

전자정부표준프레임워크 기반의 Open PaaS 개발

BOSH 설치 가이드(설치패키지)

|  |  |  |
| --- | --- | --- |
| 작성자 | 문철호 (인) | 2015.7.25 |
| 검토자 | 김기현 (인) |  |
| 승인자 |  |  |

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

개 정 이 력

| 버전 | 작성일 | 변경내용[[1]](#footnote-1) | 작성자 | 승인자 |
| --- | --- | --- | --- | --- |
| 0.1 | 2015.5.25 | 최초작성 | 문철호 |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

목차

[1. 문서 개요 5](#_Toc428813589)

[*1.1.* *목적* 5](#_Toc428813590)

[*1.2.* *범위* 5](#_Toc428813591)

[*1.3.* *참고자료* 5](#_Toc428813592)

[2. Basic BOSH Workflow 6](#_Toc428813593)

[3. BOSH 설치 패키지 다운로드 7](#_Toc428813594)

[4. Install BOSH CLI 8](#_Toc428813595)

[*4.1.* *Ruby 설치* 8](#_Toc428813596)

[*4.2.* *BOSH CLI설치* 9](#_Toc428813597)

[5. Install MicroBOSH 10](#_Toc428813598)

[*5.1.* *설치 절차* 10](#_Toc428813599)

[*5.2.* *IaaS 환경설정 - OpenStack* 10](#_Toc428813600)

[*5.3.* *배포 Manifest 작성* 12](#_Toc428813601)

[*5.4.* *MicroBOSH 배포* 13](#_Toc428813602)

[6. Install BOSH 16](#_Toc428813603)

[*6.1.* *설치 절차* 16](#_Toc428813604)

[*6.2.* *BOSH Stemell 업로드* 16](#_Toc428813605)

[*6.3.* *BOSH Release 업로드* 18](#_Toc428813606)

[*6.4.* *배포 Manifest 작성* 20](#_Toc428813607)

[*6.5.* *BOSH 배포* 25](#_Toc428813608)

Executive Summary

본 문서는 BOSH에 대한 설치 가이드입니다. BOSH를 설치하고 기본 동작하는 부분을 설명하였습니다.

본 문서는 다음과 같은 내용들을 포함한다.

* BOSH CLI 설치
* MicroBOSH 설치
* BOSH 설치

# 문서 개요

## ***목적***

클라우드 환경에 서비스 시스템을 배포할 수 있는 BOSH는 릴리즈 엔지니어링, 개발, 소프트웨어 라이프사이클 관리를 통합한 오픈소스 프로젝트로 본 문서에서는 로컬 환경에서 MicroBOSH와 BOSH를 설치하여 기능을 테스트할 수 있는 환경을 구축하는 데 목적이 있다..

## ***범위***

본 가이드에서는 Linux 환경(Ubuntu 14.4)을 기준으로 설치환경 구성 및 BOSH CLI 설치하고, 이를 이용하여 MicroBOSH 및 BOSH를 설치하는 것을 기준으로 작성하였다.

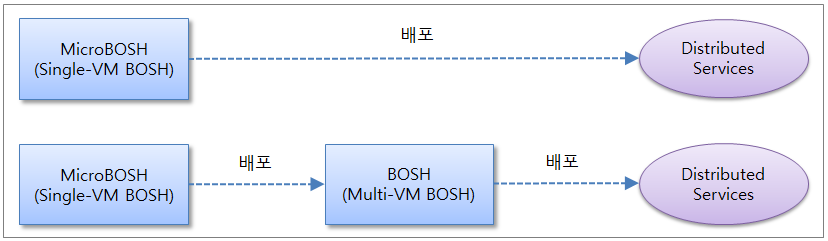
## ***참고자료***

본 문서는 Cloud Foundry의 BOSH Document를 참고로 작성하였습니다.

BOSH Document : <http://bosh.io>

# Basic BOSH Workflow

본 문서의 목적에서도 언급했듯이 BOSH는 클라우드 환경에 서비스를 배포 관리하는 소프트웨어로 BOSH자체도 클라우드에 배포되어야 하는 서비스로 배포는 아래와 같이 2가지 유형이 존재하니 설치 환경에 따라 MicroBOSH 또는 BOSH를 설치하여 서비스를 운영할 수 있다.

MicroBOSH (Single-VM BOSH)는 소프트웨어 배포 및 관리를 하나의 VM에서 수행하는 소프트웨어로 MicroBOSH를 배포하기 위해서는 Stemcell 이미지와 배포 Manifest파일이 필요하고 별도의 소프트웨어 Release는 필요로하지 않는다. (설치에 필요한 Release들은 Stemcell내부에 탑재되어 있다. )

MicroBOSH를 설치한 후에는 선택에 따라 MicroBOSH를 이용하여 BOSH (Multi-VM BOSH)를 배포하거나 또는 다른 서비스들을 배포할 수도 있다.

# BOSH 설치 패키지 다운로드

OpenPaaS에서 제공하는 설치 관련 패키지들을 다운로드 받습니다.

# 설치 패키지 다운로드 주소

|  |
| --- |
| **http://openpaas.org (?)** |

다음 표는 MicroBOSH와 BOSH를 설치하기 위해 필요한 패키지 및 구성항목으로 상기 주소에서 모두 다운로드 받습니다.

|  |  |  |  |
| --- | --- | --- | --- |
| **구성 항목** | | | **설명** |
| **구분** | **디렉토리** | **파일명** |
| 공통 | OpenPaaS-Env/unzip | unzip\_6.0-18\_amd64.deb | Off-line에서  Ubuntu unzip Package |
| OpenPaaS-Env/ruby-dependency | bosh-ruby-dependency-packages.tar.gz | Off-Line에서  Ruby Dependency |
| OpenPaaS-Env/ruby | ruby-2.2.2.tar.gz | Ruby 2.2.2 버전 |
| OpenPaaS-Env/gem-dependency | bosh\_cli\_gem\_install\_package.tar | Off-line에서  BOSH CLI Dependency |
| OpenPaaS-Dev-Tools | bosh\_cli.zip | BOSH CLI |
| OpenPaaS-IaaS-Controller | bosh-187.tgz | BOSH 187 릴리즈 |
| OpenStack | OpenPaaS-Stemcells | bosh-stemcell-3016-openstack-kvm-ubuntu-trusty-go\_agent.tgz | OpenStack 3016버전  스템셀 이미지 |
| OpenPaaS-Deployment | openpaas-bosh-micro-openstack.yml | OpenStack MicroBOSH 배포 Manifest Sample 파일 |
| openpaas-bosh-openstack.yml | OpenStack BOSH 배포 Manifest Samle 파일 |

# Install BOSH CLI

BOSH는 Ruby언어로 구현되어 있으므로 Ruby개발 환경 설치 후 BOSH CLI를 설치해야 합니다. BOSH CLI설치 가능한 Ruby버전은 1.9.3, 2.0.x, 2.1.x, 2.2.x 이고 개발 환경에 이미 설치가 되어 있다면 4.1절 Ruby설치는 스킵하시면 됩니다.

## ***Ruby 설치***

BOSH CLI가 지원하는 Ruby버전 중 2.2.2버전을 설치를 대상으로 기술되었고, 상세 설치 절차는 다음과 같습니다.

1. 기준 디렉토리(“workspace”) 생성 후 3장에서 설치에 필요한 모든 구성항목을 다운로드하여 이 폴더로 옮겨놓습니다.

|  |
| --- |
| **# 기준 디렉토리 생성**  $ mkdir ~/workspace  $ cd workspace  **# Ubuntu unzip 패키지 설치**  $ sudo dpkg –i ./unzip/unzip\_6.0-18\_amd64.deb |

1. Ruby 설치에 필요한 Dependency들을 설치한다.

# On-Line의 경우

|  |
| --- |
| **# Ruby Dependency 설치**  $ sudo apt-get update  $ sudo apt-get install git-core curl zlib1g-dev build-essential libssl-dev libreadline-dev libyaml-dev libsqlite3-dev sqlite3 libxml2-dev libxslt1-dev libcurl4-openssl-dev python-software-properties libffi-dev |

# Off-Line의 경우

|  |
| --- |
| **# 다운로드 받은 bosh-ruby-dependency-package 압축해제**  $ tar –xvzf bosh-ruby-dependency-packages.tar.gz  $ cd ruby-dependency-packages  **# Ruby Dependency 설치**  $ sudo ./ruby-dependency-install.sh ./ |

1. 기준 디렉토리(“workspace”)로 이동하여 아래와 같이 명령어를 수행합니다.

|  |
| --- |
| **# Ruby Runtime 압축 해제**  $ tar –xvzf ruby-2.2.2.tar.gz  $ cd ruby-2.2.2  $ ./configure  $ make  **# Ruby Runtime 설치**  $ sudo make install |

상기의 절차로 설치 후 아래의 명령어로 ruby 설치 버전을 확인합니다.

|  |
| --- |
| $ ruby –v  **ruby 2.2.2p95 (2015-04-13 revision 50295) [x86\_64-linux]** |

## ***BOSH CLI설치***

BOSH CLI를 설치하는 절차는 다음과 같습니다.

1. 기준 디렉토리(workspace)로 이동하여 아래와 같이 명령어를 수행합니다.

# On-Line의 경우

|  |
| --- |
| **# bosh-cli.zip 압축해제**  $ cd ~/workspace  $ mkdir bosh; cd bosh;  $ unzip ~/workspace/bosh-cli.zip  **# BOSH CLI 설치**  $ sudo gem install bosh\_cli bosh\_cli\_plugin\_micro --no-ri --no-rdoc |

# Off-line의 경우

|  |
| --- |
| **# bosh-cli.zip 압축해제**  $ cd ~/workspace  $ mkdir bosh; cd bosh;  $ unzip ~/workspace/bosh-cli.zip  **# BOSH CLI Dependency 압축해제**  $ sudo tar -xvf ~/workspace/bosh\_cli\_gem\_install\_package.tar -C /usr/local/lib/ruby/gems/2.2.0  **# 압축 해제한 BOSH CLI 설치**  $ sudo gem install --force --local --no-ri --no-rdoc /usr/local/lib/ruby/gems/2.2.0/cache/\*.gem |

1. bosh 또는 bosh help명령어를 실행하시면 사용 가능한 CLI목록이 출력된다.

|  |
| --- |
| $ bosh or bosh help |

# Install MicroBOSH

본 장에서는 단일 VM에서 동작하는 MicroBOSH를 OpenStack에 설치하는 절차를 기술합니다.

## ***설치 절차***

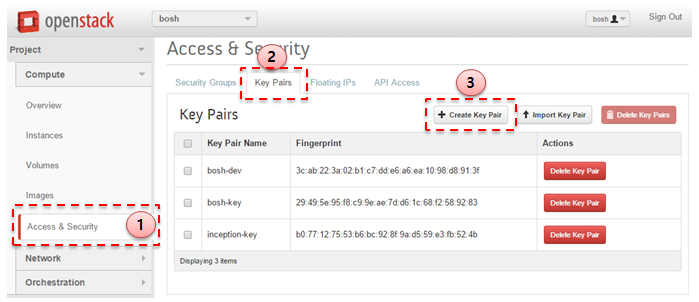
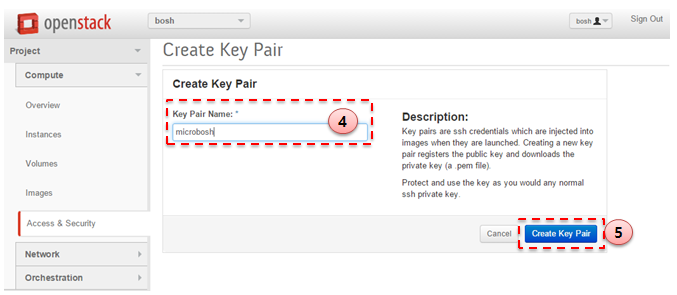
MicroBOSH를 배포하기 위해서는 아래와 같이 몇가지 준비 사항들이 사전에 준비되어 되어있어야 한다.

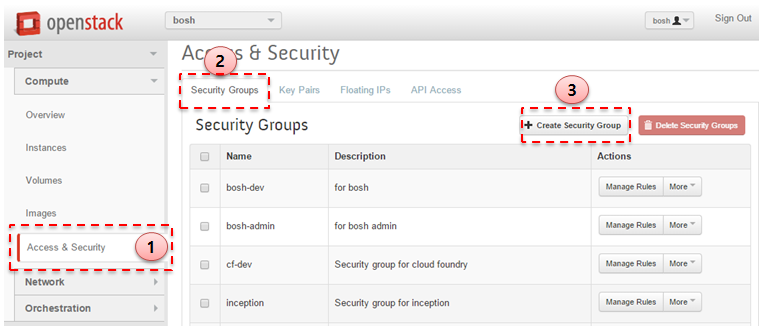
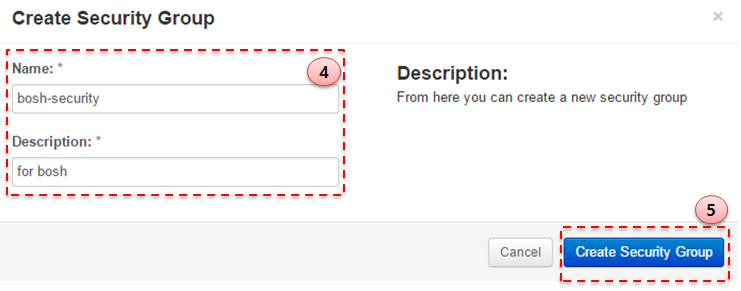
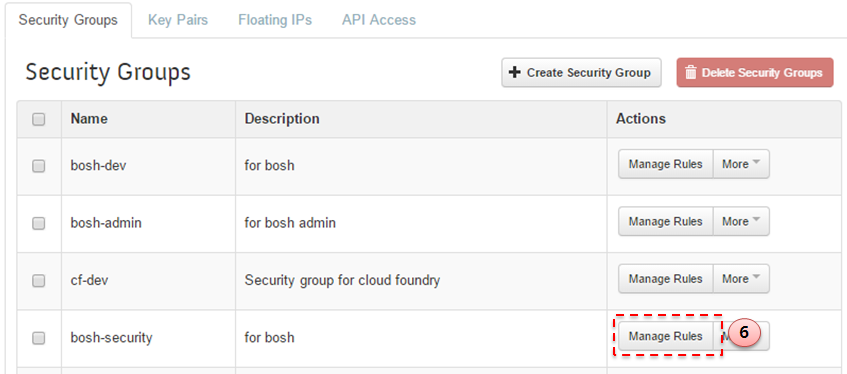
* IaaS 환경설정 : OpenStack 또는 Vsphere
* 배포 Manifest 파일 작성
* MicroBOSH 배포

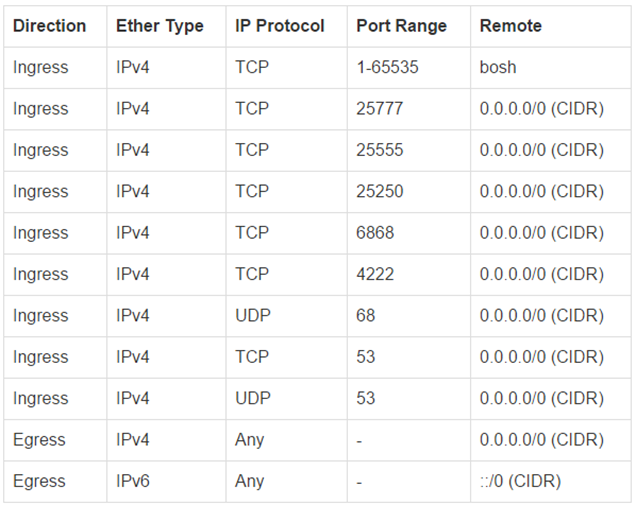
## ***IaaS 환경설정***

MicroBOSH를 배포하기 위한 OpenStack 사전 준비사항 및 필요한 사항을 아래의 가이드를 참조하여 설정한다.

* 사전 준비 및 확인 사항
  + 지원 가능한 OpenStack Release 버전 : Havana, IceHouse(권장), Juno
  + OpenStack 서비스 사용 가능 확인
    - Identity : 인증 및 OpenStack 서비스들의 Endpoint 조회
    - Compute : VM 생성과 IP 할당, 볼륨 생성 및 Attach
    - Image : 이미지 서비스를 이용한 Stemcell 저장
  + OpenStack 네트워크
    - 외부 또는 내부 네트워크의 Subnet
  + OpenStack 프로젝트 생성
* Key Pair 생성

1. 오픈스택 horizon의 왼쪽 메뉴에서 Compute 🡪 Access & Security 선택
2. “Key Pairs” Tab 선택
3. 화면 상단의 “Create Key Pair” 선택
4. Key pair 이름 입력 후 “Create Key Pair” 선택 및 저장

* Security Group 생성 및 구성
  1. 오픈스택 horizon의 왼쪽 메뉴에서 Compute 🡪 Access & Security 선택
  2. Security Groups Tab 선택
  3. “Create Security Group” 선택
  4. 생성할 Security Group의 Name, Description 입력 후 “Create Security Group” 선택
  5. Security Group 목록에서 생성된 Security Group의 “Manage Rules” 선택
  6. 다음 표와 같이 Bosh 관련Security Rule을 추가한다.



## ***배포 Manifest 작성***

다음은 MicroBOSH 배포하기 위한 Manifest파일을 작성하는 절차로 배포에 필요한 정보들을 YAML 파일 형식으로 정의한다.

1. 기준 디렉토리(“workspace”)로 이동 후 OpenStack MicroBOSH 배포 Manifest 샘플파일을 적당한 이름으로 네이밍하여 복사한다.

|  |
| --- |
| $ cd ~/workspace  $ cp openpaas-bosh-micro-openstack.yml bosh-micro.yml  $ vi bosh-micro.yml |

1. 아래와 속성 설명을 참고로하여 배포할 OpenStack 환경에 맞게 편집 후 저장한다.

---

name: microbosh **# 배포이름 설정**

network:

type: manual

vip: **FLOATING-IP # Floating IP 설정, Optional**

ip: **SUBNET-POOL-IP-ADDRESS # MicroBOSH IP 설정**

cloud\_properties:

net\_id: **NETWORK-UUID # MicroBOSH가 위치할 Subnet의 ID**

resources:

persistent\_disk: 40000 **# MicroBOSH VM을 위한 DISK 크기 설정**

cloud\_properties:

instance\_type: **m1.large** **# MicroBOSH VM의 사이즈로 오픈스택의 Flaver명**

cloud:

plugin: openstack **# IaaS Plugin명으로 openstack으로 설정**

properties:

openstack:

auth\_url: **IDENTITY-API-ENDPOINT # OpenStack Identity API End Point 설정**

tenant: **TENANT # OpenStack Tenant 이름 설정**

username: **USERNAME # OpenStack 계정명 설정**

api\_key: **PASSWORD # OpenStack 계정 비밀번호 설정**

default\_key\_name: **KEYPAIR-NAME # Key-Pair 이름 설정**

private\_key: **KEYPAIR-PATH # Key-Pair 경로 설정**

default\_security\_groups: [**SECURITY-GROUP-NAME**] **# Security Group 설정**

apply\_spec:

properties:

director: {max\_threads: 3} **# MicroBOSH Director의 Worker Thread 개수 설정**

hm: {resurrector\_enabled: true} **# Health Monitor VM 재시작 여부 설정**

ntp: **# NTP 서버 설정**

- 0.north-america.pool.ntp.org

- 1.north-america.pool.ntp.org

## ***MicroBOSH 배포***

다음의 절차를 이용해서 MicroBOSH를 배포한다.

1. 5.3절에서 작성한 MicroBOSH 배포 Manifest파일을 BOSH CLI에 설정한다.

$ **bosh micro deployment ~/workspace/bosh-micro.yml**

WARNING! Your target has been changed to `https://<MicroBOSH-IP>:25555'!

**Deployment set to '/home/ubuntu/workspace/bosh-micro.yml'**

1. 기준 디렉토리(“workspace”)안에 포함되어 있는 Stemcell파일을 인자로 배포를 실행한다.

**$ bosh micro deploy ~/workspace/bosh-stemcell-3016-openstack-kvm-ubuntu-trusty-go\_agent.tgz**

Deploying new MicroBOSH instance `microbosh-openstack/microbosh.yml' to `https://<MicroBOSH-IP>:25555' (type 'yes' to continue): yes

Verifying stemcell...

File exists and readable OK

Manifest not found in cache, verifying tarball...

Read tarball OK

Manifest exists OK

Stemcell image file OK

Writing manifest to cache...

Stemcell properties OK

Stemcell info

-------------

Name: bosh-openstack-kvm-ubuntu-trusty-go\_agent

Version: 3016

Deploy MicroBOSH

unpacking stemcell (00:00:02)

uploading stemcell (00:00:35)

creating VM from 04a1bdfe-4479-492e-8622-54380032a13a (00:01:21)

create disk (00:00:05)

mount disk (00:00:14)

stopping agent services (00:00:01)

applying MicroBOSH spec (00:00:16)

starting agent services (00:00:00)

waiting for the director (00:00:15)

Done 11/11 00:04:19

WARNING! Your target has been changed to `https://[MicroBOSH-IP]:25555'!

Deployment set to '~/workspace/micro-bosh.yml'

Deployed ` micro-bosh.yml' to `https://[MicroBOSH-IP]:25555', took 00:04:19 to complete

1. 배포가 성공하면 다음과 같이 Target 설정하면서 기본 계정인 admin/admin으로 로그인한다.

# MicroBOSH-IP는 Manifest파일에 지정한 IP

**$ bosh target https://[MicroBOSH-IP]:25555**

Target set to `microbosh'

Your username: **admin**

Enter password:

Logged in as `admin'

1. MicroBOSH의 상태정보를 확인한다.

**$ bosh status**

Config

/home/ubuntu/.bosh\_config **# BOSH CLI 설정파일 위치**

Director **# 현재 설정된 Director 정보**

Name microbosh **# Director 이름**

URL <https://[MicroBOSH-IP]:25555> **# Director URL**

Version 1.3001.0 (00000000) **# Director 버전**

User admin **# Director 로그인 계정**

**UUID dfb961e2-2b0a-4fce-b201-dff4ea3137bc # Director UUID**

CPI openstack **# Director의 CPI**

dns enabled (domain\_name: microbosh) **# Director의 DNS**

compiled\_package\_cache disabled **# Package 캐싱 여부 설정**

snapshots disabled **# Snapshot 기능 설정 여부**

Deployment

not set

# Install BOSH

본 장에서는 이전 장에서 설치한 MicroBOSH 이용하여 BOSH를 설치하는 절차를 기술합니다.

## ***설치 절차***

BOSH 를 설치하는 절차는 다음과 같다.

* BOSH Stemcell 업로드
* BOSH Release 업로드
* 배포 Manifest 작성
* BOSH 배포

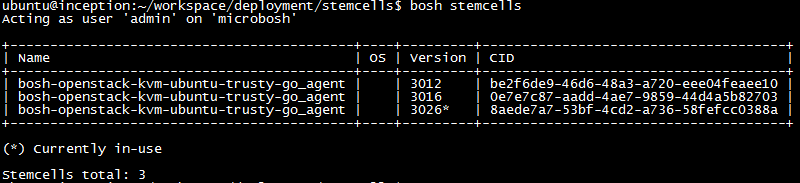
## ***BOSH Stemell 업로드***

Stemcell 업로드는 MicroBOSH가 BOSH VM인스턴스를 생성하기 위한 기반 이미지를 등록하는 절차로 기준 디렉토리(“workspace”)에 다운로드 받은 3016버전의 Stemcell을 MicroBOSH에 업로드 합니다.

. BOSH CLI 🡺 $ bosh upload stemcell [Stemcell 파일명]

|  |
| --- |
| $ **bosh upload stemcell ~/workspace/bosh-stemcell-3016-openstack-kvm-ubuntu-trusty-go\_agent.tgz**  Acting as user 'admin' on 'microbosh'  Verifying stemcell...  File exists and readable OK  Verifying tarball...  Read tarball OK  Manifest exists OK  Stemcell image file OK  Stemcell properties OK  Stemcell info  -------------  Name: bosh-openstack-kvm-ubuntu-trusty-go\_agent  Version: 3016  Checking if stemcell already exists...  No  Uploading stemcell...  bosh-stemcell: 100% |oooooooooooooooooooooooooooooooo| 533.8MB 26.1MB/s Time: 00:00:20  Director task 584  Started update stemcell  Started update stemcell > Extracting stemcell archive. Done (00:00:20)  Started update stemcell > Verifying stemcell manifest. Done (00:00:00)  Started update stemcell > Checking if this stemcell already exists. Done (00:00:00)  Started update stemcell > Uploading stemcell bosh-openstack-kvm-ubuntu-trusty-go\_agent/3016 to the cloud. Done (00:00:26)  Started update stemcell > Save stemcell bosh-openstack-kvm-ubuntu-trusty-go\_agent/3016 (0e7e7c87-aadd-4ae7-9859-44d4a5b82703). Done (00:00:00)  Done update stemcell (00:00:46)  Task 584 done  Started 2015-08-17 04:40:55 UTC  Finished 2015-08-17 04:41:41 UTC  Duration 00:00:46  Stemcell uploaded and created. |

Stemcell 업로드가 성공하면 나면 bosh stemcells 명령어로 아래와 같이 확인 가능하다.



## ***BOSH Release 업로드***

기준 디렉토리(“workspace”)에 다운로드 받은 187 버전의 release파일을 아래와 같이 CLI를 이용하여 MicroBOSH에 업로드한다.

* BOSH CLI 🡺 $ bosh upload release [release 파일명]

$ **bosh upload release ~/workspace/bosh-187.tgz**

Acting as user 'admin' on 'microbosh'

Verifying release...

File exists and readable OK

Extract tarball OK

Manifest exists OK

Release name/version OK

Read package 'health\_monitor' (1 of 14) OK

Package 'health\_monitor' checksum OK

Read package 'nginx' (2 of 14) OK

…

Uploading release

release-repac: 100% |ooooooooooooooooooooooooooooooooooooooo| 83.9MB 24.8MB/s Time: 00:00:03

Director task 589

Started extracting release > Extracting release. Done (00:00:01)

Started verifying manifest > Verifying manifest. Done (00:00:01)

Started resolving package dependencies > Resolving package dependencies. Done (00:00:00)

Started creating new packages

Started creating new packages > health\_monitor/cf4fe2fe2327a318d4ba1c80e68a0b188bf46d9f. Done (00:00:01)

Started creating new packages > nginx/1d356bbd17ed8c349fd1053093222d78559687ec. Done (00:00:00)

Started creating new packages > registry/ad2188b1497d2e07030b581d4904286de17ebd33. Done (00:00:00)

Started creating new packages > ruby/c2b6c610123a00a406b66ea4ccd123e24bfcd404. Done (00:00:01)

Started creating new packages > blobstore/69eead438a0ed8187b5f5194720f5452a7b266ee. Done (00:00:00)

…

Started release has been created > bosh/187. Done (00:00:00)

Task 589 done

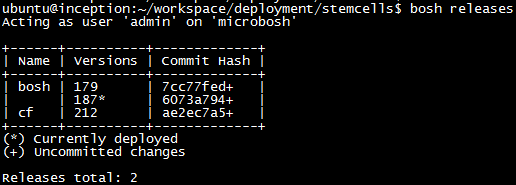
Started 2015-08-18 01:12:55 UTC

Finished 2015-08-18 01:13:00 UTC

Duration 00:00:05

Release uploaded

BOSH Release 업로드가 성공하면 bosh releases 명령어로 등록된 Release 정보를 아래와 같이 확인 가능하다.



## ***배포 Manifest 작성***

MicroBOSH 배포 Manifest파일 작성과 유사하게 기준 디렉토리(“workspace”) 이동 후 아래의 절차에 따라 배포 Manifest를 작성한다.

1. 기준 디렉토리(workspace)로 이동 후 BOSH 배포 Manifest 샘플파일을 적당한 이름으로 네이밍하여 복사하고, 오픈한다.

|  |
| --- |
| $ cd ~/workspace  $ cp openpaas-bosh-openstack.yml bosh-openstack.yml  $ vi bosh-openstack.yml |

1. 아래와 속성 설명을 참고로하여 배포할 OpenStack 환경에 맞게 편집 후 저장한다.

|  |
| --- |
| ---  name: bosh-openstack  director\_uuid: dfb961e2-2b0a-4fce-b201-dff4ea3137bc **# MicroBOSH Director의 UUID**  release:  name: bosh **# 설치할 BOSH Release 이름**  version: 187 **# 설치할 BOSH Release 버전**  compilation: **# 배포할 패키지들을 컴파일하기 위한 설정들**  workers: 3 **# 패키지 컴파일하기 위해 생성할 VM의 개수**  network: default **# VM생성 시 사용할 networks 섹션의 이름**  reuse\_compilation\_vms: true **# 패키지 컴파일 VM 재사용 여부 설정**  cloud\_properties:  instance\_type: m1.medium **# 패키지 컴파일을 위해 생성할 VM의 Flaver 설정**  update:  canaries: 1 # **Canary 인스턴스 개수**  canary\_watch\_time: 3000-120000 **# Canary 인스턴스가 완성될때까지 기다리는 시간(단위:ms)**  update\_watch\_time: 3000-120000 **# Canary 인스턴스가 업데이트될때까지 기다리는 시간 (단위: ms)**  max\_in\_flight: 4 **# 동일 Job에 대한 VM 인스턴스를 병렬로 생성할 최대 개수 설정**  networks:  - name: default **# 네트워크 이름 설정**  type: manual **# 네트워크 타입 설정 (타입종류 : manual, dynamic, vip)**  subnets:  - range: 172.16.130.0/24 **# Subnet IP Range(CIDR) 설정**  gateway: 172.16.130.1 **# Subne Gateway IP 설정**  dns:  - 8.8.8.8 **# DNS IP 설정**  static: [172.16.130.101 – 172.16.130.107] **# Static IP Range, Job VM 인스턴스가 사용할 IP**  cloud\_properties: **# OpenStack의 설정 정보**  net\_id: 2d478baa-69a6-4840-8e8e-1732a3f5a1c3 **# BOSH가 위치할 Subnet의 ID**  resource\_pools:  - name: medium **# Unique한 Resource Pool 이름 설정**  network: default **# networks 섹션의 이름**  size: 7 **# 해당 Resource Pool에 생성할 VM의 개수 설정**  stemcell: **# 해당 Resource Pool에서 사용할 Stemcell 설정**  name: bosh-openstack-kvm-ubuntu-trusty-go\_agent **# Stemcell 이름**  version: 3012 **# Stemcell 버전**  cloud\_properties:  instance\_type: m1.medium # **OpenStack의 Flaver 설정**  jobs:  - name: nats **# Unique한 Job 이름 지정**  template: nats **# 해당 VM에 설치할 Job Template 설정**  instances: 1 **# VM 인스턴스의 개수**  resource\_pool: medium **# 사용할 resource\_pools섹션의 이름 설정**  networks:  - name: default **# 사용할 networks 섹션의 이름**  default: [dns, gateway] **# Default DNS, Gateway 설정**  static\_ips: [172.16.130.101] **# Change, VM 인스턴스의 IP 설정**  - name: redis  template: redis  instances: 1  resource\_pool: medium  networks:  - name: default  default: [dns, gateway]  static\_ips: [172.16.130.102] **# Change**  - name: postgres  template: postgres  instances: 1  resource\_pool: medium  persistent\_disk: 16384 **# Change, postgres DB가 사용할 Persistent Disk 설정**  networks:  - name: default  default: [dns, gateway]  static\_ips: [172.16.130.103] **# Change**  - name: blobstore  template: blobstore  instances: 1  resource\_pool: medium  persistent\_disk: 51200 **# Change, blobstore가 사용할 Persistent Disk 설정**  networks:  - name: default  default: [dns, gateway]  static\_ips: [172.16.130.104] **# Change**  - name: director  templates:  - name: director  - name: powerdns  instances: 1  resource\_pool: medium  persistent\_disk: 16384 **# Change, director가 사용할 Persistent Disk 크기 설정**  networks:  - name: default  default: [dns, gateway]  static\_ips: [172.16.130.105] **# Change**  - name: registry  template: registry  instances: 1  resource\_pool: medium  networks:  - name: default  default: [dns, gateway]  static\_ips: [172.16.130.106] **# Change**  - name: health\_monitor  template: health\_monitor  instances: 1  resource\_pool: medium  networks:  - name: default  default: [dns, gateway]  static\_ips: [172.16.130.107] **# Change**  properties:  nats:  address: [172.16.130.101] **# Change, nats IP 주소 설정**  port: 4222 **# NATS 포트 설정**  user: nats **# NATS 계정 이름**  password: nats **# NATS 계정 비밀번호**  redis:  address: [172.16.130.102] **# Change, redis IP 주소 설정**  password: redis **# redis 비밀번호**  postgres: &bosh\_db  host: [172.16.130.103] **# Change, postgresql IP 주소 설정**  user: postgres **# postgres DB 계정 이름**  password: postgres **# postgres DB 계정 비밀번호**  database: bosh **# postgres DB 이름**  blobstore:  address: [172.16.130.104] **# Change, blobstore IP 설정**  agent: **# agent가 blobstore에 접근하기 위한 계정 설정**  user: agent  password: agent  director: **# director가 blobstore에 접근하기 위한 계정 설정**  user: director  password: director  director:  name: bosh  address: [172.16.130.105] **# Change, director IP 설정**  db: \*bosh\_db **# 상기 postgres Property에서 설정한 속성 사용**  registry:  address: [172.16.130.106] **# Change, registry IP 설정**  db: \*bosh\_db **# 상기 postgres Property에서 설정한 속성 사용**  http: **# registry에 접속할 계정 설정**  user: registry  password: registry  dns:  address: [172.16.130.105] **# Change, dns IP 설정**  db: \*bosh\_db **# 상기 postgres Property에서 설정한 속성 사용**  hm:  director\_account: **# director에 접근할 계정 설정**  user: admin  password: admin  resurrector\_enabled: true **# VM Resurrector Pluging 설정 여부**  ntp: **# NTP 서버 설정**  - 0.north-america.pool.ntp.org  - 1.north-america.pool.ntp.org  openstack:  auth\_url: **IDENTITY-API-ENDPOINT # OpenStack Identity API End Point 설정**  tenant: **TENANT # OpenStack Tenant 이름 설정**  username: **USERNAME # OpenStack 계정 이름 설정**  api\_key: **PASSWORD # OpenStack 계정 비밀번호 설정**  default\_key\_name: **KEYPAIR-NAME # Key-Pair 이름 설정**  private\_key: **KEYPAIR-PATH # Key-Pair 경로 설정**  default\_security\_groups: [**SECURITY\_GROUP\_NAME**] **# Security Group 설정** |

## ***BOSH 배포***

다음의 절차를 이용해서 BOSH를 배포한다.

1. 터미널 상에 작성한 BOSH 배포 Manifest파일을 CLI에 설정한다.

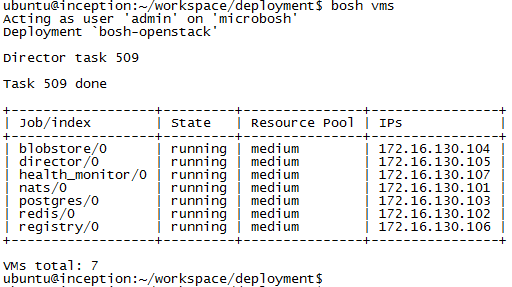
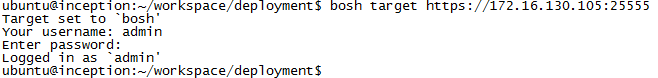
# 작성한 BOSH 배포 Manifest파일을 인자로 지정하여 Deployment 설정

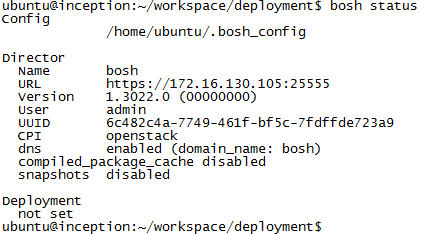
$ **bosh deployment ~/workspace/bosh-openstack.yml**

**Deployment set to '/home/ubuntu/workspace/bosh-openstack.yml'**

1. 아래와 같이 bosh deploy명령어를 이용하여 배포를 실행한다.

|  |
| --- |
| **$ bosh deploy**  Acting as user 'admin' on deployment 'bosh-openstack' on 'microbosh'  Getting deployment properties from director...  Please review all changes carefully  Deploying  ---------  Are you sure you want to deploy? (type 'yes' to continue): yes  Director task 508  Started unknown  Started unknown > Binding deployment. Done (00:00:00)  Started preparing deployment  Started preparing deployment > Binding releases. Done (00:00:00)  Started preparing deployment > Binding existing deployment. Done (00:00:00)  Started preparing deployment > Binding resource pools. Done (00:00:00)  Started preparing deployment > Binding stemcells. Done (00:00:00)  Started preparing deployment > Binding templates. Done (00:00:00)  Started preparing deployment > Binding properties. Done (00:00:00)  Started preparing deployment > Binding unallocated VMs. Done (00:00:00)  Started preparing deployment > Binding instance networks. Done (00:00:00)  Started preparing package compilation > Finding packages to compile. Done (00:00:00)  Started compiling packages  Started compiling packages > powerdns/256336d00b1689138490c385c03ad3a8f54b4a9e  Started compiling packages > genisoimage/008d332ba1471bccf9d9aeb64c258fdd4bf76201  Started compiling packages > mysql/e5309aed88f5cc662bc77988a31874461f7c4fb8  Done compiling packages > powerdns/256336d00b1689138490c385c03ad3a8f54b4a9e (00:02:40)  Started compiling packages > libpq/f181aa97dd63567e04d897762f0440fb2bef1517  Done compiling packages > mysql/e5309aed88f5cc662bc77988a31874461f7c4fb8 (00:02:51)  Started compiling packages > nginx/0ccc40df032599285cb16a4f2dfdf324a4a61a26  Done compiling packages > libpq/f181aa97dd63567e04d897762f0440fb2bef1517 (00:00:24)  Started compiling packages > postgres/aa7f5b110e8b368eeb8f5dd032e1cab66d8614ce  Done compiling packages > genisoimage/008d332ba1471bccf9d9aeb64c258fdd4bf76201 (00:03:11)  Started compiling packages > redis/37eae530889cb9ef4e84f9c3d0827bab5ae5cb66  Done compiling packages > postgres/aa7f5b110e8b368eeb8f5dd032e1cab66d8614ce (00:00:08)  Started compiling packages > ruby/c28e01e561dd7e1bd4ce44f134970dc9f5d7e8fc  Done compiling packages > redis/37eae530889cb9ef4e84f9c3d0827bab5ae5cb66 (00:00:29)  Done compiling packages > nginx/0ccc40df032599285cb16a4f2dfdf324a4a61a26 (00:00:54)  Done compiling packages > ruby/c28e01e561dd7e1bd4ce44f134970dc9f5d7e8fc (00:02:39)  Started compiling packages > health\_monitor/10a74646b7b8c093ab1816e8689d513b3c3a5ef6  Started compiling packages > registry/e95305aaef1f8377a7a25e4eadd6ff1612de483a  Started compiling packages > director/2be1abbf39d80ff40aa86e0253c91daa857c109b  Done compiling packages > health\_monitor/10a74646b7b8c093ab1816e8689d513b3c3a5ef6 (00:02:43)  Started compiling packages > nats/6a31c7bb0d5ffa2a9f43c7fd7193193438e20e92  Done compiling packages > registry/e95305aaef1f8377a7a25e4eadd6ff1612de483a (00:02:50)  Done compiling packages > nats/6a31c7bb0d5ffa2a9f43c7fd7193193438e20e92 (00:00:14)  Done compiling packages > director/2be1abbf39d80ff40aa86e0253c91daa857c109b (00:04:24)  Done compiling packages (00:10:15)  Started preparing dns > Binding DNS. Done (00:00:00)  Started creating bound missing vms  Started creating bound missing vms > medium/0  Started creating bound missing vms > medium/1  Started creating bound missing vms > medium/2. Done (00:02:30)  Started creating bound missing vms > medium/3  Done creating bound missing vms > medium/1 (00:02:34)  Started creating bound missing vms > medium/4  Done creating bound missing vms > medium/0 (00:02:34)  Started creating bound missing vms > medium/5. Done (00:02:14)  Started creating bound missing vms > medium/6  Done creating bound missing vms > medium/4 (00:02:14)  Done creating bound missing vms > medium/3 (00:02:20)  Done creating bound missing vms > medium/6 (00:02:09)  Done creating bound missing vms (00:06:57)  Started binding instance vms  Started binding instance vms > nats/0  Started binding instance vms > redis/0  Started binding instance vms > postgres/0. Done (00:00:00)  Started binding instance vms > blobstore/0  Done binding instance vms > nats/0 (00:00:00)  Started binding instance vms > director/0  Done binding instance vms > blobstore/0 (00:00:00)  Started binding instance vms > registry/0  Done binding instance vms > director/0 (00:00:00)  Started binding instance vms > health\_monitor/0  Done binding instance vms > registry/0 (00:00:00)  Done binding instance vms > health\_monitor/0 (00:00:00)  Done binding instance vms > redis/0 (00:00:01)  Done binding instance vms (00:00:01)  Started preparing configuration > Binding configuration. Done (00:00:01)  Started updating job nats > nats/0 (canary). Done (00:00:12)  Started updating job redis > redis/0 (canary). Done (00:00:10)  Started updating job postgres > postgres/0 (canary). Done (00:00:55)  Started updating job blobstore > blobstore/0 (canary). Done (00:00:27)  Started updating job director > director/0 (canary). Done (00:03:19)  Started updating job registry > registry/0 (canary). Done (00:00:25)  Started updating job health\_monitor > health\_monitor/0 (canary). Done (00:00:12)  Task 508 done  Started 2015-07-28 09:12:26 UTC  Finished 2015-07-28 09:35:29 UTC  Duration 00:23:03  Deployed `bosh-openstack' to `microbosh' |

1. 배포가 성공하면 배포된 VM정보를 확인한다.
2. 배포한 BOSH의 Director로 Target 설정하고 기본 계정인 admin/admin으로 로그인한다.
3. BOSH의 상태정보를 확인한다.



**END**

1. 변경 내용: 변경이 발생되는 위치와 변경 내용을 자세히 기록(장/절과 변경 내용을 기술한다.) [↑](#footnote-ref-1)