

전자정부표준프레임워크 기반의 Open PaaS 개발

BOSH 설치 가이드(설치패키지)

|  |  |  |
| --- | --- | --- |
| 작성자 | 문철호 (인) | 2015.8.5 |
| 검토자 | 김태형 (인) | 2015.8.7 |
| 승인자 | 김기현 (인) | 2015.8.10 |

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

개 정 이 력

| 버전 | 작성일 | 변경내용[[1]](#footnote-1) | 작성자 | 승인자 |
| --- | --- | --- | --- | --- |
| 0.1 | 2015.8.5 | 최초작성 | 문철호 | 김기현 |
| 1.0 | 2016.1.11 | Bosh-init 설치 내용 추가 | 김태형 | 김기현 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

목차

[1. 문서 개요 4](#_Toc440542029)

[*1.1.* *목적* 4](#_Toc440542030)

[*1.2.* *범위* 4](#_Toc440542031)

[*1.3.* *참고자료* 4](#_Toc440542032)

[2. Basic BOSH Workflow 5](#_Toc440542033)

[3. BOSH 설치 패키지 구성 요소 6](#_Toc440542034)

[4. Install BOSH CLI 7](#_Toc440542035)

[*4.1.* *Ruby 설치* 7](#_Toc440542036)

[*4.2.* *BOSH INIT 설치* 10](#_Toc440542037)

[*4.3.* *BOSH CLI 설치* 11](#_Toc440542038)

[5. Install MicroBOSH 14](#_Toc440542039)

[*5.1.* *설치 절차* 14](#_Toc440542040)

[*5.2.* *AWS 환경설정* 14](#_Toc440542041)

[*5.3.* *배포 Manifest 작성* 23](#_Toc440542042)

[*5.4.* *MicroBOSH 배포* 27](#_Toc440542043)

[6. Install BOSH 29](#_Toc440542044)

[*6.1.* *설치 절차* 29](#_Toc440542045)

[*6.2.* *BOSH Stemell 업로드* 29](#_Toc440542046)

[*6.3.* *BOSH Release 업로드* 29](#_Toc440542047)

[*6.4.* *배포 Manifest 작성* 29](#_Toc440542048)

[*6.5.* *BOSH 배포* 36](#_Toc440542049)

[*6.6.* *BOSH 배포 Troubleshooting* 36](#_Toc440542050)

Executive Summary

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

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 환경에 MicroBOSH와 BOSH를 설치하여 기능을 테스트할 수 있는 환경을 구축하는 데 목적이 있다.

## ***범위***

본 가이드에서는 오프라인 Linux 환경(Ubuntu 14.04)을 기준으로 설치환경 구성 및 BOSH CLI 설치하고, 이를 이용하여 MicroBOSH 및 BOSH를 설치하는 것을 기준으로 작성하였다.

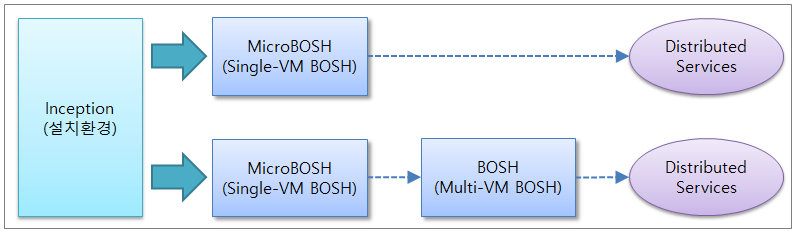
## ***참고자료***

본 문서는 Cloud Foundry의 BOSH Document를 참고로 작성하였다.

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

# Basic BOSH Workflow

본 문서의 목적에서도 언급했듯이 BOSH는 클라우드 환경에 서비스를 배포 관리하는 소프트웨어로 BOSH자체도 클라우드에 배포되어야 하는 서비스로 설치 환경이 구성되어 있는 서버로부터 MicroBOSH만을 설치하여 운영하는 방법과 MiroBOSH 로부터 배포된 BOSH를 이용하는 방법 2가지 유형이 있으니 자신의 로컬 설치 환경에 따라 선택할 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 구성 요소

다음의 설치 패키지는 MicroBOSH와 BOSH를 설치하기 위한 패키지로 아래의 구성 항목을 로컬 설치 환경의 기준 디렉토리(“workspace”)로 복사한다.

|  |  |  |  |
| --- | --- | --- | --- |
| **구성 항목** | | | **설명** |
| **구분** | **디렉토리** | **파일명** |
| 공통 | OpenPaaS-Env | bosh-init-0.0.81-linux-amd64 | BOSH init 실행 파일 |
| bosh\_gem\_package.tar | BOSH CLI 설치 패키지 |
| ruby-2.1.6.tar.gz | Ruby 2.1.6 버전 |
| OpenPaaS-Env/bosh\_env\_packages | 디렉토리 전체 | BOSH 실행 환경 설치 패키지 |
| OpenPaaS-IaaS-Controller | bosh-233.tgz | BOSH v233 릴리즈 |
| AWS | OpenPaaS-IaaS-Controller | bosh-aws-cpi-release-36.tgz | BOSH Plugin CPI |
| OpenPaaS-Stemcells | light-bosh-stemcell-3147-aws-xen-hvm-ubuntu-trusty-go\_agent.tgz | AWS v3147 스템셀 이미지  Bosh-init 설치용 |
| bosh-stemcell-3147-aws-xen-ubuntu-trusty-go\_agent.tgz | AWS v3147 스템셀 이미지  Bosh 설치용 |
| OpenPaaS-Deployment | openpaas-bosh-init-aws-1.0.yml | AWS MicroBOSH 배포 Manifest Sample 파일 |
| openpaas-bosh-aws-1.0.yml | AWS BOSH 배포 Manifest Samle 파일 |

# Install BOSH CLI

BOSH는 Ruby언어로 구현되어 있으므로 Ruby개발 환경 설치 후 BOSH CLI를 설치해야 한다. BOSH CLI설치 가능한 Ruby버전은 1.9.3 이상이며, 개발 환경에 이미 설치가 되어 있다면 4.1절 Ruby설치는 건너뛰면 된다.

## ***Ruby 설치***

BOSH CLI가 지원하는 Ruby버전 중 2.1.6버전을 설치 대상으로 기술하였고, 상세 설치 절차는 다음과 같다.

1. Bosh Dependency를 설치한다.

* **온라인인 경우(권장)**

|  |  |  |  |
| --- | --- | --- | --- |
| * **Ubuntu의 경우**  |  | | --- | | **$ sudo apt-get update**  **$ sudo apt-get install -y build-essential zlibc zlib1g-dev ruby ruby-dev openssl libxslt-dev libxml2-dev libssl-dev libreadline6 libreadline6-dev libyaml-dev libsqlite3-dev sqlite3 libxslt1-dev libpq-dev libmysqlclient-dev** |  * **CentOS의 경우**  |  | | --- | | **$ sudo yum install gcc ruby ruby-devel mysql-devel postgresql-devel postgresql-libs sqlite-devel libxslt-devel libxml2-devel yajl-ruby** |  * **OSX의 경우**  |  | | --- | | **$ xcode-select --install**  **xcode-select: note: install requested for command line developer tools** | |

* **오프라인인 경우**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| # 설치환경이 ubuntu (trusty 버전)인 경우 다음의 스크립트를 실행한다.  **$ cd ~/workspace/OpenPaaS-Env/bosh\_env\_packages**  **$ chmod +x ./bosh-packages-install.sh**  **$ ./bosh-packages-install.sh**   * **BOSH 설치 패키지의 구성은 다음과 같다.**  |  |  |  | | --- | --- | --- | | No | 패키지명 | 패키지 파일명 (Ubuntu trusty amd64) | | 1 | libcomerr2 | libcomerr2\_1.42.9-3ubuntu1.3\_amd64.deb | | 2 | libssl1.0.0 | libssl1.0.0\_1.0.1f-1ubuntu2.16\_amd64.deb | | 3 | libkrb5support0 | libkrb5support0\_1.12+dfsg-2ubuntu5.2\_amd64.deb | | 4 | libk5crypto3 | libk5crypto3\_1.12+dfsg-2ubuntu5.2\_amd64.deb | | 5 | libkrb5-3 | libkrb5-3\_1.12+dfsg-2ubuntu5.2\_amd64.deb | | 6 | libgssapi-krb5-2 | libgssapi-krb5-2\_1.12+dfsg-2ubuntu5.2\_amd64.deb | | 7 | libxml2 | libxml2\_2.9.1+dfsg1-3ubuntu4.6\_amd64.deb | | 8 | libgssrpc4 | libgssrpc4\_1.12+dfsg-2ubuntu5.2\_amd64.deb | | 9 | libkadm5clnt-mit9 | libkadm5clnt-mit9\_1.12+dfsg-2ubuntu5.2\_amd64.deb | | 10 | libkdb5-7 | libkdb5-7\_1.12+dfsg-2ubuntu5.2\_amd64.deb | | 11 | libkadm5srv-mit9 | libkadm5srv-mit9\_1.12+dfsg-2ubuntu5.2\_amd64.deb | | 12 | mysql-common | mysql-common\_5.5.46-0ubuntu0.14.04.2\_all.deb | | 13 | libmysqlclient18 | libmysqlclient18\_5.5.46-0ubuntu0.14.04.2\_amd64.deb | | 14 | libxslt1.1 | libxslt1.1\_1.1.28-2build1\_amd64.deb | | 15 | libyaml-0 | libyaml-0-2\_0.1.4-3ubuntu3.1\_amd64.deb | | 16 | openssl\_1.0.1f-1ubuntu2.16 | openssl\_1.0.1f-1ubuntu2.16\_amd64.deb | | 17 | libstdc++-4.8-dev | libstdc++-4.8-dev\_4.8.4-2ubuntu1~14.04\_amd64.deb | | 18 | g++-4.8 | g++-4.8\_4.8.4-2ubuntu1~14.04\_amd64.deb | | 19 | g++ | g++\_4%3a4.8.2-1ubuntu6\_amd64.deb | | 20 | libdpkg-perl | libdpkg-perl\_1.17.5ubuntu5.5\_all.deb | | 21 | dpkg-dev | dpkg-dev\_1.17.5ubuntu5.5\_all.deb | | 22 | build-essential | build-essential\_11.6ubuntu6\_amd64.deb | | 23 | libfakeroot | libfakeroot\_1.20-3ubuntu2\_amd64.deb | | 24 | fakeroot | fakeroot\_1.20-3ubuntu2\_amd64.deb | | 25 | comerr-dev | comerr-dev\_2.1-1.42.9-3ubuntu1.3\_amd64.deb | | 26 | krb5-multidev | krb5-multidev\_1.12+dfsg-2ubuntu5.2\_amd64.deb | | 27 | libalgorithm-diff-perl | libalgorithm-diff-perl\_1.19.02-3\_all.deb | | 28 | libalgorithm-diff-xs-perl | libalgorithm-diff-xs-perl\_0.04-2build4\_amd64.deb | | 29 | libalgorithm-merge-perl | libalgorithm-merge-perl\_0.08-2\_all.deb | | 30 | libtinfo-dev | libtinfo-dev\_5.9+20140118-1ubuntu1\_amd64.deb | | 31 | libreadline6-dev | libreadline6-dev\_6.3-4ubuntu2\_amd64.deb | | 32 | libfile-fcntllock-perl | libfile-fcntllock-perl\_0.14-2build1\_amd64.deb | | 33 | zlib1g-dev | zlib1g-dev\_1%3a1.2.8.dfsg-1ubuntu1\_amd64.deb | | 34 | libmysqlclient-dev | libmysqlclient-dev\_5.5.46-0ubuntu0.14.04.2\_amd64.deb | | 35 | libpq5 | libpq5\_9.3.10-0ubuntu0.14.04\_amd64.deb | | 36 | libssl-dev | libssl-dev\_1.0.1f-1ubuntu2.16\_amd64.deb | | 37 | libpq-dev | libpq-dev\_9.3.10-0ubuntu0.14.04\_amd64.deb | | 38 | libsqlite3-dev | libsqlite3-dev\_3.8.2-1ubuntu2.1\_amd64.deb | | 39 | libssl-doc | libssl-doc\_1.0.1f-1ubuntu2.16\_all.deb | | 40 | libxml2-dev | libxml2-dev\_2.9.1+dfsg1-3ubuntu4.6\_amd64.deb | | 41 | libxslt1-dev | libxslt1-dev\_1.1.28-2build1\_amd64.deb | | 42 | sqlite3 | sqlite3\_3.8.2-1ubuntu2.1\_amd64.deb | | 43 | Zlibc | zlibc\_0.9k-4.1\_amd64.deb | | 44 | libyaml-dev | libyaml-dev\_0.1.4-3ubuntu3.1\_amd64.deb | | 45 | libruby1.9.1 | libruby1.9.1\_1.9.3.484-2ubuntu1.2\_amd64.deb | | 46 | ruby1.9.1-dev | ruby1.9.1-dev\_1.9.3.484-2ubuntu1.2\_amd64.deb | | 47 | ruby-dev | ruby-dev\_1%3a1.9.3.4\_all.deb | | 48 | ruby | ruby\_1%3a1.9.3.4\_all.deb | | 49 | ruby1.9.1 | ruby1.9.1\_1.9.3.484-2ubuntu1.2\_amd64.deb |  * CentOS, RHEL 또는 OSX의 경우는 표에 해당하는 패키지를 설치한다. |

1. Ruby 설치

|  |
| --- |
| **$ tar -xvzf ~/workspace/OpenPaaS-Env/ruby-2.1.6.tar.gz**  **$ cd ruby-2.1.6/**  **$ ./configure**  **$ make**  **$ sudo make install** |

* Ruby를 컴파일하여 설치할 경우 기본으로 /usr/local/lib에 설치한다.
* 온라인의 경우, rvm 또는 rbenv 등을 이용하여 ruby를 설치할 수 있다. 단, ruby 1.9.3버전 이상을 설치하도록 한다.

1. 설치 확인

|  |
| --- |
| **# Ruby 버전 확인 (터미널창을 새로 열어서 확인한다.)**  **$ ruby -v**  $ ruby -v  ruby 2.1.6p336 (2015-04-13 revision 50298) [x86\_64-linux]  **# Ruby 환경설정 확인**  **$ gem env**  $ gem env  RubyGems Environment:  - RUBYGEMS VERSION: 2.2.3  - RUBY VERSION: 2.1.6 (2015-04-13 patchlevel 336) [x86\_64-linux]  - **INSTALLATION DIRECTORY: /usr/local/lib/ruby/gems/2.1.0 <- gem 설치 디렉토리**  - RUBY EXECUTABLE: /usr/local/bin/ruby  - EXECUTABLE DIRECTORY: /usr/local/bin  - SPEC CACHE DIRECTORY: /home/ubuntu/.gem/specs  - RUBYGEMS PLATFORMS:  - ruby  - x86\_64-linux  - GEM PATHS:  - /usr/local/lib/ruby/gems/2.1.0  - /home/ubuntu/.gem/ruby/2.1.0  - GEM CONFIGURATION:  -:update\_sources => true  -:verbose => true  -:backtrace => false  -:bulk\_threshold => 1000  - REMOTE SOURCES:  - https://rubygems.org/  - SHELL PATH:  - /usr/local/sbin  - /usr/local/bin  - /usr/sbin  - /usr/bin  - /sbin  - /bin  - /usr/games  - /usr/local/games |

## ***BOSH INIT 설치***

Bosh-init은 MicroBOSH를 설치에 사용하는 도구이다. Bosh-init을 설치하는 절차는 다음과 같다.

1. Bosh\_init 설치

* 온라인의 경우

|  |
| --- |
| # 다음의 웹사이트에서 설치환경에 맞는 bosh-init 바이너리 파일을 다운로드한다.  **https://bosh.io/docs/install-bosh-init.html** |

|  |
| --- |
| # 다운로드 받은 bosh-init에 실행 권한을 부여하고 설치한다. (~/Download 디렉토리에 파일을 다운로드 받았다고 가정)  $ chmod +x ~/Downloads/bosh-init-\*  $ sudo mv ~/Downloads/bosh-init-\* /usr/local/bin/bosh-init |

* 오프라인의 경우

|  |
| --- |
| # OpenPaaS 패키지의 bosh-init을 설치  $ chmod +x ~/workspace/OpenPaaS-Env/bosh-init-0.0.81-linux-amd64  $ sudo mv ~/workspace/OpenPaaS-Env/bosh-init-\* /usr/local/bin/bosh-init |

1. Bosh-init 설치 확인

|  |
| --- |
| $ bosh-init -v |

## ***BOSH CLI 설치***

BOSH CLI를 설치하는 절차는 다음과 같다.

1. Bosh\_cli 설치

* 온라인의 경우

|  |
| --- |
| #bosh\_cli 패키지 설치  **$ sudo gem install bosh\_cli**  *※ gem 설치 디렉토리 소유자가 로그인한 사용자와 동일한 경우는 ‘sudo’를 사용하지 않는다.* |

* 오프라인의 경우

|  |
| --- |
| **#bosh\_cli 의존 패키지 압축해제**  **$ sudo tar -xvf ~/workspace/OpenPaaS-Env/bosh\_gem\_package.tar -C *<gem 설치 디렉토리>/cache***  *※ gem 설치 디렉토리는 ‘gem env’를 실행하여 확인한다. (예: /usr/local/lib/ruby/gems/2.1.0)*  **#압축해제한 gem 설치**  **$ sudo gem install --force --local --no-ri --no-rdoc *<gem 설치 디렉토리>*/cache/\*.gem**  *※ gem 설치 디렉토리 소유자가 로그인한 사용자와 동일한 경우는 ‘sudo’를 사용하지 않는다.*  **#참고: 기존에 설치한 gem 삭제**  **$ sudo gem uninstall -aIx** |

1. Bosh 설치 확인

|  |
| --- |
| **$ bosh**  $ /OpenPaaS-Env/virtualbox-5.0$ bosh  BOSH CLI helps you manage your BOSH deployments and releases.  Usage: bosh [<options>] <command> [<args>]  -c, --config FILE Override configuration file. Also can be overridden by BOSH\_CONFIG environment variable. Defaults to $HOME/.bosh\_config. Override precedence is command-line option, then environment variable, then home directory.  --parallel MAX Sets the max number of parallel downloads  --[no-]color Toggle colorized output  -v, --verbose Show additional output  -q, --quiet Suppress all output  -n, --non-interactive Don't ask for user input  -N, --no-track Return Task ID and don't track  -P, --poll INTERVAL Director task polling interval  -t, --target URL Override target  -u, --user USER Override username  -p, --password PASSWORD Override password  -d, --deployment FILE Override deployment  -h, --help here you go  add blob <local\_path> [<blob\_dir>]  Add a local file as BOSH blob  alias <name> <command>  Create an alias <name> for command <command>  <<이하 생략>> |

# Install MicroBOSH

본 장에서는 단일 VM에서 동작하는 MicroBOSH를 상용 클라우드 플랫폼인 AWS에 설치하는 절차를 기술한다.

## ***설치 절차***

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

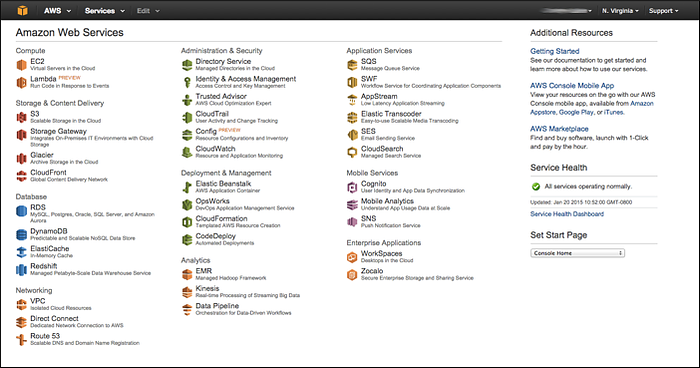
* IaaS 환경설정: AWS
* 배포 Manifest 파일 작성
* MicroBOSH 배포

## ***AWS 환경설정***

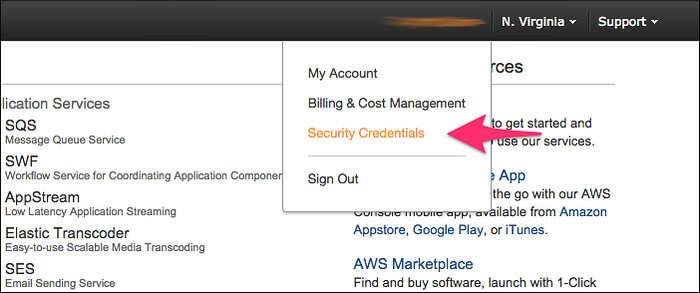
MicroBOSH를 배포하기 위한 AWS 사전 준비사항 및 필요한 사항을 아래의 가이드를 참조하여 설정한다.

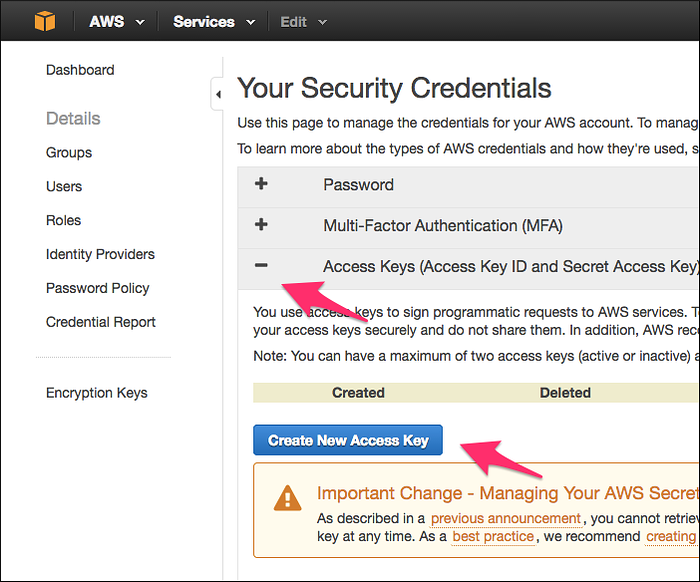
* 사전 준비 및 확인 사항
  + AWS 계정
* Access Key 설정

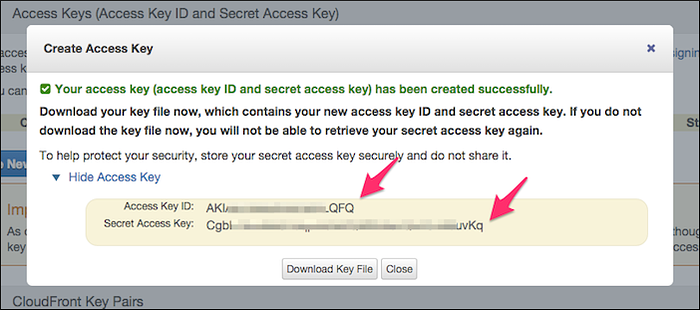
1. AWS에 로그인: <https://console.aws.amazon.com/console/home>



1. 화면 우측 상단의 계정을 선택하여 Security Credentials를 선택



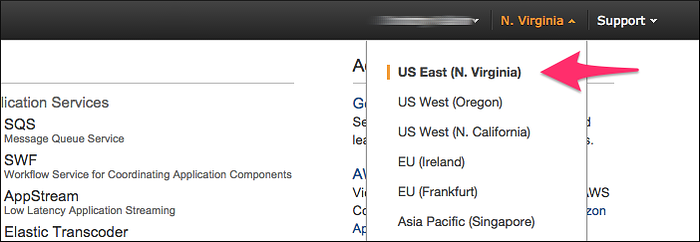
1. 'AWS IAM' 확인 팝업이 나타나면 'Continue to Security Credentials' 버튼을 선택하여 Security Credentials 화면으로 이동
2. Access Keys를 선택하여 Create New Access Key 버튼을 눌러 Access Key를 생성한다.
3. 생성한 키 정보를 확인한다.

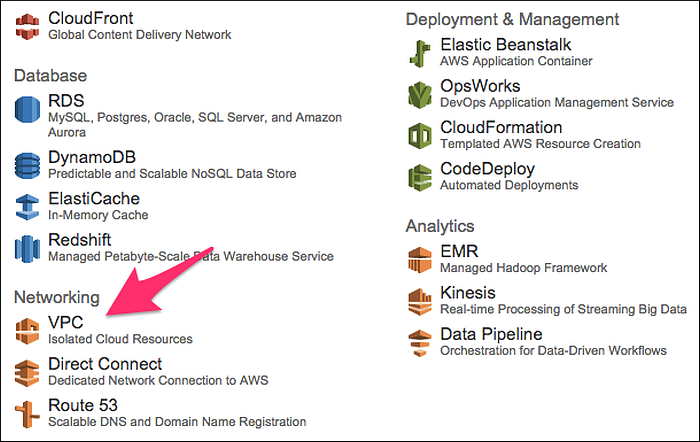


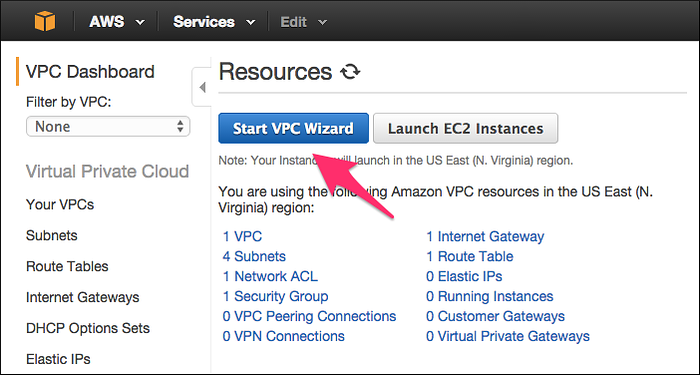
화면의 Access Key ID를 manifest의 access\_key\_id에 설정한다.

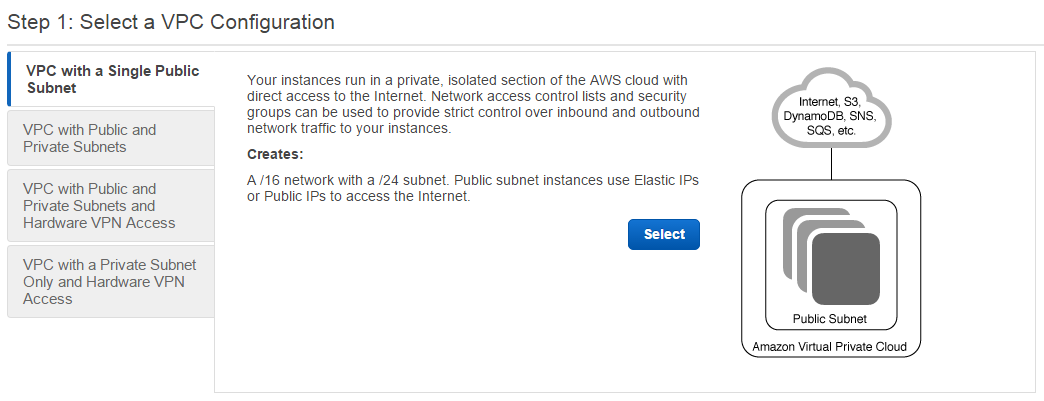
화면의 Secret Access Key를 manifest의 secret\_access\_key에 설정한다.

1. 다이얼로그 화면을 닫는다.

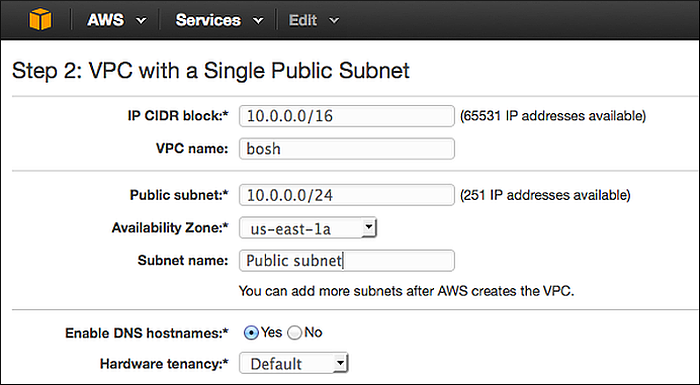
* Virtual Private Cloud (VPC) 구성
  1. 화면 우측 상단의 지역메뉴를 선택한다. (현재 N. Virginia 지역에서만 light stemcell을 사용할 수 있다.)
  2. AWS 콘솔 화면에서 VPC 메뉴를 선택한다.



* 1. VPC 마법사를 선택한다.
  2. “VPC with a Single Public Subnet” 선택



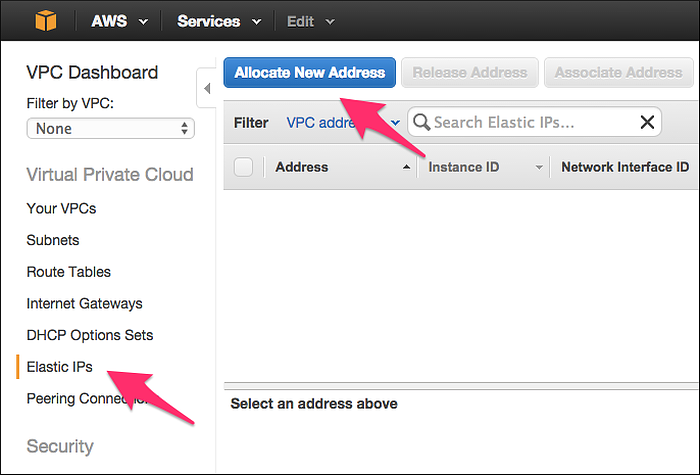
* 1. 다음과 같이 네트워크 정보를 입력하고 VPC 생성 버튼을 눌러 VPC를 생성한다.

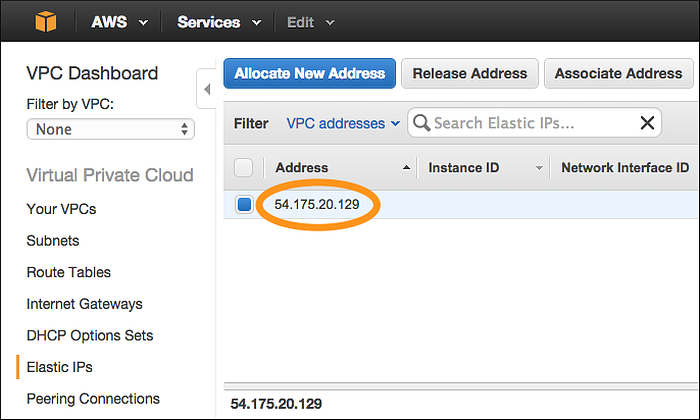


* 1. 아래와 같이 생성한 VPC의 목록이 출력된다.

Subnet ID, CIDR, Availability Zone을 manifest의 네트워크에 설정한다.

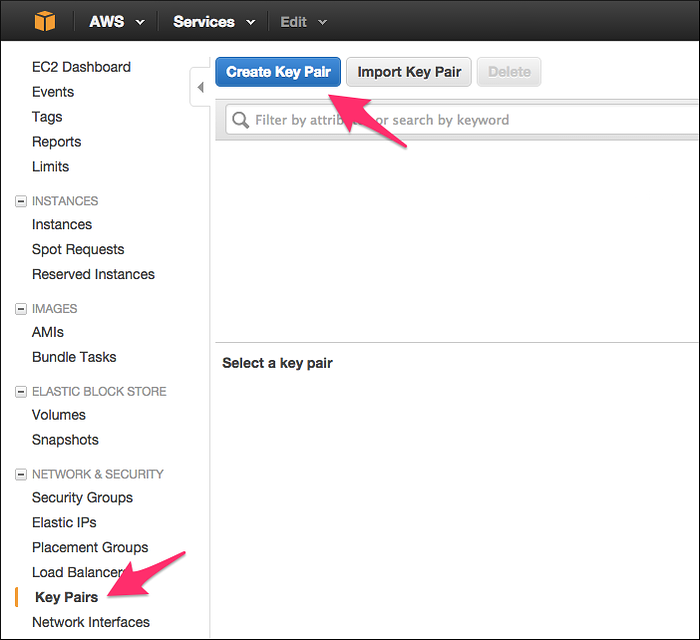
* Elastic IP 생성
  1. VPC 대시보드에서 ‘Elastic IPs’ 버튼과 ‘Allocate New Address’ 버튼을 차례로 눌러 Elastic IP를 생성한다.

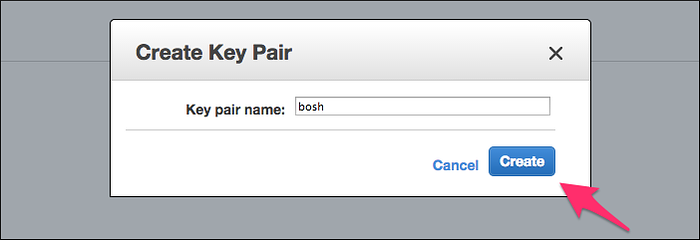


* 1. 팝업창에서 ‘Yes, Allocate’ 버튼을 눌러 Elastic IP를 생성한다.
  2. 생성한 Elastic IP를 확인한다.

Manifest 설정에서 Elastic IP를 사용한다.

* Key Pair 생성
  1. AWS 콘솔 화면에서 ‘EC2’ 메뉴를 선택하여 EC2 대시보드 화면으로 이동한다.
  2. ‘Key Pairs’와 ‘Create Key Pair’ 버튼을 차례로 선택한다.



* 1. Key Pair 생성 다이얼로그 화면에서 Key Pair명을 입력하여 Key Pair를 생성한다.

생성한 Key를 다운로드 한다.

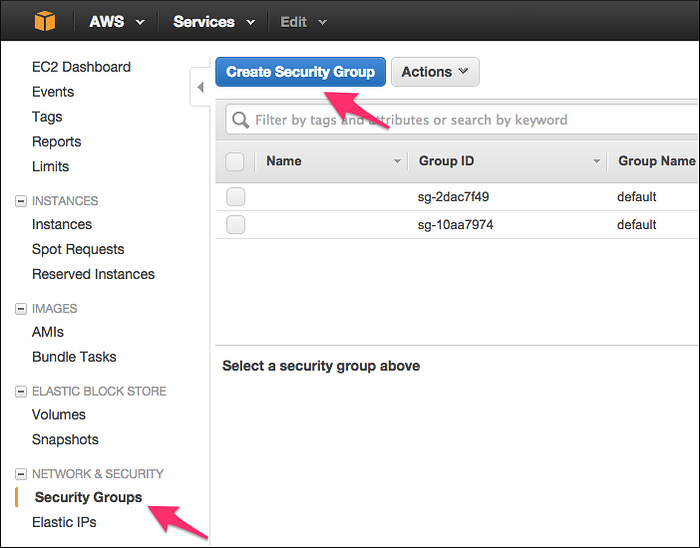
Key pair name에 입력한 값을 manifest의 default\_key\_name에 설정한다.

* 1. 다운로드한 Key(예: bosh.pem)를 키 보관 디렉토리에 옮기고 권한을 변경한다.

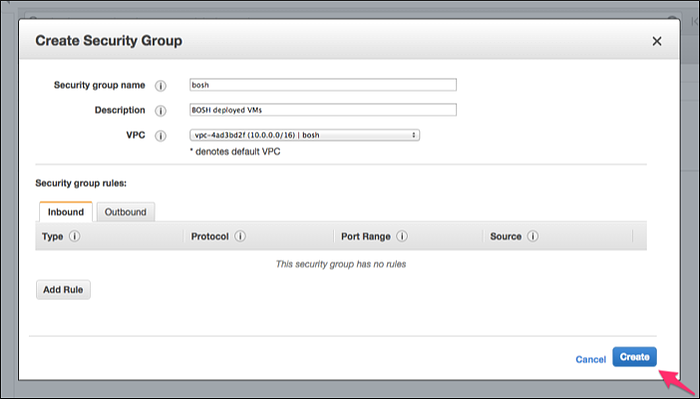
|  |
| --- |
| # Key pair name이 ‘bosh’인 키를 ‘~/Downloads’ 디렉토리에 다운로드 한 경우  $ mv ~/Downloads/bosh.pem ~/.ssh/  $ chmod 400 ~/.ssh/bosh.pem |

‘~/.ssh/bosh.pem’ 값을 manifest의 private\_key에 설정한다.

* 시큐리티 그룹 생성
  1. EC2 대시보드 화면에서 ‘Security Groups’과 ‘Create Security Group’ 버튼을 차례대로 누른다.



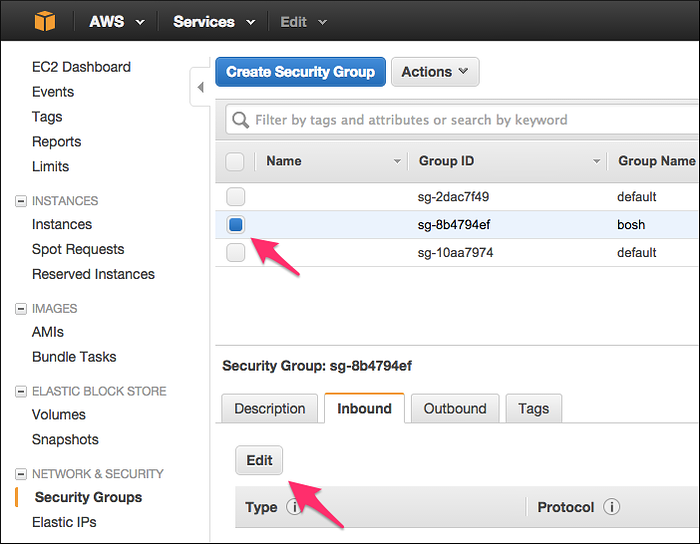
* 1. 시큐리티 그룹 생성 팝업화면에서 다음과 같이 값을 입력하여 시큐리티 그룹을 생성한다.



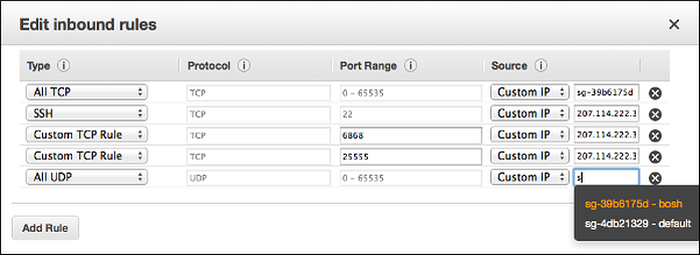
Security group name: bosh

VPC: VPC 구성에서 생성한 VPC 선택

* 1. 생성한 시큐리티 그룹에 보안정책을 설정하기 위해 ‘Inbound’ 탭의 ‘Edit’을 선택한다.



* 1. 아래표와 같이 보안정책을 설정한다.



|  |  |  |  |
| --- | --- | --- | --- |
| Type | Protocol | Port Range | Source |
| Custom TCP Rule | TCP | 6868 | My IP |
| Custom TCP Rule | TCP | 25555 | My IP |
| SSH | TCP | 22 | My IP |
| All TCP | TCP | 0-65535 | Custom IP |
| All UDP | UDP | 0-65535 | Custom IP |

## ***배포 Manifest 작성***

MicroBOSH 배포 Manifest는 배포에 필요한 컴포넌트 및 속성 정보를 YAML 파일 형식으로 정의한다.

1. Manifest파일을 저장할 Local 배포 디렉토리를 생성한다.
2. 생성한 디렉토리에서 배포 Manifest YAML 파일을 생성하고, 아래의 MicroBOSH 배포 Manifest Template을 복사하여 붙여넣고, 붉은 글씨 부분을 자신의 AWS 환경에 맞게 편집 후 저장한다. 입력값은 ‘OpenPaaS-Deployment/openpaas-bosh-init-aws-1.0.yml’ 참조

|  |
| --- |
| ---  name: bosh  releases:  - name: bosh # bosh 릴리즈  url: file:///***<로컬 경로>/<릴리즈 파일명>*** # bosh 릴리즈 파일이 있는 경로 및 파일명  - name: bosh-aws-cpi # bosh-aws-cpi 릴리즈  url: file:///***<로컬 경로>/<릴리즈 파일명>*** # bosh-aws-cpi 릴리즈 파일이 있는 경로 및 파일명  resource\_pools:  - name: vms  network: private  stemcell:  url: file:///***<로컬 경로>/<stemcell 파일명>*** # stemcell 파일이 있는 경로 및 파일명  env:  bosh: # bosh-init의 VM password: admin  password: ***$6$gWEwKrGEeUO$79t/Os3vZoKnVpNkzQwEXMNpveTDYhnhnfhvrg8ShvK705ktXxRv/z3rIsJKei9Zb5u3d38oPL/Crsc1UpJGX1***  cloud\_properties:  instance\_type: m3.xlarge  ephemeral\_disk: {size: 25\_000, type: gp2}  availability\_zone: ***AVAILABILITY-ZONE*** # Availability Zone  disk\_pools:  - name: disks  disk\_size: 20\_000  cloud\_properties: {type: gp2}  networks:  - name: private  type: manual  subnets: # MicroBOSH를 설치할 VPC 정보  - range: ***10.0.0.0/24*** # 네트워크 대역  gateway: ***10.0.0.1*** # VPC 게이트웨이  dns: [***10.0.0.2***] # VPC DNS  cloud\_properties: {subnet: ***SUBNET-ID***} # VPC의 subnet id  - name: public  type: vip  jobs:  - name: bosh  instances: 1 # 설치 VM 수  templates: # 설치할 템플릿 목록 및 템플릿이 참조할 릴리즈명  - {name: nats, release: bosh}  - {name: redis, release: bosh}  - {name: postgres, release: bosh}  - {name: blobstore, release: bosh}  - {name: director, release: bosh}  - {name: health\_monitor, release: bosh}  - {name: registry, release: bosh}  - {name: aws\_cpi, release: bosh-aws-cpi}  resource\_pool: vms # 참조할 리소스풀 명칭  persistent\_disk\_pool: disks # 참조할 디스크풀 명칭  networks: # job에 설정할 네트워크 정보  - name: private  static\_ips: [***10.0.0.6***] # VM에 설정할 내부 아이피  default: [dns, gateway]  - name: public  static\_ips: [***ELASTIC-IP***] # VM에 설정할 ELASTIC 아이피  properties: # job에 설정할 프로퍼티 정보  nats: # nats 프로퍼티 정보  address: 127.0.0.1  user: nats  password: nats-password  redis: # redis 프로퍼티 정보  listen\_address: 127.0.0.1  address: 127.0.0.1  password: redis-password  postgres: &db # postgres db 프로퍼티 정보  listen\_address: 127.0.0.1  host: 127.0.0.1  user: postgres  password: postgres-password  database: bosh  adapter: postgres  registry: # registry 프로퍼티 정보  address: ***10.0.0.6*** # VM에 설정할 내부 아이피  host: ***10.0.0.6*** # VM에 설정할 내부 아이피  db: \*db  http: {user: admin, password: admin, port: 25777}  username: admin  password: admin  port: 25777  blobstore: # blobstore 프로퍼티 정보  address: ***10.0.0.6*** # VM에 설정할 내부 아이피  port: 25250  provider: dav  director: {user: director, password: director-password}  agent: {user: agent, password: agent-password}  director: # 디렉터 프로퍼티 정보  address: 127.0.0.1  name: my-bosh # 디렉터명  db: \*db  cpi\_job: aws\_cpi # 플러그인 cpi명  max\_threads: 10 # 디렉터의 worker thread 개수  user\_management:  provider: local  local:  users:  - {name: admin, password: admin}  - {name: hm, password: hm-password}  hm: # hm 프로퍼티  director\_account: {user: hm, password: hm-password}  resurrector\_enabled: true # job이 중지된 경우, 해당 job에 대한 재시작 여부 설정  aws: &aws  access\_key\_id: ***ACCESS-KEY-ID*** # aws access id  secret\_access\_key: ***SECRET-ACCESS-KEY*** # aws access key  default\_key\_name: ***bosh*** # 기본 키 이름  default\_security\_groups: [***bosh***] # 기본 시큐리티 그룹명  region: ***us-east-1*** # aws 지역  agent: {mbus: "nats://nats:nats-password@***10.0.0.6***:4222"} # VM에 설정할 내부 아이피  ntp: &ntp [***0.pool.ntp.org, 1.pool.ntp.org***] # ntp 서버 설정  cloud\_provider:  template: {name: aws\_cpi, release: bosh-aws-cpi}  ssh\_tunnel:  host: ***ELASTIC-IP*** # VM에 설정할 ELASTIC 아이피  port: 22 # VM에 설정할 ssh 포트  user: vcap # VM에 설정할 사용자 아이디  private\_key: ***./bosh.pem*** # VM과 연결할 수 있는 개인키 정보  mbus: "https://mbus:mbus-password@***ELASTIC-IP***:6868" # VM에 설정할 ELASTIC 아이피  properties:  aws: \*aws  agent: {mbus: "https://mbus:mbus-password@0.0.0.0:6868"}  blobstore: {provider: local, path: /var/vcap/micro\_bosh/data/cache}  ntp: \*ntp |

## ***MicroBOSH 배포***

Bosh-init을 이용해 MicroBOSH를 배포한다.

1. 터미널 상에 bosh-init의 deploy를 실행한다.

|  |
| --- |
| $ bosh-init deploy ***<작성한 manifest>*** |

※ Deploy를 실행하면 manifest파일 있는 디렉토리에 <manifest명>-state.json 파일이 생긴다. MicroBOSH를 삭제할 경우 필요한 파일이므로 삭제하지 않는다.

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

|  |
| --- |
| $ bosh target ***<ELASTIC-IP>***  **$ bosh target 172.16.xxx.xxx**  Target set to `my-bosh'  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 my-bosh **# Director 이름**  URL **https:// *ELASTIC*-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 aws\_cpi **# 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 |

1. deploy를 삭제할 경우

|  |
| --- |
| $ bosh-init delete ***<작성한 manifest>*** |

1. bosh-init 배포에 대한 상세한 로그를 출력하고 싶은 경우, 다음과 같이 설정하고 bosh-init을 실행한다.

|  |
| --- |
| # bosh-init 로그출력 레벨 설정  # 설정 가능한 로그 레벨은 debug, info, warn, error, none로 none이 기본 설정되어 있다.  $ export BOSH\_INIT\_LOG\_LEVEL=***<로그 레벨>***  # 로그를 파일로 출력할 경우  $ export BOSH\_INIT\_LOG\_PATH=***<로그를 출력할 경로>/<로그 파일명>*** |

# Install BOSH

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

## ***설치 절차***

BOSH CLI를 5장에서 설치한 MicroBOSH로 타겟 설정한 이후 아래의 절차로 설치를 진행한다.

* 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 Stemcell을 MicroBOSH에 업로드 한다.

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

|  |
| --- |
| $ **bosh upload stemcell *~/workspace/OpenPaaS-Stemcells/<stemcell 파일명>*** |

* Bosh 설치시는 ‘hvm’ 타입이외의 버전을 사용한다.

Stemcell 업로드가 성공하면 나면 bosh stemcells 명령어을 실행하여 stemcell을 확인한다.

|  |
| --- |
| $ **bosh stemcells** |

## ***BOSH Release 업로드***

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

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

$ **bosh upload release *~/workspace/OpenPaaS-IaaS-Controller/<release 파일명>***

release 업로드가 성공하면 나면 bosh releases 명령어을 실행하여 release를 확인한다.

|  |
| --- |
| $ **bosh releases** |

## ***배포 Manifest 작성***

MicroBOSH 배포 Manifest파일 작성과 동일하게 로컬에 YAML파일을 생성하고, 아래의 BOSH 배포 Manifest을 복사하여 붙여넣고, 붉은 글씨 부분을 자신의 AWS 환경에 맞게 편집 후 저장한다. 입력값은 ‘OpenPaaS-Deployment/openpaas-bosh-aws-1.0.yml’ 참조

|  |
| --- |
| ---  name: bosh # bosh 배포명  director\_uuid: ***3d44c981-d458-47b9-8e95-62d07b87c68f*** # MicroBOSH uuid  release: # bosh 릴리즈 정보  name: bosh # 릴리즈명  version: ***latest*** # 릴리즈 버전  compilation:  cloud\_properties:  instance\_type: m1.small # 설치 패키지를 컴파일 할 VM의 타입 정보  network: default # 설치 패키지를 컴파일 할 VM에 할당할 네트워크 정보  reuse\_compilation\_vms: true  workers: 3 # 설치 패키지를 컴파일 할 VM 개수  update:  canaries: 1 # job설치를 시험하기 위한 생성 VM 수  canary\_watch\_time: 3000-120000 # 시험용 VM의 생성 응답대기 시간 (밀리초)  max\_in\_flight: 4 # 동일 Job에 대한 VM을 동시에 생성할 최대 수  update\_watch\_time: 3000-120000 # 시험용 VM의 업데이트 응답대기 시간 (밀리초)  networks:  - name: elastic # 네트워크 이름 설정  type: vip # 네트워크 타입 설정 (타입: manual, dynamic, vip)  cloud\_properties: {}  - name: default  type: manual  subnets: # VPC 네트워크 서브넷 정보 설정  - range: ***10.0.0.0/24*** # VPC 네트워크 대역 설정  dns: [***8.8.8.8***] # VPC 네트워크 DNS 설정  gateway: ***10.0.0.1*** # VPC 네트워크 게이트웨이 설정  static: # 생성할 VM에 할당할 아이피 대역 설정  - ***10.0.0.6 - 10.0.0.20***  cloud\_properties:  security\_groups:  - ***bosh*** # subnet id와 연계한 시큐리티 그룹명  subnet: ***subnet-e8d03a9e*** # VPC의 subnet id  resource\_pools: # VM에 할당할 리소스풀  - name: small # small 타입 리소스  network: default # small 타입 리소스에 할당할 네트워크 이름  size: 6 # small 타입의 job(=VM) 개수  stemcell: # small 타입의 job에 적용할 stemcell 정보  name: ***bosh-aws-xen-ubuntu-trusty-go\_agent*** # stemcell 이름  version: ***latest*** # stemcell 버전  cloud\_properties:  instance\_type: m1.small # 리소스에 적용할 aws flavor 타입  - name: medium # medium 타입 리소스  network: default  size: 1  stemcell:  name: ***bosh-aws-xen-ubuntu-trusty-go\_agent***  version: ***latest***  env:  bosh: # medium 타입의 VM에 적용할 sha-512로 암호화된 비밀번호  password: ***$6$JA/VRhS7guR2t$kruB3wpqcgyi7Ql2IZIfVC7WP.ZTxjRJ9hbyJV4Xn/wr.mcaOR0MwqAulcKY5gxkFQ0/DJul7UELCKsQ3LrZ71***  cloud\_properties:  instance\_type: m1.medium  jobs:  - instances: 1 # 생성할 VM 수  name: postgres # job(=VM) 이름  networks: # job에 할당할 네트워크 설정 정보  - default: # 기본 설정 네트워크 정보  - dns # 참조 대상의 DNS (네트워크 이름이 default의 DNS를 참조)  - gateway # 참조 대상의 게이트웨이 (네트워크 이름이 default의 게이트웨이를 참조)  name: default # 참조 대상의 네트워크 이름  static\_ips:  - ***10.0.0.6*** # job에 할당할 아이피  persistent\_disk: 2048 # VM에 persistent 디스크를 할당 할 경우, 디스크 사이즈 설정  resource\_pool: small # job에 할당할 리소스 타입의 이름  template: postgres # job 생성시에 참조할 템플릿 이름  - instances: 1  name: nats  networks:  - default:  - dns  - gateway  name: default  static\_ips:  - ***10.0.0.7***  resource\_pool: small  template: nats  - instances: 1  name: redis  networks:  - default:  - dns  - gateway  name: default  static\_ips:  - ***10.0.0.8***  resource\_pool: small  template: redis  - instances: 1  name: blobstore  networks:  - default:  - dns  - gateway  name: default  static\_ips:  - ***10.0.0.9***  persistent\_disk: 51200  resource\_pool: small  template: blobstore  - instances: 1  name: registry  networks:  - default:  - dns  - gateway  name: default  static\_ips:  - ***10.0.0.10***  resource\_pool: small  template: registry  - instances: 1  name: director  networks:  - default:  - dns  - gateway  name: default  static\_ips:  - ***10.0.0.11***  - name: elastic  static\_ips:  - ***52.99.999.999*** # 디렉터의 경우 관리를 위해 외부에서 접속할 수 있는 아이피를 할당  persistent\_disk: 4096  resource\_pool: medium  templates:  - name: powerdns  release: bosh  - name: director  release: bosh  - instances: 1  name: health\_monitor  networks:  - default:  - dns  - gateway  name: default  static\_ips:  - ***10.0.0.12***  resource\_pool: small  template: health\_monitor  properties:  blobstore:  address: ***10.0.0.9*** # blobstore에 할당한 아이피(job에서 설정한 아이피와 동일)  agent: # agent 설정 정보  password: agent # agent 비밀번호  user: agent # agent 사용자 아이디  director: # 디렉터 설정 정보  password: director # 디렉터 비밀번호  user: director # 디렉터 사용자 아이디  director:  address: ***10.0.0.11*** # 디렉터에 할당한 아이피(job에서 설정한 아이피와 동일)  db:  database: bosh # 데이터베이스 이름  host: ***10.0.0.6*** # postgres에 할당한 아이피(job에서 설정한 아이피와 동일)  password: postgres # 데이터베이스 접속 비밀번호  user: postgres # 데이터베이스 사용자 아이디  name: bosh # 디렉터 이름  dns:  address: ***52.99.999.999*** # PowerDNS에 할당한 아이피(job에서 설정한 아이피와 동일)  database:  name: powerdns  db:  database: bosh  host: ***10.0.0.6*** # postgres에 할당한 아이피(job에서 설정한 아이피와 동일)  password: postgres  user: postgres  password: powerdns  recursor: ***52.99.9.999*** # MicroBOSH에 할당한 아이피  replication:  basic\_auth: replication:zxKDUBeCfKYX  password: powerdns  user: replication  user: powerdns  webserver:  password: powerdns  env: null  hm:  director\_account:  password: admin  user: admin  email\_notifications: false  event\_nats\_enabled: false  http:  password: hm  user: hm  pagerduty\_enabled: false  tsdb\_enabled: false  nats:  address: ***10.0.0.7*** # nats에 할당한 아이피(job에서 설정한 아이피와 동일)  password: nats  port: 4222  user: nats  postgres:  database: bosh  host: ***10.0.0.6*** # postgres에 할당한 아이피(job에서 설정한 아이피와 동일)  password: postgres  user: postgres  redis:  address: ***10.0.0.8*** # redis 에 할당한 아이피(job에서 설정한 아이피와 동일)  password: redis  registry:  address: ***10.0.0.10*** # registry에 할당한 아이피(job에서 설정한 아이피와 동일)  db:  database: bosh  host: ***10.0.0.6*** # postgres에 할당한 아이피(job에서 설정한 아이피와 동일)  password: postgres  user: postgres  http:  password: registry  user: registry  ntp: [***1.kr.pool.ntp.org, 0.asia.pool.ntp.org, 1.asia.pool.ntp.org***] # NTP 서버 설정  aws:  access\_key\_id: ***AKxxxxxxxxxxxxxxxx*** # aws access id  default\_key\_name: ***bosh*** # 기본 키 이름  default\_security\_groups: [***bosh***] # 기본 시큐리티 그룹명  region: ***us-east-1*** # aws 지역  secret\_access\_key: ***oDxxxxxxxxxxxxxxxxxxxxxxxxxxxxx*** # aws access key |

※ VM의 vcap 아이디의 비밀번호는 다음의 절차를 통해 변경할 수 있다.

|  |
| --- |
| 1. 암호화된 비밀번호 생성도구 설치  **$ sudo apt-get install whois**  2. vcap 비밀번호 생성  **$ mkpasswd -m sha-512**  Password: ***vcap@password***  $6$4A2jmWvsaYv2O37d$WtT5ul2L4IeRcYcGljeYpd9NHuA3dbdUWcHuDInLG9542g0sU.j4/fj.0uVfEQuutTxw3FW4DANgIqn1OGVTq1  3. 생성한 암호화된 비밀번호를 manifest의 env.bosh.password에 적용 |

## ***BOSH 배포***

다음의 절차를 이용해서 BOSH를 배포한다.

1. 터미널에서 작성한 BOSH 배포 Manifest파일을 CLI에 설정한다.

|  |
| --- |
| $ bosh deployment ***<작성한 manifest>*** |

1. 아래와 같이 bosh deploy명령어를 이용하여 배포를 실행한다.

|  |
| --- |
| $ bosh -n deploy |

1. Target 설정하면서 기본 계정인 admin/admin으로 로그인한다.

|  |
| --- |
| $ bosh target ***<BOSH-ELASTIC-IP>*** |

1. 배포한 BOSH의 상태정보를 확인한다.

|  |
| --- |
| $ bosh status |

1. BOSH를 삭제할 경우

|  |
| --- |
| $ bosh target *<****MicroBOSH-ELASTIC-IP>***  $ bosh delete deployment ***<BOSH-DEPLOYMENT-NAME>*** |

1. BOSH 배포에서 오류가 발생한 경우, 아래의 명령어를 이용하여 로그를 확인한다.

|  |
| --- |
| $ bosh task ***<BOSH 배포 task 번호>*** --debug > ***<로그 출력 경로>/<로그 파일명>*** |

## ***BOSH 배포 Troubleshooting***

|  |
| --- |
| Stemcell does not contain an AMI for this region (us-west-2c) |

Region과 Availability\_zone을 혼동하여 설정하거나 AWS에서 지원하지 않는 region을 설정한 경우에 발생한다. 올바른 Region(일반적으로 끝자리가 숫자로 되어있다.) 또는 Availability\_zone(일반적으로 끝자리가 알파벳으로 되어있다.)을 설정한다.

|  |
| --- |
| Non-Windows instances with a virtualization type of 'hvm' are currently not supported for this instance type. |

hvm 타입의 stemcell을 지원하지 않는 region 또는 Availability\_zone을 설정한 경우 발생한다. 해당 region에 배포해야 할 경우, hvm타입이 아닌 stemcell을 사용한다.

위 예제이외의 오류에 대해서는 <https://bosh.io/docs/aws-cpi.html>를 참조한다.

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