OpenStack Ironicによるベアメタルプロビジョニング

2015.01.28 OCPJ Engineering Workshop

Creationline Inc. / OCDET 森 優輝

自己紹介

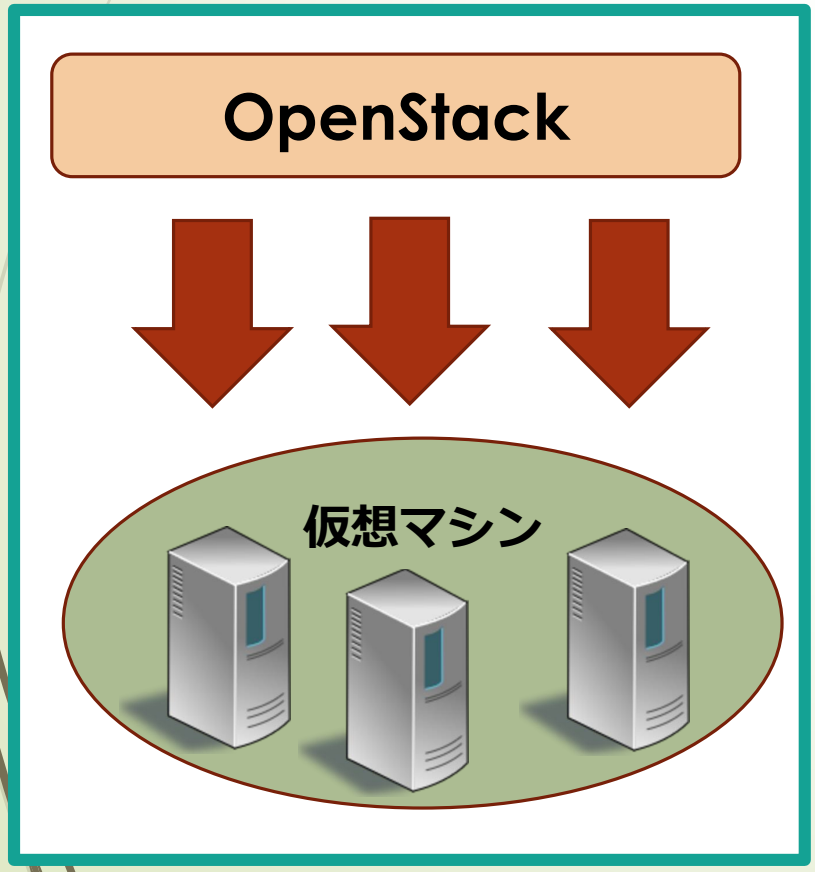
森　優輝

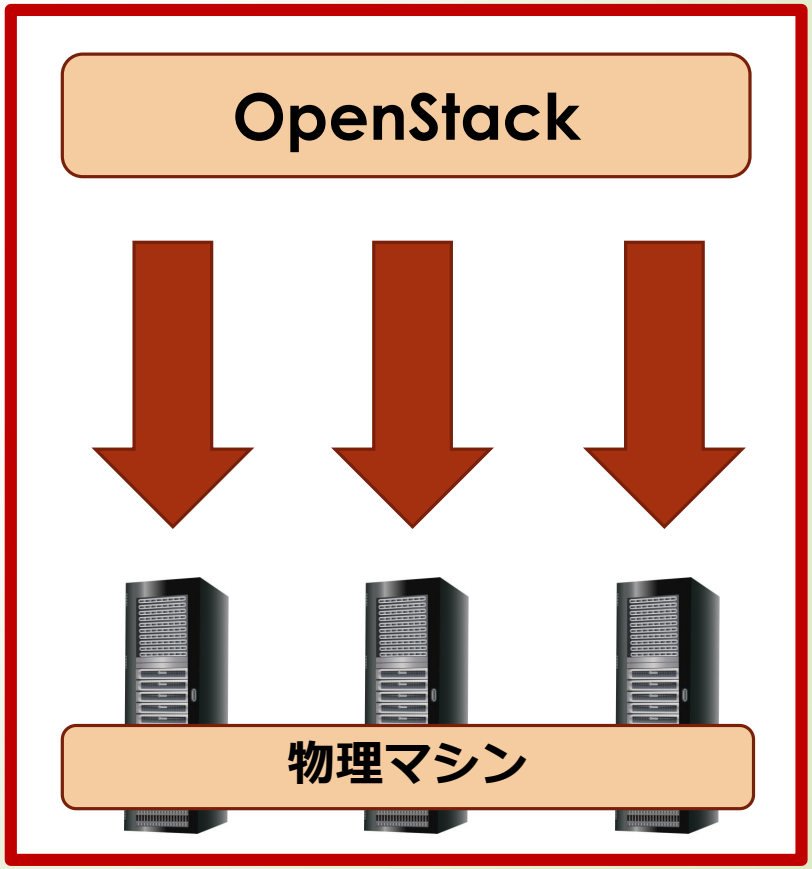
CREATIONLINE,INC.
PLANNING, PROPOSAL, CREATION, AND SOLUTIONS

- クリエーションライン株式会社
- オープンクラウド実証実験タスクフォース(OCDET)
- 電気通信大学
  - 学部2年
- 最近の出来事
  - Kindleで漫画を読むようになった
  - “1-Click” 恐怖症

ベアメタル・プロビジョニング

- ベアメタル：空の物理マシン
- インスタンスを物理マシンに直接展開





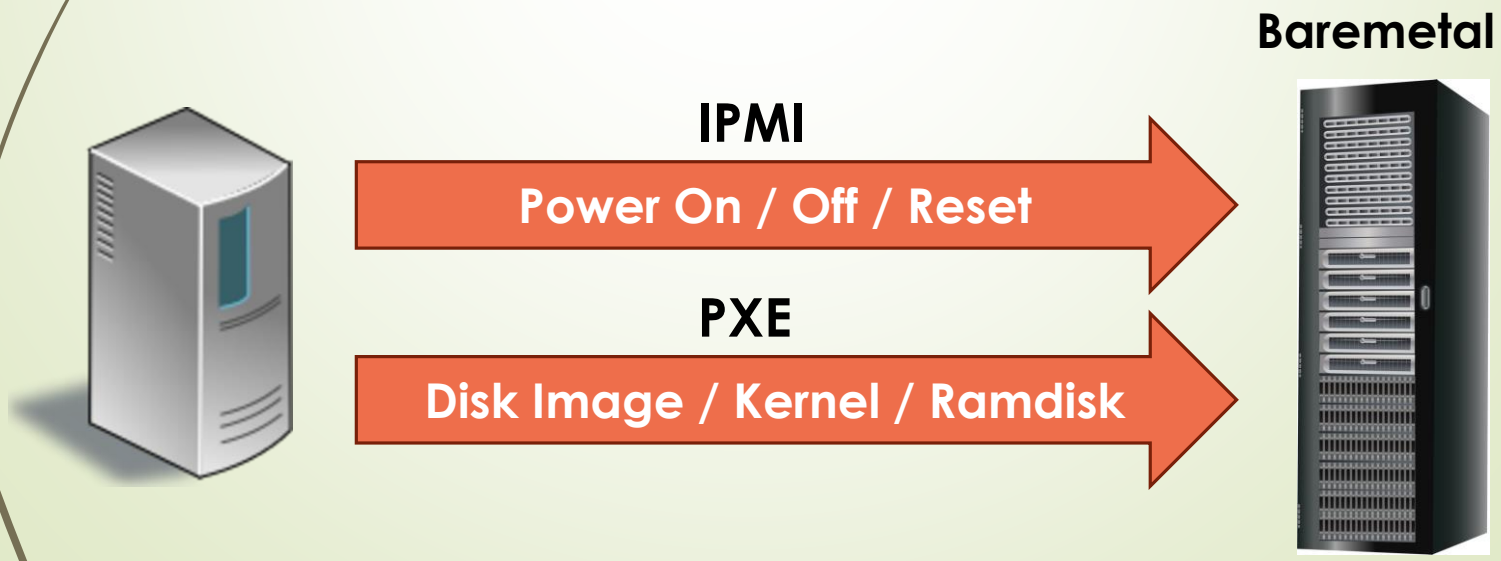
OpenStack Ironic

- ベアメタル・プロビジョニング
  - Nova BareMetal より派生
- OpenStack Kiloで正式採用予定
- マスコットキャラクターも存在する
  - OpenStackの中では初？

“Pixie Boots, the Ironic drummer bear”
by Lucas Alvares Gomes, used under CC BY-SA / Desaturated from original

キーとなる技術

- **IPMI**
  - ノードの電源管理(On/Off/Reset)
- **PXE**
  - プロビジョニング、ノード起動



Ironicのドライバ

- pxe_ipmitool
  - 標準で使用されている
  - PXE + iSCSI + dd
- agent_ipmitool
  - Ironic Python Agent（後述）
- iscsi_ilo
  - HP iLOの仮想メディア機能などを利用してプロビジョニング
- その他

今回はこれを使用して検証しました

Ironic Python Agent

- デプロイ用Ramdiskに組み込まれ、以下の操作が出来るようなREST APIを提供する
  - Erase disks
  - Partition disks
  - Install bootloaders
  - Install an OS image
  - Update firmware
  - Configure RAID
  - And more...
- 今後標準で使われるようになる予定

https://wiki.openstack.org/wiki/Ironic-python-agentより

プロビジョニングの基本的な流れ

Nova

- Ironic用Compute Driver
- フレーバーに合う物理ノードを選択

Ironic

- IPMIで物理ノードの電源ON
- コールバック待機

PXE iSCSI

- Deploy用イメージで起動
- ddでHDDにイメージを書き込み

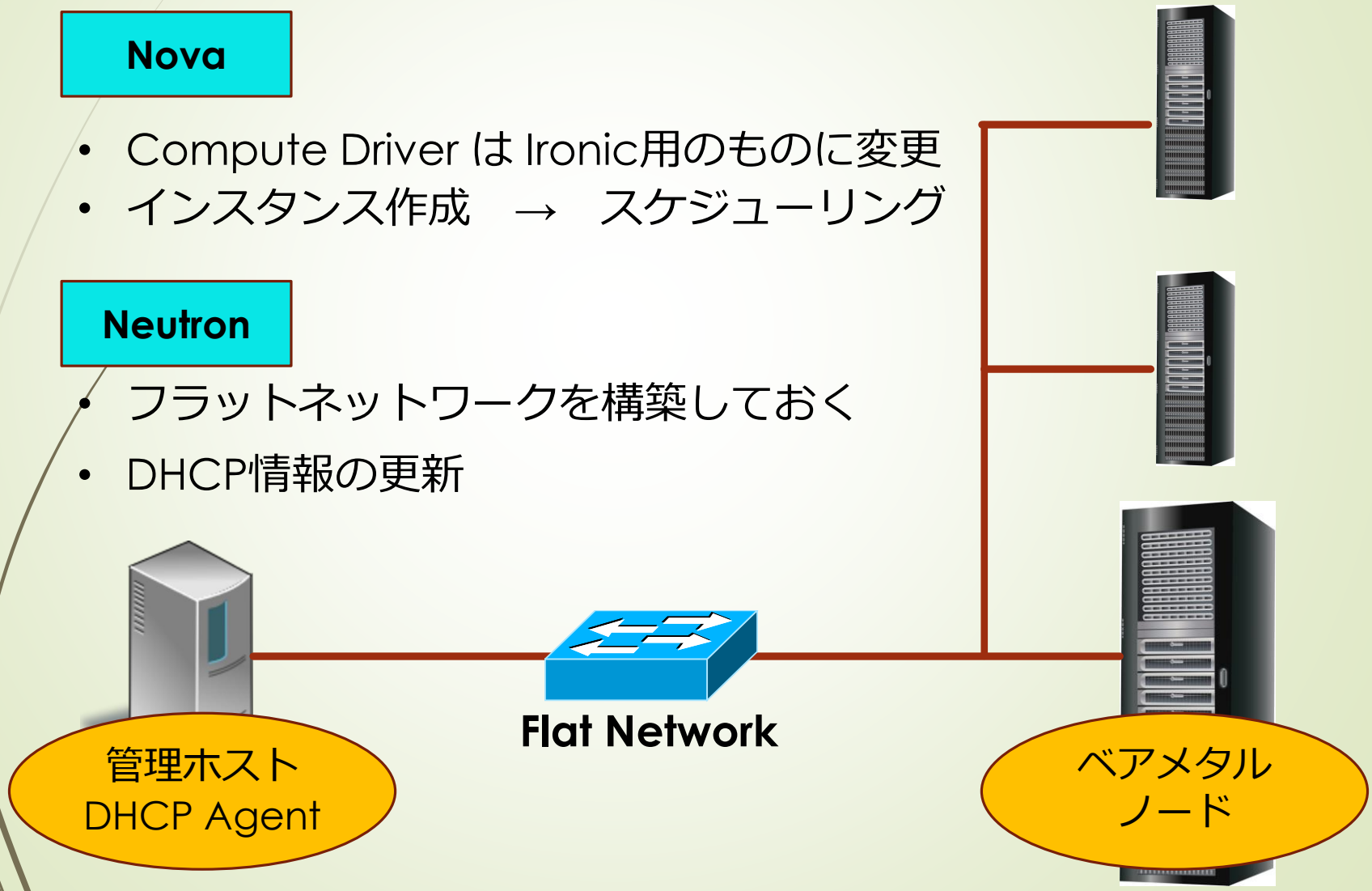
プロビジョニングの基本的な流れ

Nova

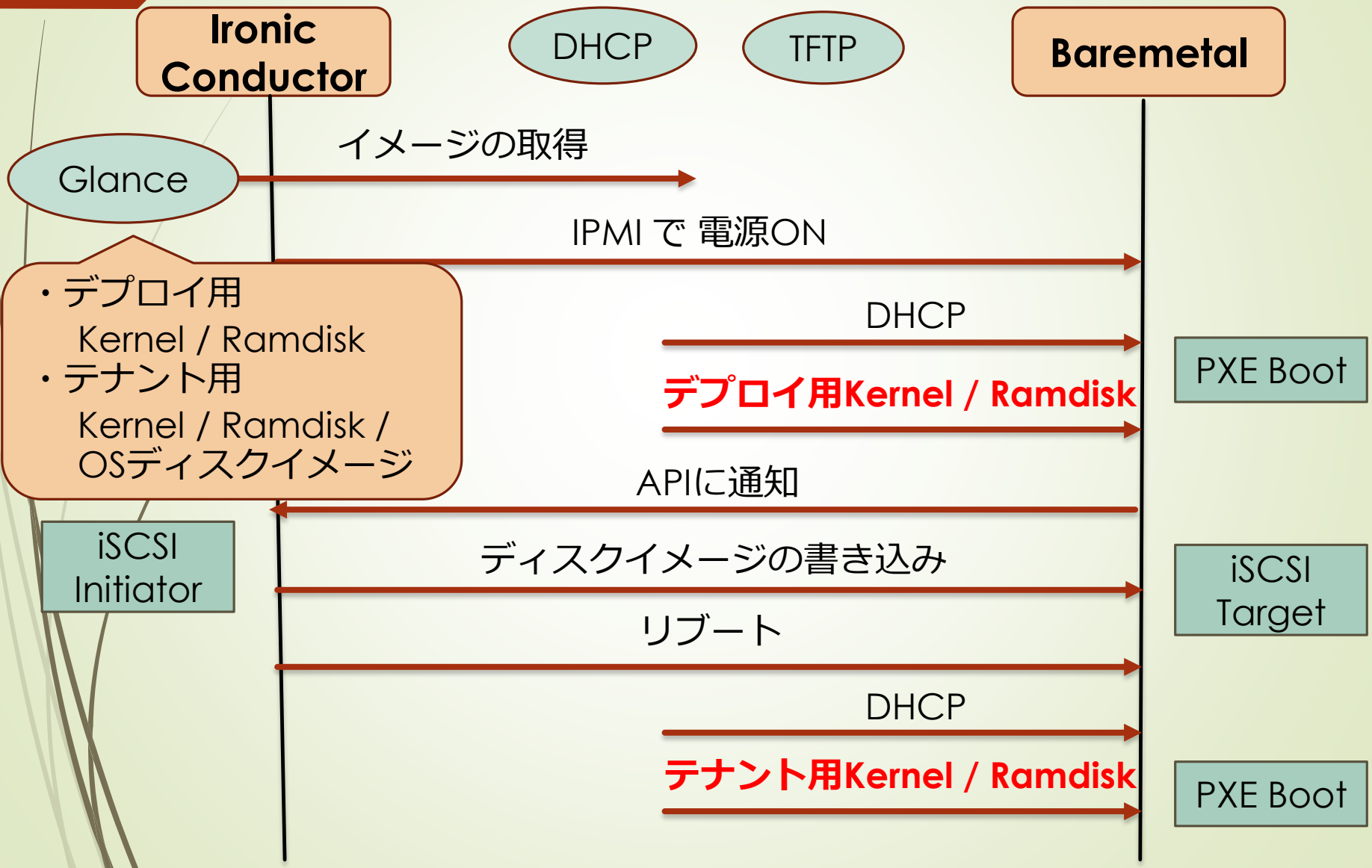
- Compute Driver は Ironic用のものに変更
- インスタンス作成　→　スケジューリング

Neutron

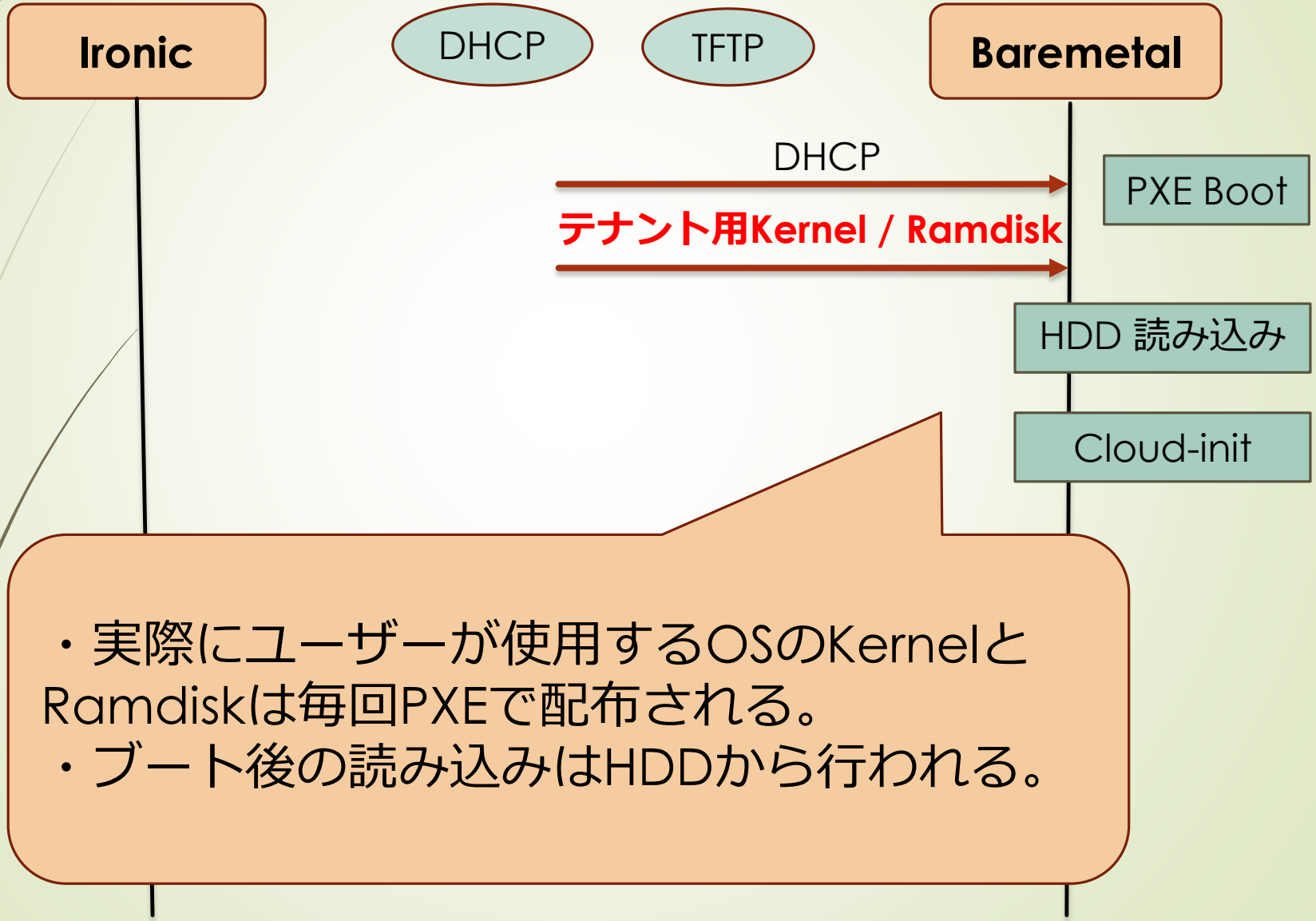
- フラットネットワークを構築しておく
- DHCP情報の更新



プロビジョニングの基本的な流れ



プロビジョニングの基本的な流れ



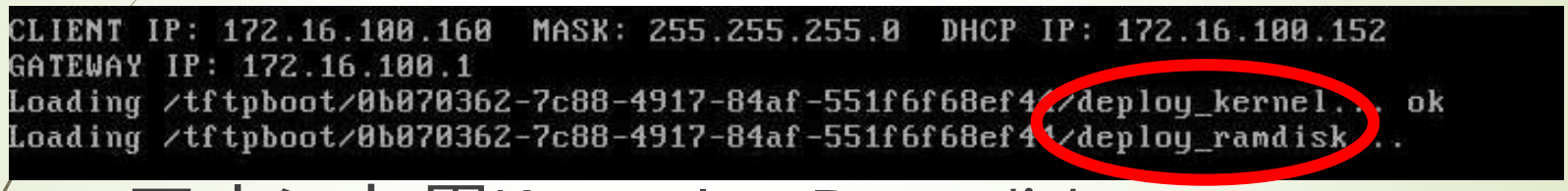
・実際にユーザーが使用するOSのKernelとRamdiskは毎回PXEで配布される。
・ブート後の読み込みはHDDから行われる。

イメージの取得

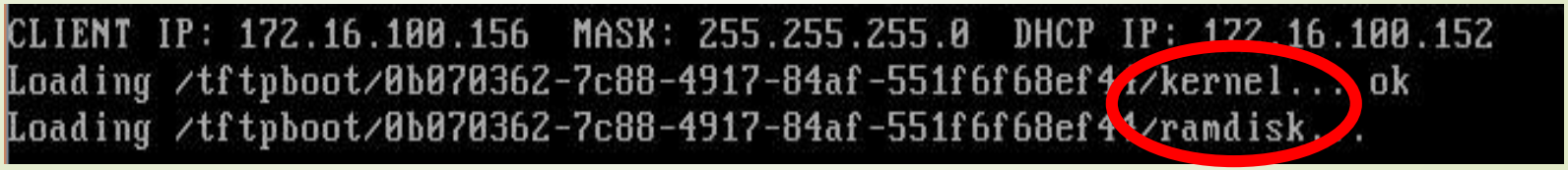
- インスタンス作成時、
  指定されたイメージがTFTPのルート
  ディレクトリにコピーされる
- デプロイ用Kernel & Ramdisk
- テナント用Kernel & Ramdisk &
  OSディスクイメージ

ディスクイメージ

- デプロイ用Kernel & Ramdisk
  - ディスクイメージを書き込むためのミニOS
  - インスタンス作成時のみ使用される



- テナント用Kernel & Ramdisk & OSディスクイメージ
  - 実際にユーザが使用するもの



ディスクイメージの書き込み

1. ベアメタルノードからIronic APIにミニOSが起動したことを通知
2. iSCSI経由で接続
3. HDDの先頭と末尾18KBをゼロ埋め
4. パーティション作成
5. OSディスクイメージの書き込み
6. ベアメタルノードからIronic APIにデプロイが完了したことを通知

ディスクイメージの作成

- Diskimage-builderを使用
  - https://github.com/openstack/diskimage-builder
- disk-image-create コマンド
  - Kernel(aki) & Ramdisk(ari) & OSディスクイメージ (qcow2)
  - Elementsはリポジトリ内のelementsディレクトリ参照
  - 例：Ubuntuイメージの作成

    disk-image-create -o my-image ¥

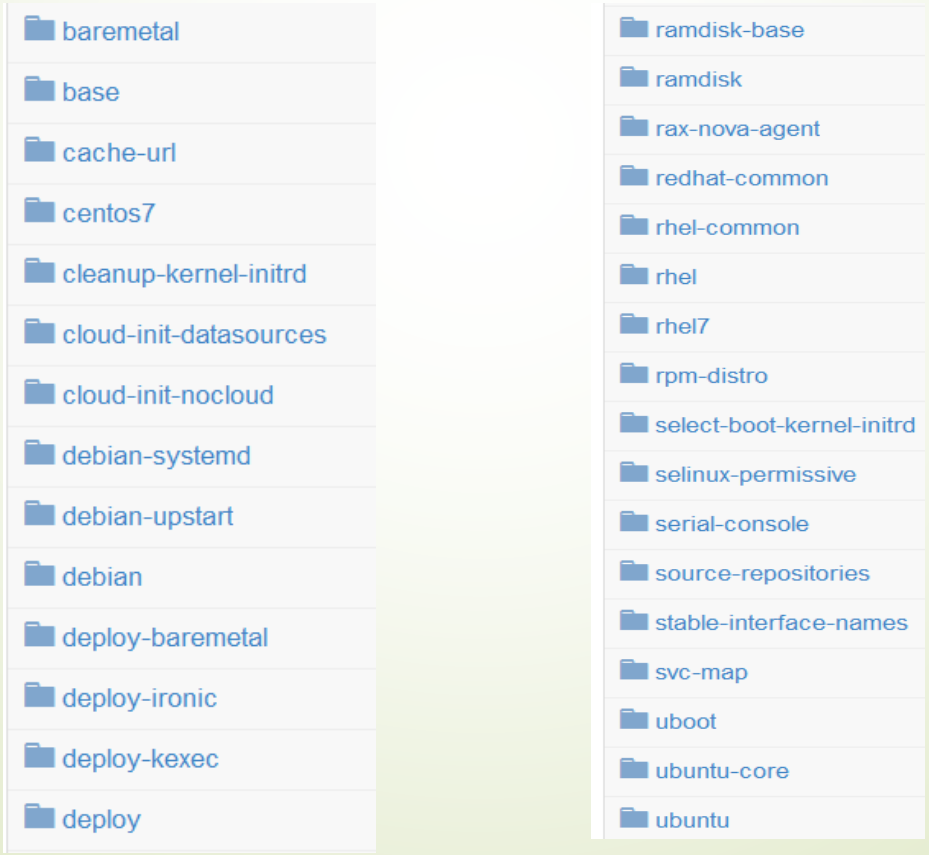
    ubuntu baremetal

Ubuntuの他に、
・**Fedora**
・**CentOS 7**
・**Debian**
などが指定できる

・**baremetal**
- KernelとRamdiskを抽出する

ディスクイメージの作成

- Elements
  - https://github.com/openstack/diskimage-builder/tree/master/elements



デプロイイメージの作成

- Diskimage-builderを使用
- ramdisk-image-create コマンド
  - Kernel(aki) & Ramdisk(ari) を作成
  - *ramdisk-image-create -o my-deploy-ramdisk ¥*
    ubuntu deploy-ironic

Ubuntuの他に、
・**Fedora**
・**CentOS 7**
・**Debian**
などが指定できる

イメージの登録

- Glanceに登録
  - Disk-format
    - Kernel : **aki**, Ramdisk : **ari**
    - OSディスクイメージ: **qcow2**
  - OSディスクイメージのみ、KernelとRamdiskを結び付けて登録する

プロビジョニングの問題・疑問

- ノード起動時に毎回PXEでKernelとRamdiskをロード
  - LinuxはいいがWindowsはどうするの？
- HDDの先頭と末尾18KBをゼロ埋め
  - 全体を消去しないと不安

電源管理

- Ironic Conductorが操作・監視
  - IPMIToolを用いる
    - 1分毎に状態確認 “*power status*”
  - OpenStack側とベアメタルノード側で電源の状態が異なる場合
    - **OpenStack側の状態に合わせる**
    - 「ベアメタル側でシャットダウンした場合、Ironic Conductorがそれを検知し、自動的に電源をONにする。」

電源管理

- 問題点
  - インスタンスをシャットダウン
    - 強制的に電源OFF
    - “***Ipmitool ~~ power off***”
  - インスタンスを再起動
    - “***Ipmitool ~~ power off***”
    - “***Ipmitool ~~ power on***”
  - メンテナンスモードにして監視を抜けるしかない？

ベアメタルノードの登録

- 用意するもの
  - IPMIのIPアドレス, User, Password
    - 電源管理で使用
  - 通信用NICのMACアドレス
    - DHCPで使用

IPMI用NICのもので
はない

ベアメタルノードの登録

- Python製のIronicクライアントを使用
  - ノード登録
    - ***ironic node-create***
  - ノードリスト
    - ***ironic node-list***
  - ノード設定の変更
    - ***ironic node-update***
  - ドライバリスト
    - ***ironic driver-list***

```
ubuntu@juno2:~$ ironic driver-list
+--------------+----------------+
| name         | hosts          |
+--------------+----------------+
| pxe_ipmitool | juno2.internal |
+--------------+----------------+
```

ベアメタルノードの登録

- ノードの作成
  - ***ironic node-create -d pxe_ipmitool***
- IPMI情報の登録
  - ***ironic node-update $NODE_UUID add ¥***
    driver_info/ipmi_username=$USER ¥
    driver_info/ipmi_password=$PASS ¥
    driver_info/ipmi_address=$ADDRESS
- ノードスペックの登録
  - ***ironic node-update $NODE_UUID add ¥***
    properties/cpus=$CPU ¥ CPUの論理コア数
    properties/memory_mb=$RAM_MB ¥
    properties/local_gb=$DISK_GB ¥
    properties/cpu_arch=$ARCH I386 / x86_64

ベアメタルノードの登録

- デプロイ用イメージの紐づけ
  - ***ironic node-update $NODE_UUID add ¥***
    driver_info/pxe_deploy_kernel=$DEPLOY_VMLINUZ_UUID ¥
    driver_info/pxe_deploy_ramdisk=$DEPLOY_INITRD_UUID
- MACアドレスの登録
  - ***ironic port-create -n $NODE_UUID -a $MAC_ADDRESS***

```
ubuntu@juno2:~$ ironic node-list
+--------------------------------------+---------------+-------------+--------------------+-------------+
| uuid                                 | instance_uuid | power_state | provision_state    | maintenance |
+--------------------------------------+---------------+-------------+--------------------+-------------+
| 0b070362-7c88-4917-84af-551f6f68ef44 | None          | power off   | None               | False       |
| 5182fa85-e430-4524-8fdc-91429dcdd456 | None          | power off   | None               | True        |
+--------------------------------------+---------------+-------------+--------------------+-------------+
```

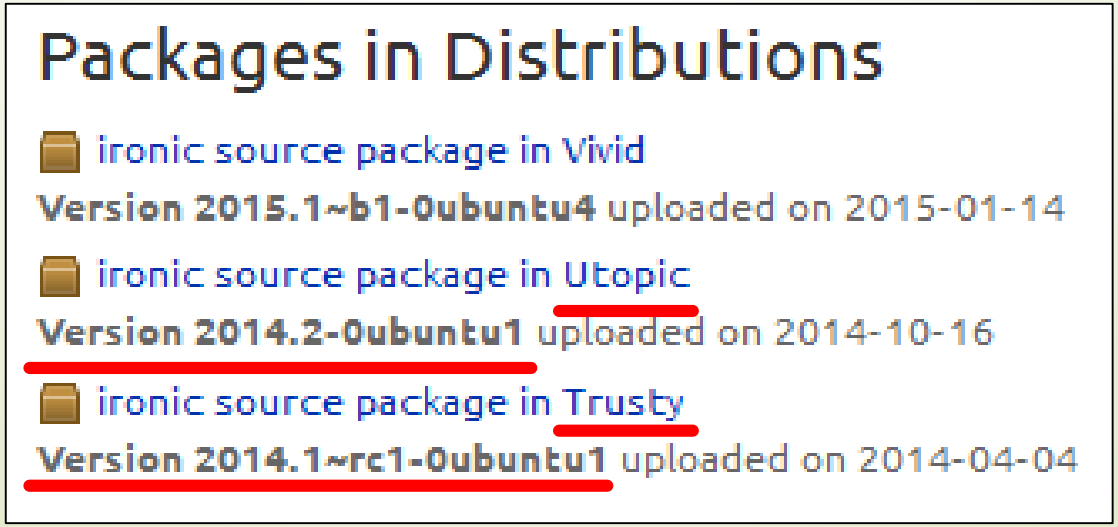
電源状態が
反映されている

Ironic インストール

- 基本的にインストールガイドを元に進める
  - http://docs.openstack.org/developer/ironic/deploy/install-guide.html
- 日々更新されています

Ironicインストール時の苦労話

- 検証環境
  - OpenStack Juno (2014.2)
  - OS: Ubuntu 14.10
    - Ubuntu 14.04 のJunoリポジトリでは、IronicはIcehouse(2014.1)のものしかなかった



Ironicインストール時の苦労話

- Diskimage-builderが実行できない
  - “dib-run-parts not found”
  - dib-utilsをインストールすると解決
    - https://github.com/openstack/dib-utils

Ironicインストール時の苦労話

- TFTPブートができない
  - Dnsmasq.confを作成しdhcp-agentに読ませて解決

```
1 log-facility = /var/log/neutron/dnsmasq.log
2 enable-tftp
3 dhcp-boot  = pxelinux.0,pxeserver,172.16.100.152
4 tftp-root=/tftpboot
```

- 実はIronic.confにて上記を設定できる
  - 要mapファイル

```
1143 [pxe]
1144 pxe_append_params = nofb nomodeset vga=normal console=ttyS0
1145 tftp_master_path = /tftpboot/master_images
1146 tftp_root = /tftpboot
1147 tftp_server = 172.16.100.152
```

Ironicインストール時の苦労話

- iSCSIターゲットに接続できない
  - *“iscsiadm ~~ --login”*
  - *“iscsiadm ~~ --logout”*
    - この間3~4秒
  - *Error: [Errno 2] No such file or directory: ‘/dev/disk/by-path/ip-….’*
    - Ironic-conductor.logより
- **rootwrapファイルの所有者をrootに変えることで解決**
  - そもそもこの問題に直面しなかったという報告もある

Ironicインストール時の苦労話

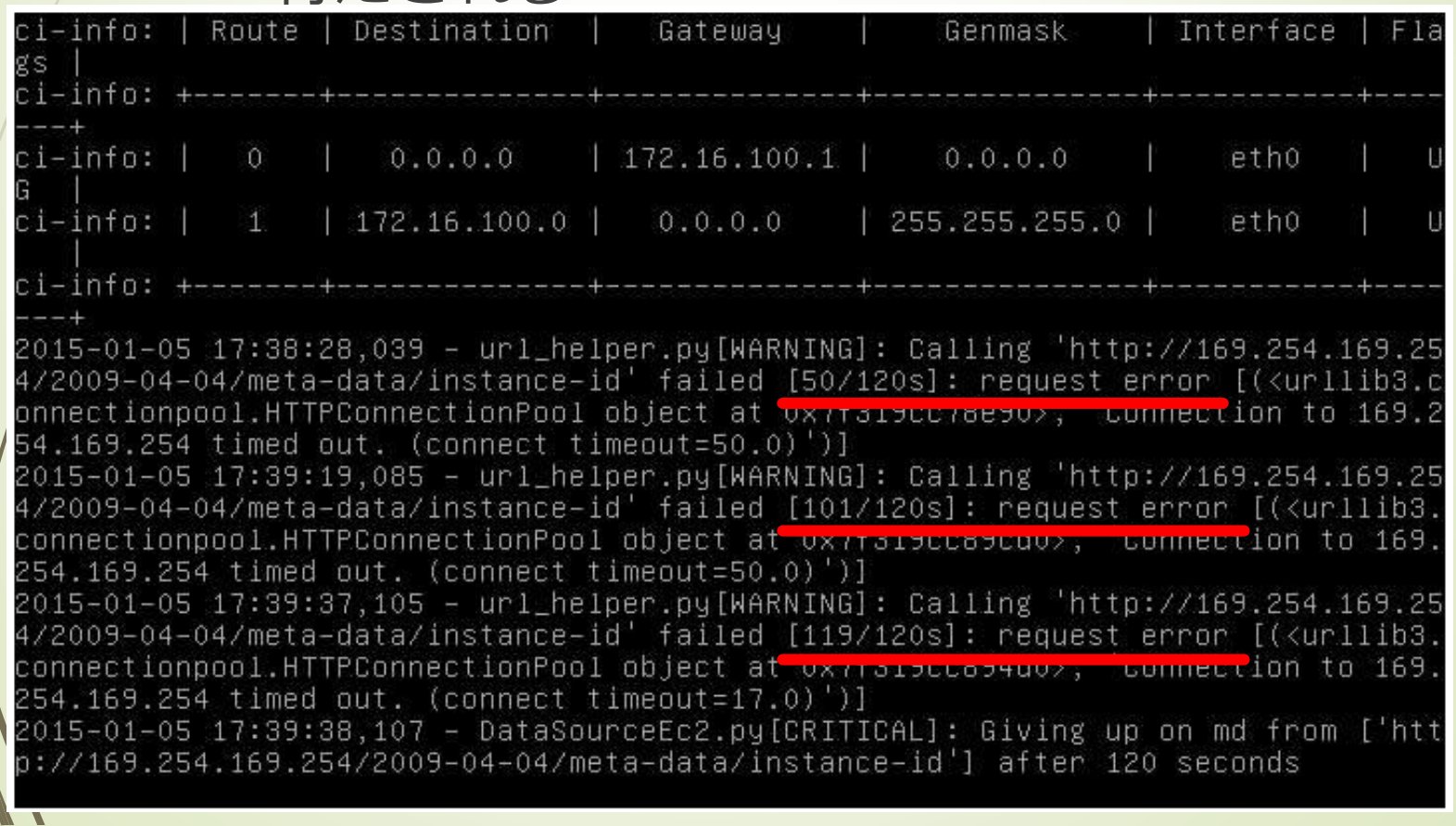
- rootwrapファイルの所有者をrootに変えることで解決

```
root@juno2:~# ll /etc/ironic
total 56
drwxr-xr-x   3 ironic ironic  4096 Jan 27 06:11 ./
drwxr-xr-x 122 root   root    4096 Jan 23 01:37 ../
-rw-r--r--   1 ironic ironic 35782 Jan 26 16:27 ironic.conf
-rw-r--r--   1 ironic ironic   119 Oct 17 00:52 policy.json
-rw-r--r--   1 ironic ironic   939 Jan 15 16:38 rootwrap.conf
drwxr-xr-x   2 ironic ironic  4096 Nov 12 06:50 rootwrap.d/
```

```
root@juno2:~# ll /etc/ironic
total 56
drwxr-xr-x   3 ironic ironic  4096 Jan 27 06:11 ./
drwxr-xr-x 122 root   root    4096 Jan 23 01:37 ../
-rw-r--r--   1 ironic ironic 35782 Jan 26 16:27 ironic.conf
-rw-r--r--   1 ironic ironic   119 Oct 17 00:52 policy.json
-rw-r--r--   1 root   root    939 Jan 15 16:38 rootwrap.conf
drwxr-xr-x   2 root   root    4096 Nov 12 06:50 rootwrap.d/
```

Ironicインストール時の苦労話

- Metadata-agentに繋がらない
  - Cloud-initで120秒間タイムアウトするまで待たされる



Ironicインストール時の苦労話

- Metadata-agentに繋がらない
  - 以下をNeutronのdhcp-agent.iniに追記することで解決
    “**enable_isolated_metadata=True**”

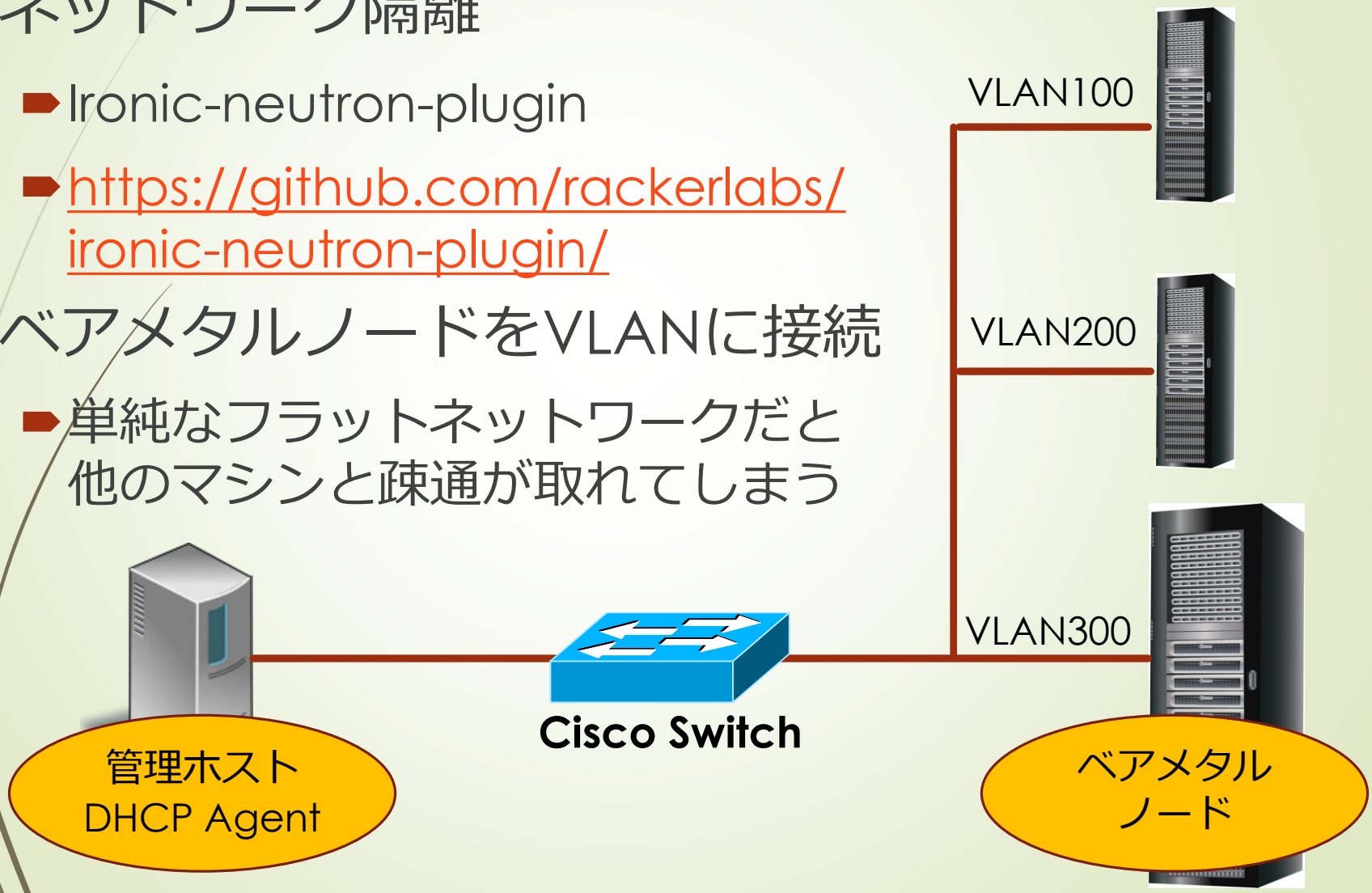
```
38 # The DHCP server can assist with providing metadata support on isolated
39 # networks. Setting this value to True will cause the DHCP server to append
40 # specific host routes to the DHCP request. The metadata service will only
41 # be activated when the subnet does not contain any router port. The guest
42 # instance must be configured to request host routes via DHCP (Option 121).
43 # enable_isolated_metadata = False
44 enable_isolated_metadata = True
```

Ironicインストール時の苦労話

- Metadata-agentに繋がらない
  - 先の方法に気付くまでは…
- ディスクイメージにSSH公開鍵を追加
  - *$ disk-image-create -o my-image ¥*
    *ubuntu baremetal **local-config***
- 上記コマンド実行ユーザの
  $HOME/.ssh/authorized_keys
  をそのままコピーしている
- プロキシの環境変数も引き継ぐ

現在の検証状況

- ネットワーク隔離
  - Ironic-neutron-plugin
  - https://github.com/rackerlabs/ironic-neutron-plugin/
- ベアメタルノードをVLANに接続
  - 単純なフラットネットワークだと他のマシンと疎通が取れてしまう



本日のまとめ

- Ironicは実用的に動作してきている
  - もっと情報が充実してもいい頃？
- 様々な問題点はあるが、今後のドライバ次第
- ネットワーク隔離についてはこれから
  - スイッチとの連携など