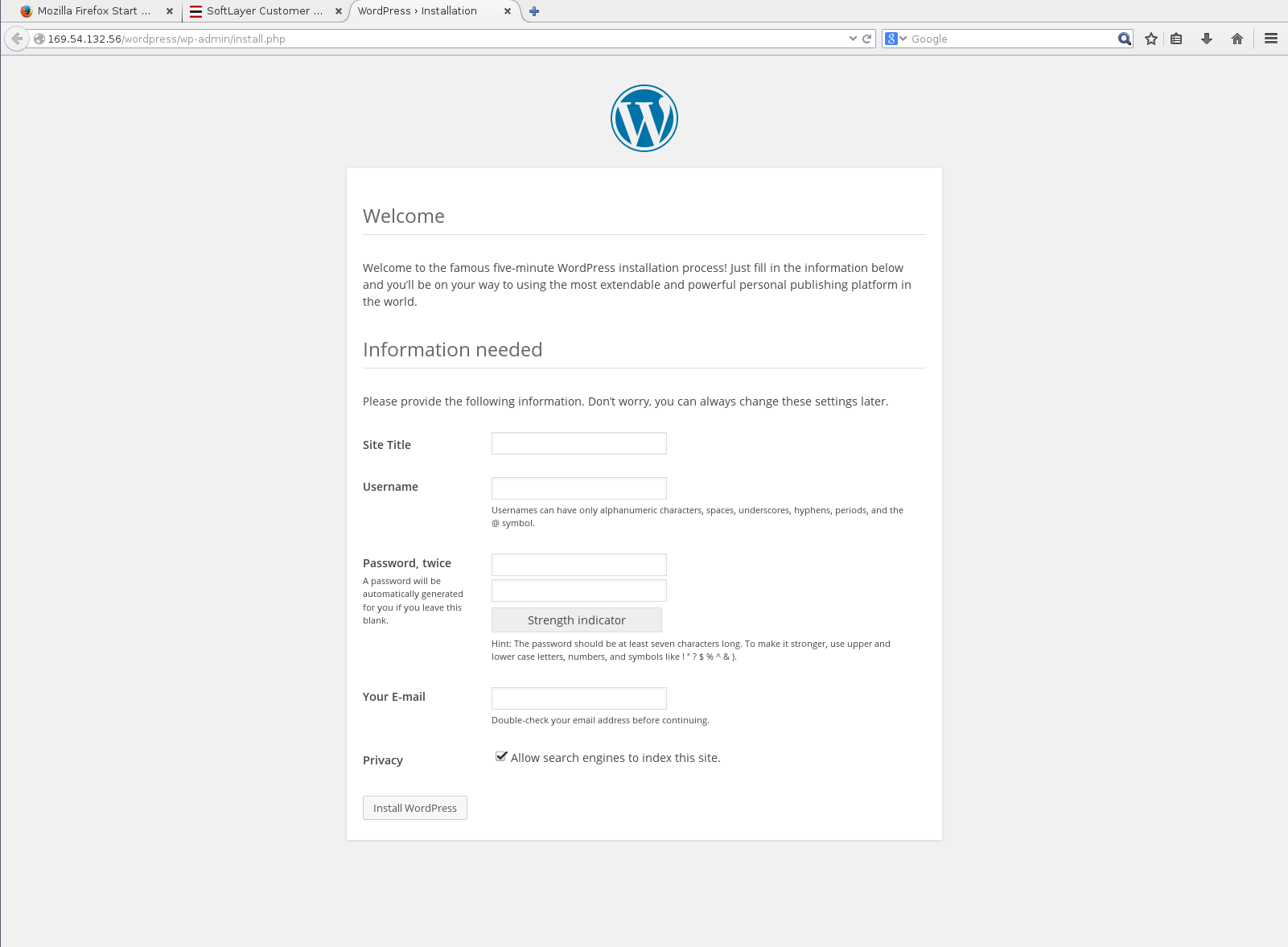
以上で追加設定は終了です。WordPressのセットアップに入りましょう。

### WordPressのセットアップ

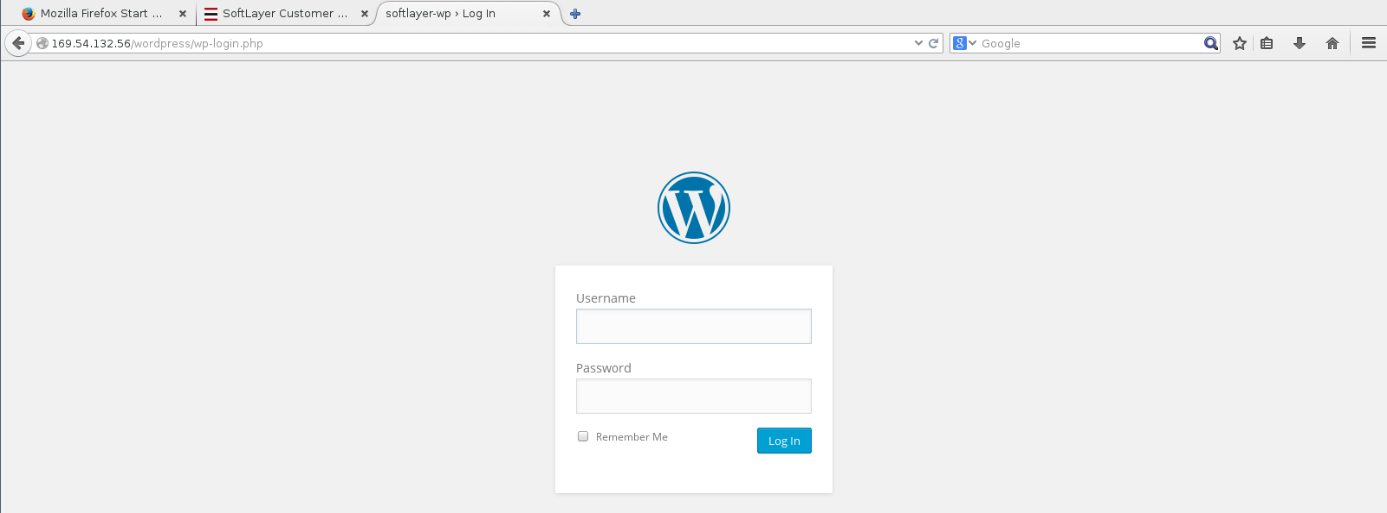
WordPressのセットアップは、ブラウザのウィザード越しに行います。ブラウザで以下のURLを開いてください。

|  |
| --- |
| [http://WordPress](http://wordpress)サーバのPublic IP Address/ |

すると、以下の様なインストール画面が表示されます。上からサイト名、管理用ユーザー名、管理用パスワード(2回入力)、Eメールアドレスを順に入力してください。



これでWordPressのセットアップは終了です。設定したパスワードでWordPressにログインしましょう。



### 二台目のセットアップ

二台目以降のWordPressノードのセットアップは非常に簡単です。二台目以降は、一台目と同じスペック、同じProvisioning Scriptを用いてデプロイし、一台目と同様に/etc/WordPress/wp-config.phpにDBサーバの設定を施せば、すぐにWordPressサーバとして稼働します。

### SoftLayer Object　Storageの設定

標準の設定では、WordPressはアップロードした画像をローカルに保持します。しかし、複数のWordPressサーバを構築して負荷分散を行う場合はアプロードされた画像を各サーバ間で共有する必要があります。そこで、各WordPressサーバ間で、SoftLayerのObject Storageを介してアップロードされたコンテンツを共有します。まずは各サーバ群と同じデータセンターにObject Storageを注文して、利用できる様にしてください。

|  |  |
| --- | --- |
| **注意** | Object Storageの注文と利用に関しては、SoftLayerハンズオン ストレージ編の5章で詳解しています。併せてお読みください。  <http://ibm.biz/slhandson22>  IBM から提供されているアカウントを利用している場合は、Object Storage の注文作業は行わずに、既存の  Object Storage アカウント「IBMOS363380-14」を利用してください。 |

#### Object Storageのセットアップ

管理ポータルから、Storage - Object Storageを選択してください。 アカウントの一覧が表示された後に、自動で Object Storage ユーザーの管理ポータルに遷移します。自動で遷 移しない場合は、オブジェクトストレージのアカウント名の部分をクリックしてください。IBM から提供されているアカウントを利用している場合は、Object Storage アカウント「IBMOS363380-14」を利用してください。

#### Locationの選択

Object Storage をデプロイ出来るデータセンター一覧が表示されます。2014 年 7 月現在、Dallas、Hong Kong、Singapore、San Jose、Amsterdam、London、Toronto の 7 箇所の中から選択することができます。 今回は「San Jose 1..」を選択します。

#### Credentialsの確認

管理ポータルから、Storage – Object Storage – Object Storageのアカウント名 – ロケーション (San Jose) で、 [View Credentials]をクリックすることで、オブジェクトストレージへアクセスする為の ユーザー名やパスワード、エンドポイントといった重要な情報を確認することができます。 [Authentication Endpoint]のうち、Public:となっているエンドポイントはインターネット上ならどこからでも、 Private:となっているエンドポイントは SoftLayer のネットワーク内部からのみ利用することができます。但し、 SoftLayer のネットワーク内であれば Hong Kong の Object Storage を London のサーバから参照することも可 能です。これらの情報はハンズオン中で利用しますので、テキストエディタ にコピーしておくと後で便利です。

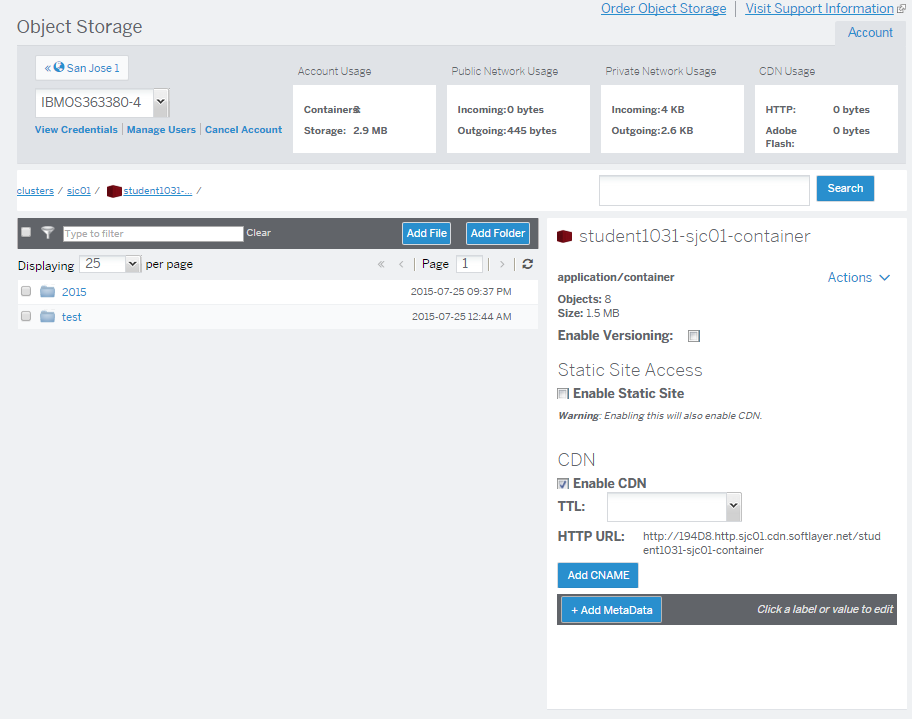
#### コンテナの追加

SoftLayer の Object Storage は、一つの Object Storage アカウントに複数地域のデータセンターが紐付けられ、 一つのデータセンター内部では更に container という形で紐付けされます。新しくコンテナを追加するには、 Object Storage の DC ごとの管理ページにて、ページ中央の[Add a container]をクリックします。 コンテナの名前を聞かれるので、適切に入力をし、OK をクリックしてください。

注意: IBM から貸与しているアカウントを利用している場合は、”アカウント名-Datacenter名-Container”という名前で Container を作成ください。例: student1031-sjc01-container

#### CDNの有効

SoftLayer Object StorageはCDNと連携させることができます。必要に応じて、Enable　CDNをチェックし、CDNを有効にしてください。



#### Cloudfuseのインストール

Object Storageが利用できるようになったら、各WordPressサーバにObject Storageをマウントして利用するためのCloudfuseというソフトウェアを導入します。まずはCloudfuseのビルドに必要なソフトウェアをインストールします。

|  |
| --- |
| # yum groupinstall "Development Tools" -y  # yum install libxml2-devel libcurl-devel fuse-devel openssl-devel json-c-devel -y |

Cloudfuseのソースコードをダウンロードします

|  |
| --- |
| # wget -O cloudfuse.tar.gz <https://github.com/redbo/cloudfuse/tarball/master> |

展開し、インストールします

|  |
| --- |
| # tar -xvf cloudfuse.tar.gz  # cd redbo-cloudfuse-21358f1  # ./configure && make && make install |

インストールが完了した後、cloudfuseが利用する設定ファイルを、rootのホームディレクトリに.cloudfuseという名前で作成します。ファイル名の先頭のドットを入れ忘れないようにご注意ください。

|  |
| --- |
| # vi ~/.cloudfuse |

.cloudfuseには、管理ポータルより、「Storage」>「Object Storage」>「sjc01」>「View Credentials」で確認できるユーザー名やパスワード、エンドポイント情報を記述します。情報は適宜ご自分の内容と読み替えてください。

|  |
| --- |
| 記述例:  username=SLOS351234-2:SL351234  api\_key=33e2b\*\*\*\*\*  authurl=https://xxxxx.objectstorage.service.networklayer.com/auth/v1.0/ |

CloudfuseでObject Storageをマウントして、WordPressにアップロードされた画像を共有します。その際、画像アップロードフォルダにNginxが書き込める必要があるため、NginxのユーザーIDでObject Storageをマウントしましょう。

|  |
| --- |
| # id nginx  uid=497(nginx) gid=498(nginx) groups=498(nginx)  # cloudfuse -o uid=497 -o allow\_other /usr/share/wordpress/wp-content/uploads  # ls /usr/share/wordpress/wp-content/uploads  student1031-sjc-container |

画像をSoftLayer Object Storageの指定コンテナにアップロードするようにwp-config.phpに、下記を追加します。

|  |
| --- |
| [root@backweb ~]# vi /etc/wordpress/wp-config.php  /\*\* Change uploads folder to SoftLayer object storage\*/  define('UPLOADS', '/wp-content/uploads/<コンテナ名 例: student1031-sjc01-container>'); |

以上で、複数サーバ間のコンテンツ共有が完了しました。アップロードされたコンテンツは、Object Storageを介してバックエンドのWordPressサーバ群同士において自動で共有されます。

## Load Balancer

### Load Balancerの導入と設定

次は負荷分散機構のLoad Balancerをデプロイしてみましょう。Load balancerには新進気鋭のWebサーバ、Nginxを利用します。Nginxは負荷分散と同時に流通しているコンテンツをキャッシュできるため、更なる通信負荷の低減を見込むことができます。Load Balancer用のProvisioning Scriptは以下に公開されています。

管理ポータルより、下記スペックの時間課金仮想インスタンス（**Virtual Server (public node) - Hourly**）を作成してください。

|  |  |
| --- | --- |
| Data Center | San Jose 1 |
| Operating System | CentOS 6.x - Minimal Install (64 bit) |
| UPLINK PORT SPEEDS | 100 Mbps Public & Private Network Uplinks |
| Advanced Monitoring | None |
| Monitoring | Host Ping |
| Response | Automated Notification |
| Provisioning Scripts  Existing Script:  URL 1: | ブラウザで下記のURLを参照してスクリプトを確認してから、コピペで入力  (new URL)  https://raw.githubusercontent.com/kkitase/handson/master/tools/frontweb.sh |
| SSH Key | 登録したSSH keyのラベルを選択 例: student1031 |
| Host Name | ご自身のアカウントを利用している方: 任意  **IBMから貸与したアカウントを利用している方: アカウント名-lb1** |
| Domain Name | ご自身のアカウントを利用している方: 任意  **IBMから貸与したアカウントを利用している方: handson.jp** |

※他はデフォルト設定、講師から何か指定がある場合はそちらに従ってください。

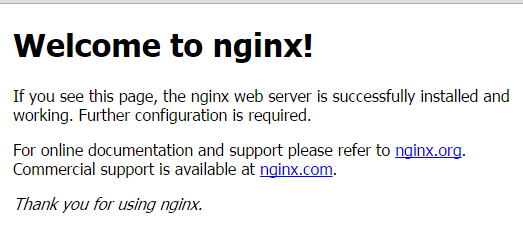
Provisioning Scriptは以下の作業を自動化します。

1. Zabbix エージェントの導入
2. Nginx Webサーバのインストール

インストールが終わったら、Nginx web serverが動いていることを確認しましょう。

|  |
| --- |
| <http://Load> BalancerマシンのPublic IP Address/ |

Nginxのデフォルトページが表示されていればインストールが無事完了しています。



Load Balancerが配下のWordPressサーバに要求を転送する設定をNginxに施します。

|  |
| --- |
| [root@frontweb ~]# vi /etc/nginx/conf.d/default.conf  編集前  location / {  root /usr/share/nginx/html;  index index.html index.htm;  }  ↓  編集後  location / {  proxy\_pass <http://wpnode/>;  proxy\_cache wpcache;  proxy\_cache\_valid 200 302 60m;  proxy\_cache\_valid 404 20m;  # root /usr/share/Nginx/html;  # index index.html index.htm;  } |

更に、設定ファイル末尾の}の行の後に、以下の設定を追加します。

|  |
| --- |
| upstream wpnode {  server WordPressサーバ1台目のPrivate IP Address;  server WordPressサーバ2台目のPrivate IP Address;  } |

続いてキャッシュの設定を施します。ファイルの末尾近くに設定を追加します。

|  |
| --- |
| [root@frontweb ~]# vi /etc/nginx/nginx.conf  編集前  #gzip on;  include /etc/nginx/conf.d/\*.conf;  ↓  編集後  #gzip on;  proxy\_cache\_path /var/cache/nginx/cache levels=1:2 keys\_zone=wpcache:4m max\_size=50m; (一行で)  include /etc/nginx/conf.d/\*.conf; |

設定変更を終えたら、設定ファイルのsyntaxを確認の上、問題なければreloadして反映します。

|  |
| --- |
| [root@frontweb ~]# nginx -t  [root@frontweb ~]# service nginx reload |

これでLoad Balancerの設定は完了です。再度以下のURLを開いてみましょう。

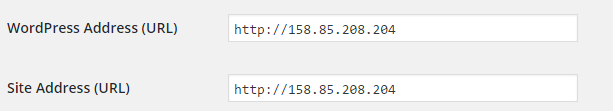
|  |
| --- |
| http://<Load BalancerのパブリックIPアドレス>/ |

適切に設定されていれば、背後のWordPressサーバに設定したWordPressのページが表示されます。

適切に表示されれば、Wordpressの管理者アカウントでログインし、「Settings」 – 「General」で、下記設定を変更してください。

WordPress Address (URL): http://<Load BalancerのパブリックIPアドレス>

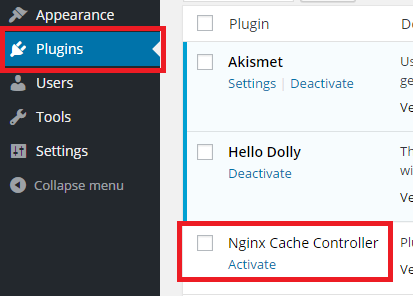
Site Address (URL): http://<Load BalancerのパブリックIPアドレス>



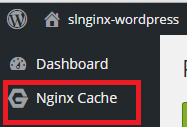
#### Nginx cache controllerの導入

コンテンツのキャッシュはサーバの応答速度の向上に非常に有用です。しかし、内容が更新されたにも関わらず古い内容がキャッシュとして残り続けるのは問題です。これを解決するために、WordPressにNginx cache controllerプラグインを導入します。このプラグインを導入することで、記事を更新した際に、リバースプロキシに対して古いキャッシュを破棄し内容を更新するよう指示することができます。

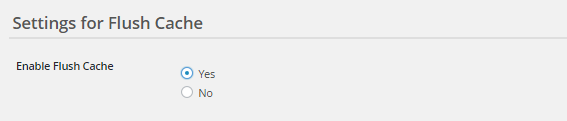
　Nginx cache controllerプラグインは既にインストールされているので、WordPressの管理メニューから有効化できます。管理メニューのPluginからNginx Cache Controllerを選択し、Activateしてください。



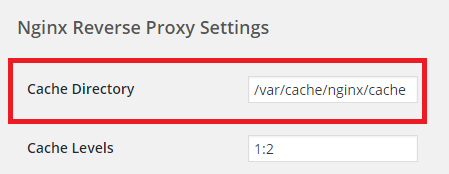
有効になると、左上にNginx Cacheメニューが出現します。



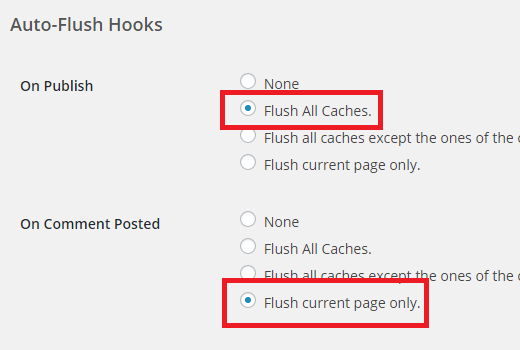
Settings for Flush Cacheのカテゴリで、Enable Flush Cache を Yesと選択します。



前述で設定した通り、Cache Directoryを変更します。



コンテンツ投稿時の振る舞いを確認します。記事が投稿された時にはキャッシュを全て、コメントが投稿された場合は該当のページのキャッシュのみを削除します。



最後に、Save Changesをクリックし、キャッシュ制御の設定は完了です。

#### コンテンツキャッシュの同期設定

Nginx Cache Controllerはコンテンツ更新時にキャッシュを制御できますが、本プラグインはWordPressサーバとリバースプロキシが同一のサーバで動作する環境を想定しているため、本ハンズオンのようにリバースプロキシとWordPressサーバを分離している環境では適切に動作しません。そこで、lsyncdを用いてフロントエンドのリバースプロキシとバックエンドのWordPressサーバ間でキャッシュファイルを同期し、WordPressサーバがNginx Cache Controllerを用いてリバースプロキシのキャッシュファイルを削除できるようにします。

リバースプロキシ、WordPressサーバともにlsyncdのセットアップはProvisioning Script内で完了しています。通信に必要な鍵ファイルを適切に配置し、lsyncdの設定ファイルを用意して自動同期環境を構築しましょう。同期はNginxユーザーで行う必要があるため、権限を適切に設定する必要があります。自動で同期を行うために、作業用インスタンス上でパスワードを要求されない空の鍵ペアを作成し、ロードバランサーとWordPressノードの双方に配置します。

|  |
| --- |
| [root@workingvm ~]# ssh-keygen -f emptypass  Generating public/private rsa key pair.  Enter passphrase (empty for no passphrase): (何も入力せずEnter)  Enter same passphrase again: (何も入力せずEnter)  Your identification has been saved in /root/.ssh/emptypass.  Your public key has been saved in /root/.ssh/emptypass.pub.  The key fingerprint is:  cc:5e:59:21:44:99:42:e3:26:5e:73:41:bb:70:af:45 root@workingvm  The key's randomart image is:  +---[RSA 2048]----+  | .o+\*o. |  | ...o+ . |  | . \*.+ E |  | . \* = \* |  | . S + o |  | . . o |  | . . |  | |  | |  +-----------------+  [root@ workingvm ~]# scp -r ~/.ssh root@[ロードバランサーのPublic IP Address]:/tmp  [root@ workingvm ~]# scp -r ~/.ssh root@[WordPressサーバ1のPublic IP Address]:/tmp  [root@ workingvm ~]# scp -r ~/.ssh root@[WordPressサーバ2のPublic IP Address]:/tmp |

作成した鍵ペアを各サーバで適切に配置します。

|  |
| --- |
| [root@frontweb ~]# ssh root@[ロードバランサーのPublic IP Address]  [root@lb ~]# mv /tmp/.ssh/emptypass.pub /tmp/.ssh/authorized\_keys  [root@lb ~]# cp -r /tmp/.ssh /var/cache/nginx  [root@lb ~]# chown -R nginx:nginx /var/cache/nginx/.ssh |

同様の設定をWordPressサーバ1と2でも行ってください。

最後に、lsyncdの設定ファイルを編集し、同期を行います。ロードバランサーは配下のWordPressサーバ1と2の両方と動機する設定を、WordPressノードはロードバランサーノードとのみ同期する設定を行います。まずはロードバランサーノードの設定です。

|  |
| --- |
| [root@lb ~]# wget -O /etc/lsyncd.conf <https://raw.githubusercontent.com/kkitase/handson/master/lsyncd-wp> (1行で)  [root@lb ~]# vi /etc/lsyncd.conf  settings {  logfile = "/var/log/lsyncd/lsyncd.log",  statusFile = "/var/log/lsyncd/lsyncd-status.log",  statusInterval = 20  }  sync {  default.rsync,  delay = 0,  source="/var/cache/nginx/cache",  target="nginx@wordpressサーバ1のIPアドレス:/var/cache/nginx/cache/",  rsync = {  rsh = "/usr/bin/ssh -i /var/cache/nginx/.ssh/emptypass -o StrictHostKeyChecking=no"  },  delete = "running",  init = false  }  sync {  default.rsync,  delay = 0,  source="/var/cache/nginx/cache",  target="nginx@wordpressサーバ2のIPアドレス:/var/cache/nginx/cache/",  rsync = {  rsh = "/usr/bin/ssh -i /var/cache/nginx/.ssh/emptypass -o StrictHostKeyChecking=no"  },  delete = "running",  init = false,  } |

WordPressノードは1、2に設定ファイルを追加し、同期するためのディレクトリを作成してください。

|  |
| --- |
| [root@backweb ~]# wget -O /etc/lsyncd.conf <https://raw.githubusercontent.com/kkitase/handson/master/lsyncd-wp> (1行で)  [root@backweb ~]# vi /etc/lsyncd.conf  settings {  logfile = "/var/log/lsyncd/lsyncd.log",  statusFile = "/var/log/lsyncd/lsyncd-status.log",  statusInterval = 20  }  sync {  default.rsync,  delay = 0,  source="/var/cache/nginx/cache",  target="nginx@ロードバランサーのIPアドレス:/var/cache/nginx/cache/",  rsync = {  rsh = "/usr/bin/ssh -i /var/cache/nginx/.ssh/emptypass -o StrictHostKeyChecking=no"  },  delete = "running",  init = false  }  [root@backweb ~]# mkdir /var/share/nginx/cache  [root@backweb ~]# chown –R nginx:nginx /var/cache/nginx/cache |

以上でキャッシュファイルを同期するlsyncdの設定が完了しました。各サーバでlsyncdを起動すると、キャッシュファイルが同期されるようになります。

|  |
| --- |
| [root@frontweb ~]# /etc/init.d/lsyncd start  Starting lsyncd: [ OK ] |

## Zabbixサーバへの登録

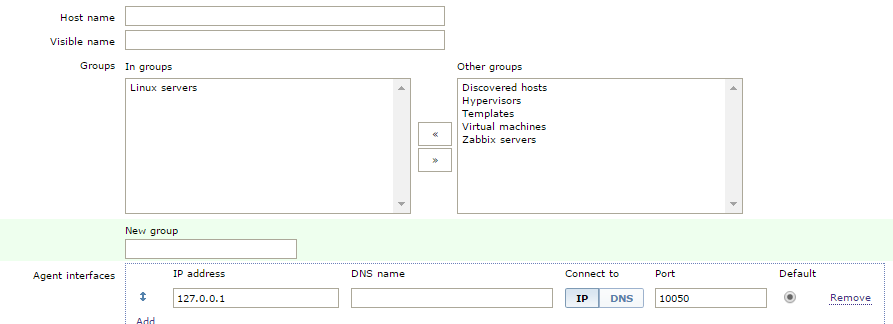
各サーバ上ではProvisioning Scriptで導入されたZabbixエージェントが起動しています。Zabbix監視サーバにこれらの子サーバ群を登録し、一括で監視できるよう設定を施しましょう。上部メニューのConfiguration→Hostsの順にクリックしましょう。



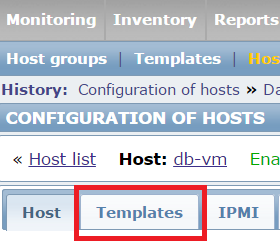
ノード一覧が表示されるので、右端のCreate hostを選択します。



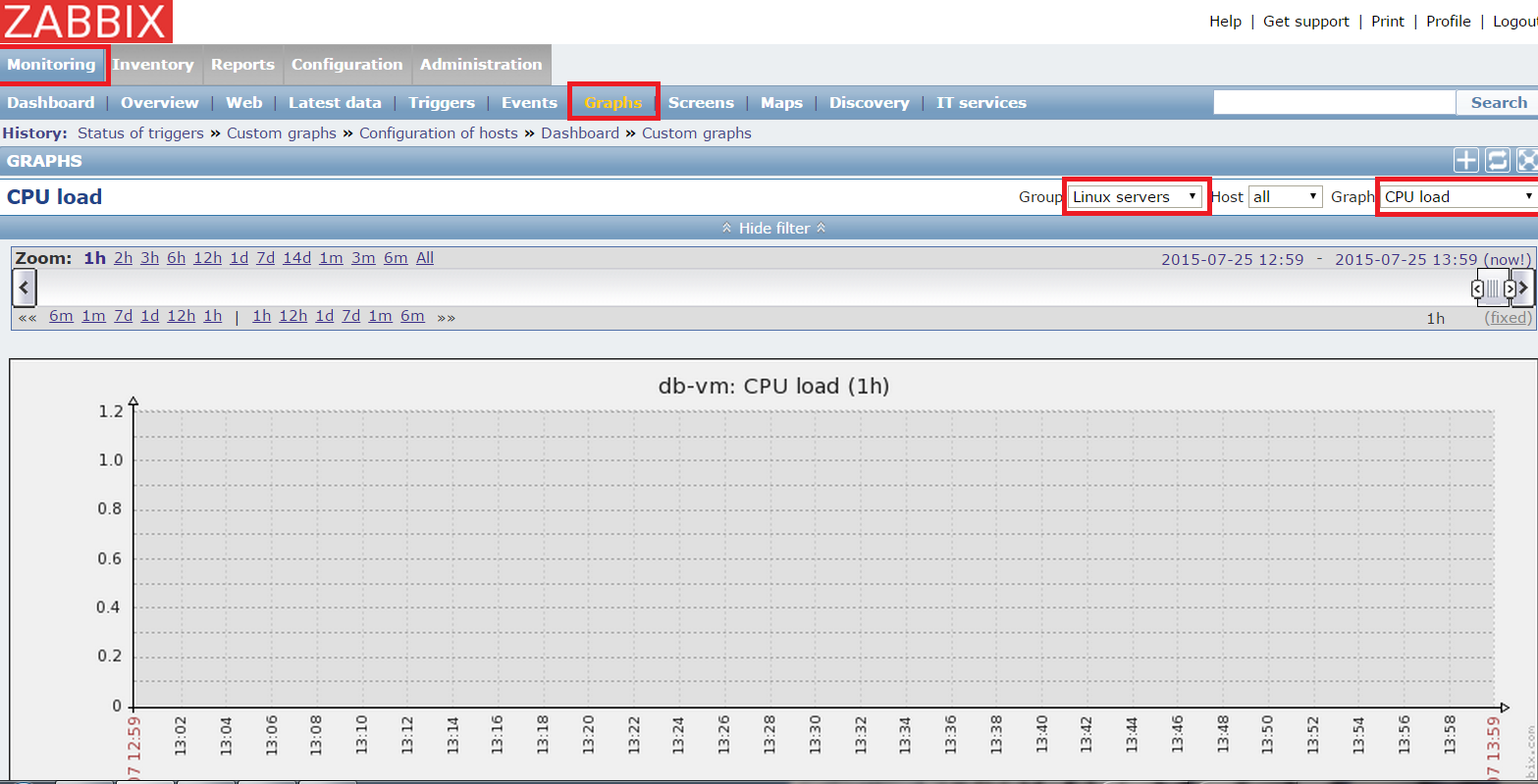
設定画面が表示されるので、内容を登録します。Host Nameに各サーバのホスト名を、Visible NameにはZabbix上で表示される名前を指定します。Groupsでは、右にあるLinux servesをクリックして << ボタンを押すことで選択できます。下のAgent InterfaceのIPアドレスには各ノードのSoftLayerのPrivate IP Addressを登録してください。入力が完了したら、最下段のAddをクリックすることで、ノードのZabbixへの登録が完了します。



監視するノードを追加したら、次は監視する項目を追加します。今度は、Templatesタブをクリックします。テンプレート追加画面を開いたら、画面中央のSelectを選択し監視する内容を選択します。データベースサーバはTemplate Add MySQLに、システム全体の監視をする場合はTemplate OS Linuxにチェックを入れて最下段のSelectを押してください。ポップアップウインドウが閉じた後、下部のAddをクリックし、更に下部のUpdateをクリックしてください。



以上で、選択したTemplateの内容がZabbixに集積され、グラフ化されてモニタリングできます。Monitoring -> Graphs -> Groupで「Linux servers」を選択 -> Graphで確認したい項目を選択することで、収集した情報の推移を確認することができます。



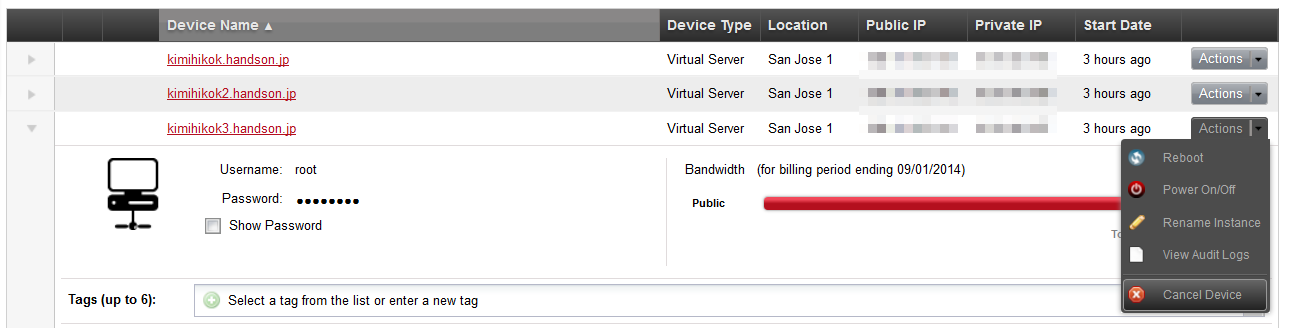
# 最後に

本ハンズオンで作成した、全ての仮想インスタンスとオブジェクトストレージを削除して終了してください。その他イメージなどを作成された方は、イメージなども削除してください。

（本ハンズオンで利用した環境を、ハンズオン終了後も利用される方は、残しておいても問題ありませんが、課金されることをご理解いただければと思います）

## 仮想インスタンスの削除

[Device] - [Device list] - 自分のサーバ - [Actions] - [Cancel Device]



## Object Storage内のファイルの削除

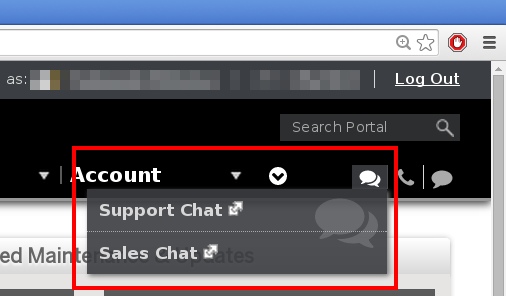
管理ポータルからは、ファイルの一括操作を行うことができないので、Webサーバーから、下記のように自分のコンテナごと削除してください

（例: rm -rf /usr/share/wordpress/wp-content/uploads/student1031-sjc01-container）

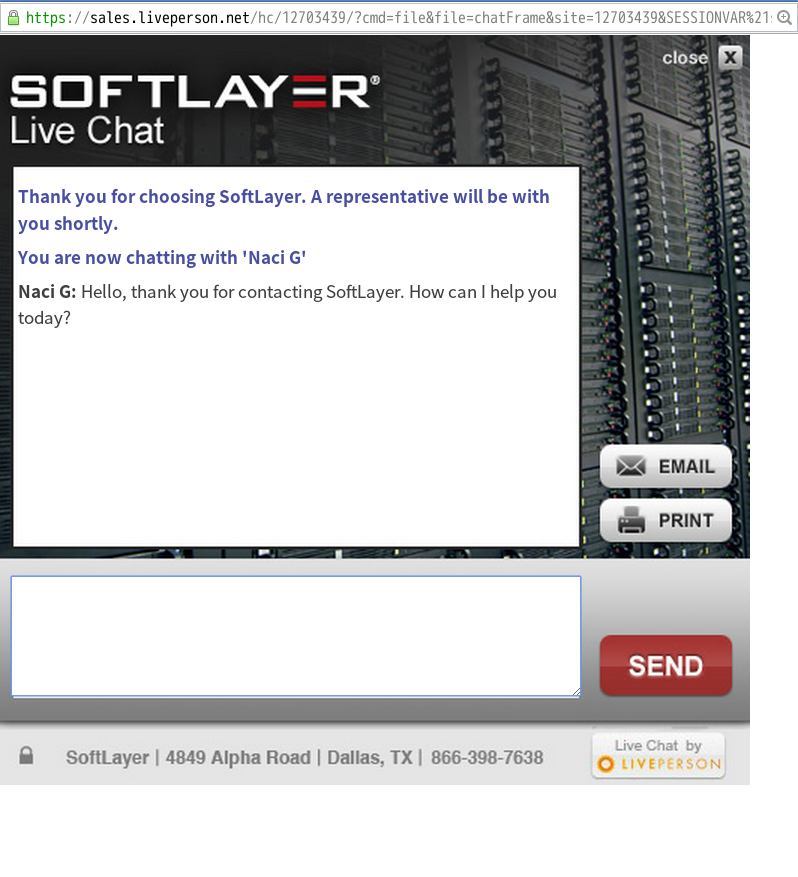
# Appendix

## サポートチャットとチケット

SoftLayerの管理ポータル右上部には、専任のスタッフと24時間直通で利用できるチャットを開始するためのボタンが配置されています。

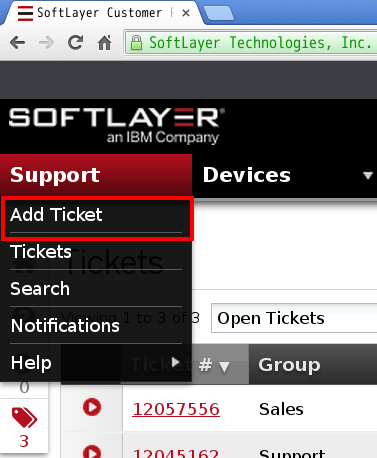


このボタンをクリックすると、新しいウィンドウがポップアップし、10秒ほどで待機している専任スタッフとの英語チャットが開始されます。



サポートチャットのカバーする範囲はあくまでSoftLayerの環境での一般的な内容に留まりますので、個々の環境での個々の事情までは応対してくれませんが、SoftLayer上で提供されている機能のカタログスペックや、実行しようとしている事の実現の可否に関しては、公式ドキュメントへのリンクと併せて非常に的確に答えてくれますので、SoftLayer上での活動に疑問を持った際に、最初に頼ることになる存在となるでしょう。

また、サポートチャットで解決できないと判断された問題は、サポートチャットの担当者からチケットを発行して問題をSoftLayer運営にエスカレーションする事を薦められる事もあります。



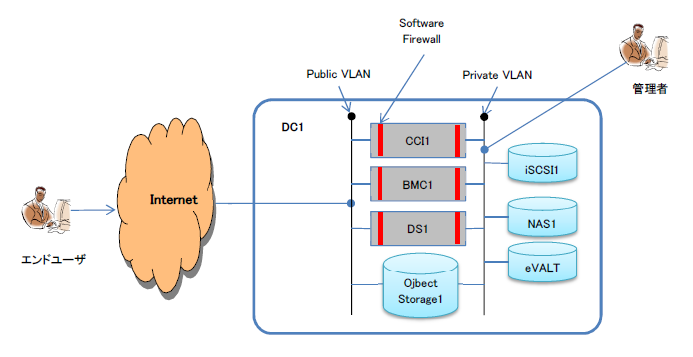
チケットを発行する場面というのは、例えば10分あれば終了すると記述されている機能のデプロイが一時間経過してもなお終了していない場合であったり、ネットワーク不調だったりと、現在の挙動と提供されるべき内容に相違があると疑わしい場合になります。

チケットを作成すると、それらのチケットは問題に関連する部門のスタッフが適切に確認し、更に上位の部門との折衝が必要な場合は、スタッフが適切な部門にチケットを転送し、また必要な場合は利用者側の環境の調査を併せて行いながら状況の確認と問題の解決に向かいます。

## SolftLayerのネットワーク構成例

**[基本構成例]**

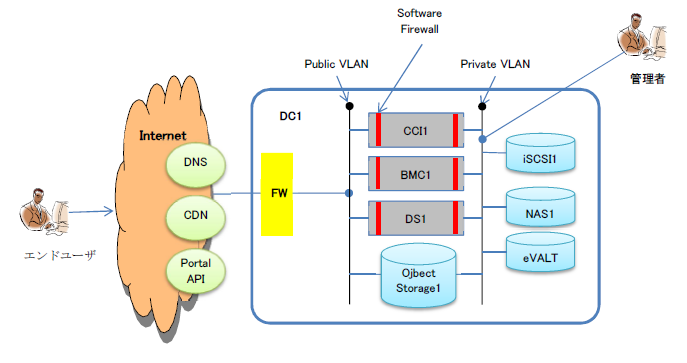
サーバの起動直後にソフトウェアファイアウォールで最低限のセキュリティを確保している構成です。なお、管理者はSSL VPNでサーバに接続しています。



CCI: クラウドインスタンス、BMC: ベアメタルインスタンス、DS: ベアメタルサーバ

**[ハードウェアファイアウォール構成例]**

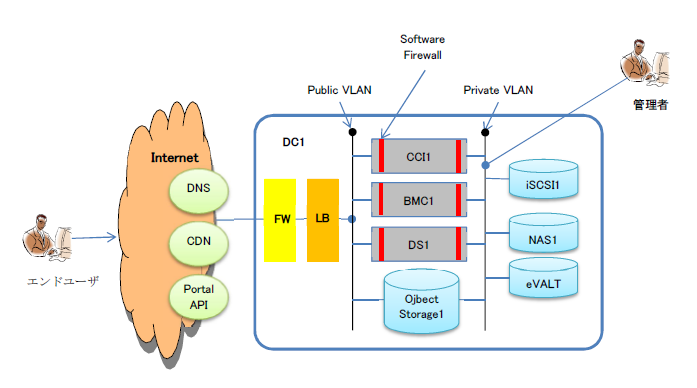
ハードウェアファイアフォールでセキュリティをより強化した構成の例です。



CCI: クラウドインスタンス、BMC: ベアメタルインスタンス、DS: ベアメタルサーバ

**[ハードウェアファイアウォール+ロードバランサー構成例]**

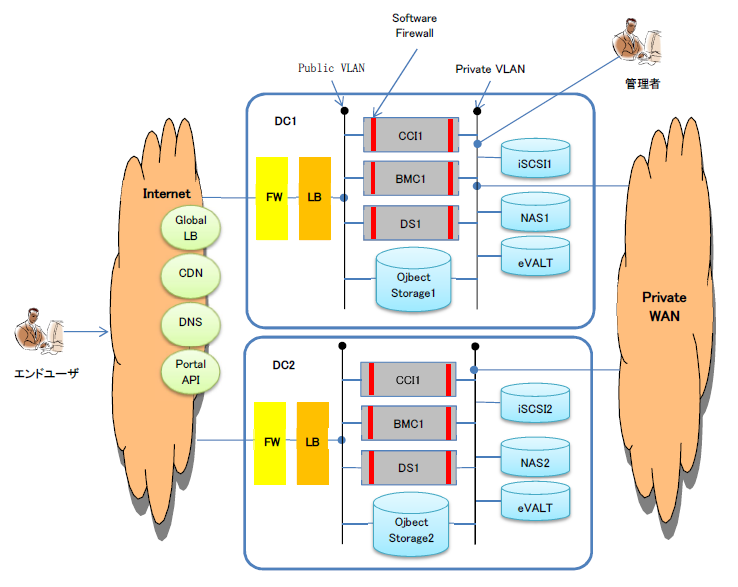
必要に応じてロードバランサーを追加できます。



CCI: クラウドインスタンス、BMC: ベアメタルインスタンス、DS: ベアメタルサーバ

**[マルチデータセンターへの分散システム構成例]**

複数のデータセンターにまたがる分散システム構成も可能です。SoftLayerのデータセンター間は専用線で結ばれており、無料でPrivate WANが構成できます。



CCI: クラウドインスタンス、BMC: ベアメタルインスタンス、DS: ベアメタルサーバ

## Apacheを利用したWordPressサーバの構築

本ハンズオンではWordPress用のHTTPサーバにNginxを利用しましたが、Apacheを用いることもできます。Apacheを利用したい場合は以下のProvisioning Scriptをご利用ください。

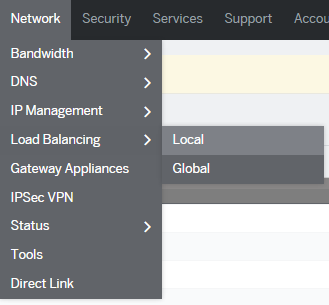
|  |
| --- |
| https://raw.githubusercontent.com/kkitase/handson/master/backweb\_apache.sh |

Apacheを利用すると、WordPressのパスが以下の通り変化します。

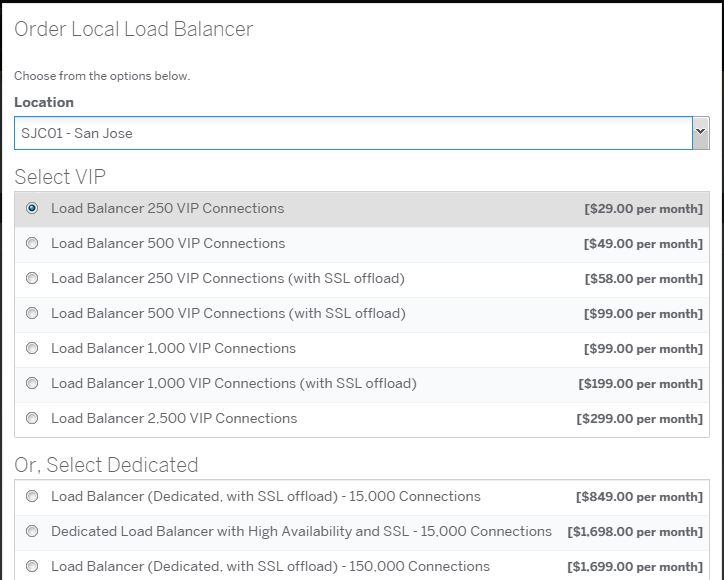
|  |
| --- |
| http://仮想インスタンスのPublic IP Address/  ↓  http://仮想インスタンスのPublic IP Address/WordPress/ |

## Local Load Balancer

本ハンズオンではロードバランサーとしてNginxを利用しましたが、SoftLayerには専用のロードバランサーアプライアンスが容易されており、ポータルから注文することができます。Network -> Localと選択し、右上の「Order Local Load Balancer」を選択します。

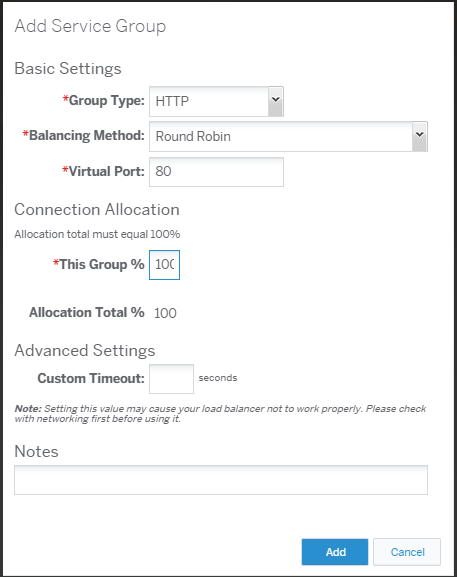


Local Load Balancerは1秒ごとに処理できる通信量やSSLのオプション等で料金が変わります。お好みのプランを選択し、Locationを使いたいデータセンターと合わせるのを忘れないようにしましょう。



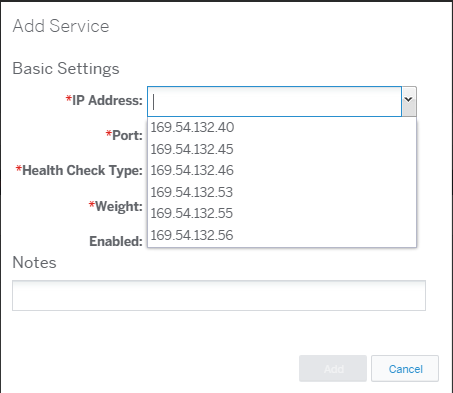
Local Load Balancerを注文したら、ロードバランスの設定を行います。再度Network -> Load Balancing -> Localを開き、注文したVIPアドレスをクリックしてください。次に、一番右の「Add Service Group」をクリックして

Group TypeにHTTPを選択し、This Groupに100と入力してAddを選択します。次は、当該のVIPに要求が来た際に要求が割り振られるIPアドレス群を設定します。



グループの追加が完了したら、次はサーバの追加です。今度は右隅のAdd Serviceをクリックします。

IP Addressの欄でグローバルIPアドレスが一覧表示されるので、Reverse Proxyのアドレスを選択します。Portには80、その他の欄はデフォルトのままで問題ありません。



複数のReverse ProxyのグローバルIPアドレスを追加していきましょう。

以上で、Load Balancing - Reverse Proxy - WordPressサーバという多段構成の構築が完了しました。Reverse ProxyサーバやWordPressサーバを迅速にデプロイ可能なので、急な負荷増大にも迅速に対応可能です。

メモ用途にお使いください

メモ用途にお使いください

© Copyright IBM Corporation 2016

IBM Global Services

Route 100

Somers, NY 10589 U.S.A.

Produced in the United States of America

May 2014

IBM, the IBM logo and ibm.com are trademarks of International Business Machines Corporation in the United States, other countries or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or TM), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. Other product, company or service names may be trademarks or service marks of others. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at ibm.com/legal/copytrade.shtml

This document is current as of the initial date of publication and may be changed by IBM at any time. Not all offerings are available in every country in which IBM operates.

The performance data discussed herein is presented as derived under specific operating conditions. Actual results may vary. It is the user’s responsibility to evaluate and verify the operation of any other products or programs with IBM products and programs. THE INFORMATION IN THIS DOCUMENT IS PROVIDED “AS IS” WITHOUT ANY WARRANTY, EXPRESS OR IMPLIED, INCLUDING WITHOUT ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND ANY WARRANTY OR CONDITION OF NON-INFRINGEMENT. IBM products are warranted according to the terms and conditions of the agreements under which they are provided.