

ansibleを使って
Dockerコンテナを
プロビジョニングする

For APG Engineer Night

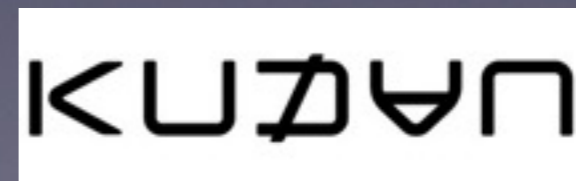
GMO AD MARKETING インフラ開発本部

Adachi Ryo

アジェンダ

- ・ 自己紹介
- ・ 今までの検証環境/docker使うとなにがいいのか
- ・ ansibleとdocker
- ・ docker connection pluginとは
- ・ 環境構築
- ・ dockerホストの監視とdockerホストWebUIについて紹介
- ・ デモ
- ・ まとめ

- @adachin0817
- Ryo Adachi
- GMO AD MARKETING
- Infra Engineer
- <https://adachin.server-on.net/>



そもそもなんでansibleなのか!

- ・ 今まで様々な構成管理ツールを使っていたが、ansibleに統一することにした
- ・ 様々なモジュールがあるため今後の将来性がある
- ・ やっぱり学習コストが低い



ANSIBLE

今までの検証環境



- ・ Vagrantを使いまくっていた
- ・ 構築するのに手間がかかる
- ・ 複数仮想マシンつくと管理がめんどくさい
- ・ dockerで全て検証したい！

dockerを使うメリット

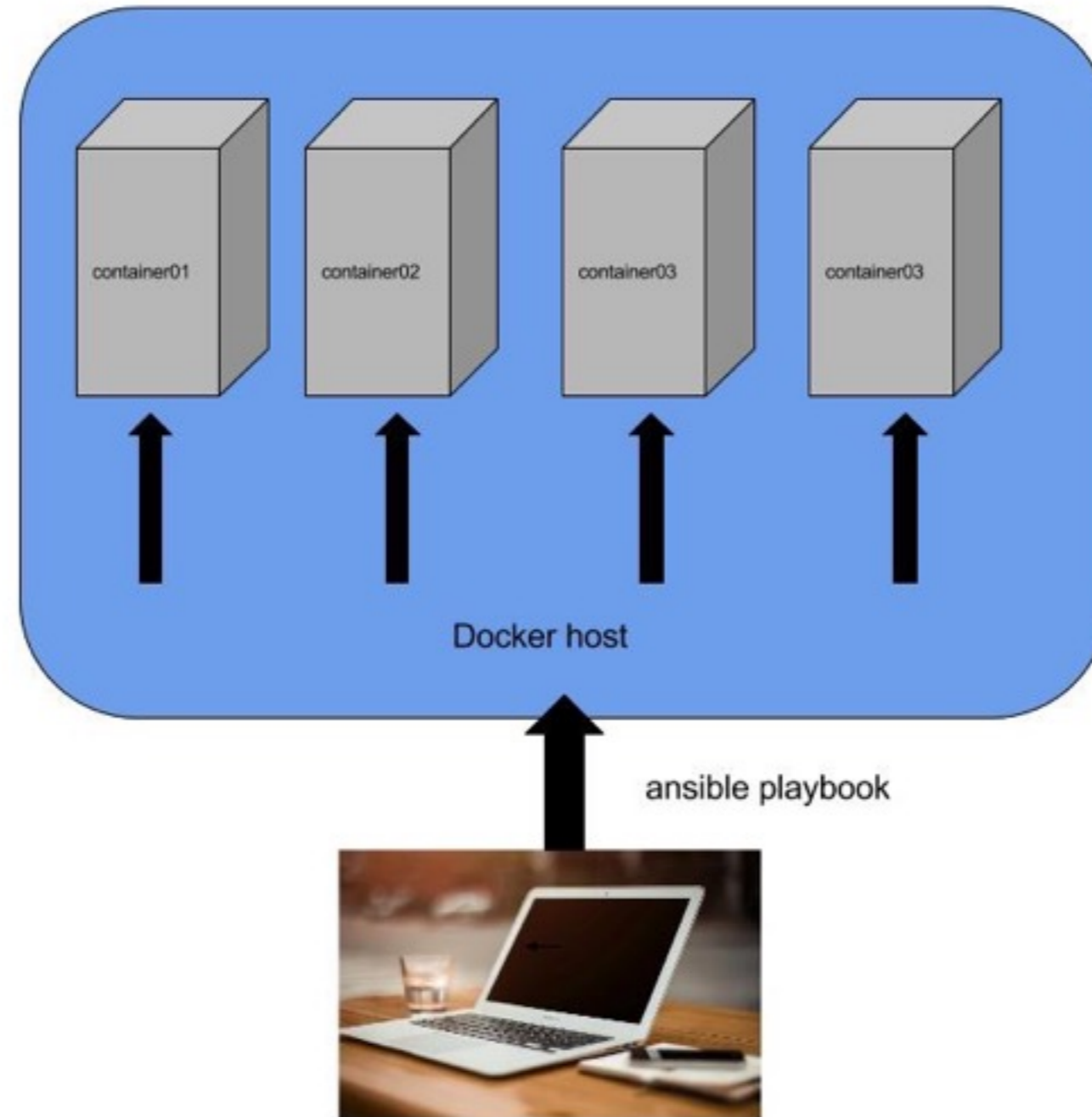
- ・ コンテナ技術が軽すぎる
- ・ ゲストイメージの起動が一瞬
- ・ IPも自動で振られる
- ・ インフラの人じゃなくてもすぐに構築ができる
- ・ サーバに優しい、サーバ構成管理が簡単に可視化できる
- ・ 学習コストはそこまで高くない
- ・ 久しぶりにvagrant系のイメージを消したら30GBも減った



ansibleとdocker

- dockerホストにansibleを入れればOKじゃないの？
→そもそもそんなことしなくていい
- ansible2.0からdockerのConnection Pluginに対応した

イメージ図



docker connection pluginとは

- ・ Docker Remote APIを利用したコンテナのプロビジョニングが可能
- ・ ansible実行環境からDockerホストへRemote APIを実行できる環境であれば、Dockerコンテナでsshdを起動しておく必要はない

ちなみに現実の世界だと



環境構築

- `# docker -v`
Docker version 1.12.1, build b9f10c9
- `$ docker-machine -v`
docker-machine version 0.8.1

ローカルにansibleインストール

- `$ brew install ansible`
- `$ ansible --version`
`ansible 2.1.1.0`

dockerホスト環境設定

- `$ docker-machine ls`
docker01 - virtualbox Running tcp://
192.168.99.100:2376 v1.12.0
- `$ eval "$(docker-machine env docker01)"`
- `$ docker info`
Containers: 0
Running: 0

ansibleイベントファイルの作成

- `$ vim hosts`
`[docker_host]`
`docker01`
- `[container]`
`LB-test01`

ansibleイベントファイルの作成

- `$ vim ssh_config`
Host docker01
HostName 192.168.99.100
User docker
UserKnownHostsFile /dev/null
IdentityFile ~/.docker/machine/machines/
docker01/id_rsa
StrictHostKeyChecking no

ansible イベントファイルの作成

- `$ vim ansible.cfg`
[defaults]
inventory = hosts
[ssh_connection]
ssh_args = -F ssh_config
scp_if_ssh = True
- `cp /etc/ansible or /usr/local/etc/ansible/`

dockerホストにPythonのインストール

- `docker@docker01:~$ tce-load -wi python`
- `docker@docker01:~$ python --version`
Python 2.7.10
- `docker@docker01:~$ curl https://bootstrap.pypa.io/get-pip.py | sudo python -`
- `docker@docker01:~$ sudo pip install docker-py`
- `docker@docker01:~$ sudo ln -s /usr/local/bin/python /usr/bin/python`

dockerホストとの接続確認

- ```
$ ansible -i hosts docker01 -m ping
docker01 | SUCCESS = {
 "changed": false,
 "ping": "pong"
}
```

# プロビジョニング

---

- hosts: docker01

become: yes

remote\_user: docker

tasks:

- name: deploy centos container

docker: image=centos:7 name=LB-test01 privileged=yes ports=7070:7070  
expose=80tty=yes command=/sbin/init

- hosts: LB-test-01

connection: docker #Docker Connection Plugin

- include: roles/common/tasks/yum\_update.yml

tags: yum\_update.yml

- include: roles/common/tasks/yum\_repos.yml

tags: yum\_repos.yml

- include: roles/common/tasks/user.yml

tags: common

# 実行

```
· $ ansible-playbook docker-site.yml
PLAY *****
TASK [setup] *****
ok: [docker01]
TASK [deploy centos container] *****
changed: [docker01]
PLAY [LB-test01] *****
TASK [setup] *****
ok: [LB-test01]
TASK [yum_update.yml] *****
changed: [LB-test01] > (item=[u'yum_update.yml'])
TASK [yum_repos.yml] *****
changed: [LB-test01] > (item= yum_repos.yml)
TASK [common] *****
changed: [LB-test01] > (item= common)
~省略~
docker01 : ok=2 changed=1 unreachable=0 failed=0
test-container : ok=3 changed=2 unreachable=0 failed=0
```

# コンテナが立ち上がってる確認

- docker@docker01:~# docker ps
- | CONTAINER ID | IMAGE    | COMMAND     | CREATED        | STATUS        | PORTS                 | NAMES     |
|--------------|----------|-------------|----------------|---------------|-----------------------|-----------|
| 9fbc512669a2 | centos:6 | "/bin/bash" | 54 minutes ago | Up 54 minutes | 0.0.0.0:80-80/<br>tcp | LB-test01 |

# コンテナログイン

- docker attach ハッシュ値だとコマンドが打てない

```
docker@docker01:~$ docker attach 0ece4f490104
[root@0ece4f490104 /]#
```

コマンドが叩けない怒

- docker exec -it Containername /bin/bashすれば問題なし

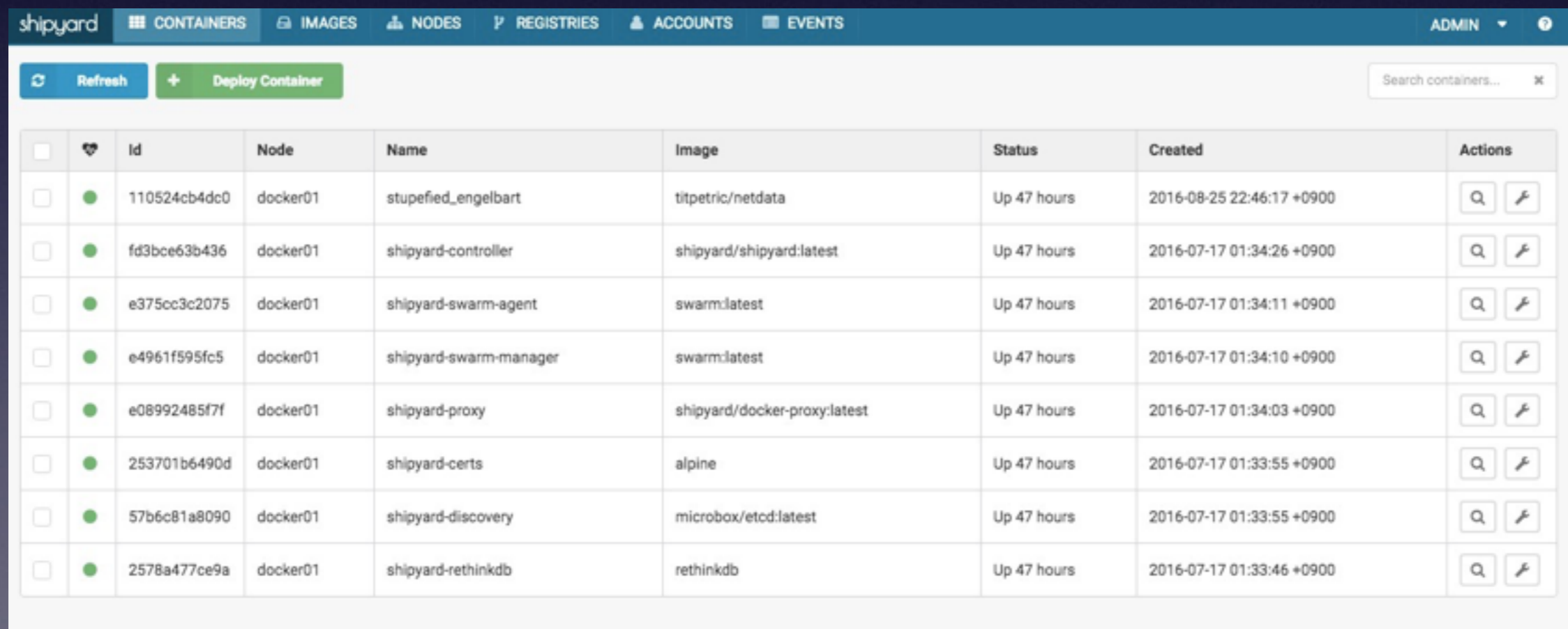
# dockerホストの監視と dockerホストWebUIについて紹介

- netdata(リアルタイムモニタリングツール)



# dockerホストの監視と dockerホストWebUIについて紹介

- shipyard(dockerホストWebUI)



The screenshot displays the shipyard web interface for monitoring Docker containers. The top navigation bar includes 'CONTAINERS', 'IMAGES', 'NODES', 'REGISTRIES', 'ACCOUNTS', and 'EVENTS'. Below the navigation, there are 'Refresh' and 'Deploy Container' buttons, and a search bar for containers. The main content is a table listing several containers with their IDs, nodes, names, images, status, and creation times.

| <input type="checkbox"/> | <input type="checkbox"/>            | Id           | Node     | Name                   | Image                        | Status      | Created                   | Actions                                           |
|--------------------------|-------------------------------------|--------------|----------|------------------------|------------------------------|-------------|---------------------------|---------------------------------------------------|
| <input type="checkbox"/> | <input checked="" type="checkbox"/> | 110524cb4dc0 | docker01 | stupefied_engelbart    | titpetric/netdata            | Up 47 hours | 2016-08-25 22:46:17 +0900 | <input type="checkbox"/> <input type="checkbox"/> |
| <input type="checkbox"/> | <input checked="" type="checkbox"/> | fd3bce63b436 | docker01 | shipyard-controller    | shipyard/shipyard:latest     | Up 47 hours | 2016-07-17 01:34:26 +0900 | <input type="checkbox"/> <input type="checkbox"/> |
| <input type="checkbox"/> | <input checked="" type="checkbox"/> | e375cc3c2075 | docker01 | shipyard-swarm-agent   | swarm:latest                 | Up 47 hours | 2016-07-17 01:34:11 +0900 | <input type="checkbox"/> <input type="checkbox"/> |
| <input type="checkbox"/> | <input checked="" type="checkbox"/> | e4961f595fc5 | docker01 | shipyard-swarm-manager | swarm:latest                 | Up 47 hours | 2016-07-17 01:34:10 +0900 | <input type="checkbox"/> <input type="checkbox"/> |
| <input type="checkbox"/> | <input checked="" type="checkbox"/> | e08992485f7f | docker01 | shipyard-proxy         | shipyard/docker-proxy:latest | Up 47 hours | 2016-07-17 01:34:03 +0900 | <input type="checkbox"/> <input type="checkbox"/> |
| <input type="checkbox"/> | <input checked="" type="checkbox"/> | 253701b6490d | docker01 | shipyard-certs         | alpine                       | Up 47 hours | 2016-07-17 01:33:55 +0900 | <input type="checkbox"/> <input type="checkbox"/> |
| <input type="checkbox"/> | <input checked="" type="checkbox"/> | 57b6c81a8090 | docker01 | shipyard-discovery     | microbox/etcd:latest         | Up 47 hours | 2016-07-17 01:33:55 +0900 | <input type="checkbox"/> <input type="checkbox"/> |
| <input type="checkbox"/> | <input checked="" type="checkbox"/> | 2578a477ce9a | docker01 | shipyard-rethinkdb     | rethinkdb                    | Up 47 hours | 2016-07-17 01:33:46 +0900 | <input type="checkbox"/> <input type="checkbox"/> |



デモやります

# まとめ/今後

- ・ dockerでも問題なくansible使って検証できる
- ・ Dockerfileを作る必要がない
- ・ 社内検証用dockerサーバも構築予定
- ・ アプリチームにはansibleで作ったplaybook(コンテナ)をイメージ側して起動するだけで環境を整えさせる

ご清聴ありがとうございました  
ございました！