
VirtualBox + RancherOS + Docker環境で、
Jupyter Notebookを動かしてみた。

2019/06/15 Python勉強会

自己紹介

- kenichiro90
 - 製造業ですが、実験データの解析でPythonを触ってます。
 - Pythonを始めてから、3年ぐらい経ちました。
 - 本職では無いので、お手柔らかに...
-

今回やったこと

- VirtualBox上に、RancherOSをインストール
- RancherOS上で、Dockerを使い、Jupyter Notebook環境を入れる。

VirtualBoxのインストール

- 公式サイトからパッケージをインストール。



VirtualBox

Download VirtualBox

Here you will find links to VirtualBox binaries and its source code.

VirtualBox binaries

By downloading, you agree to the terms and conditions of the respective license.

If you're looking for the latest VirtualBox 5.2 packages, see [VirtualBox 5.2 builds](#). discontinued in 6.0. Version 5.2 will remain supported until July 2020.

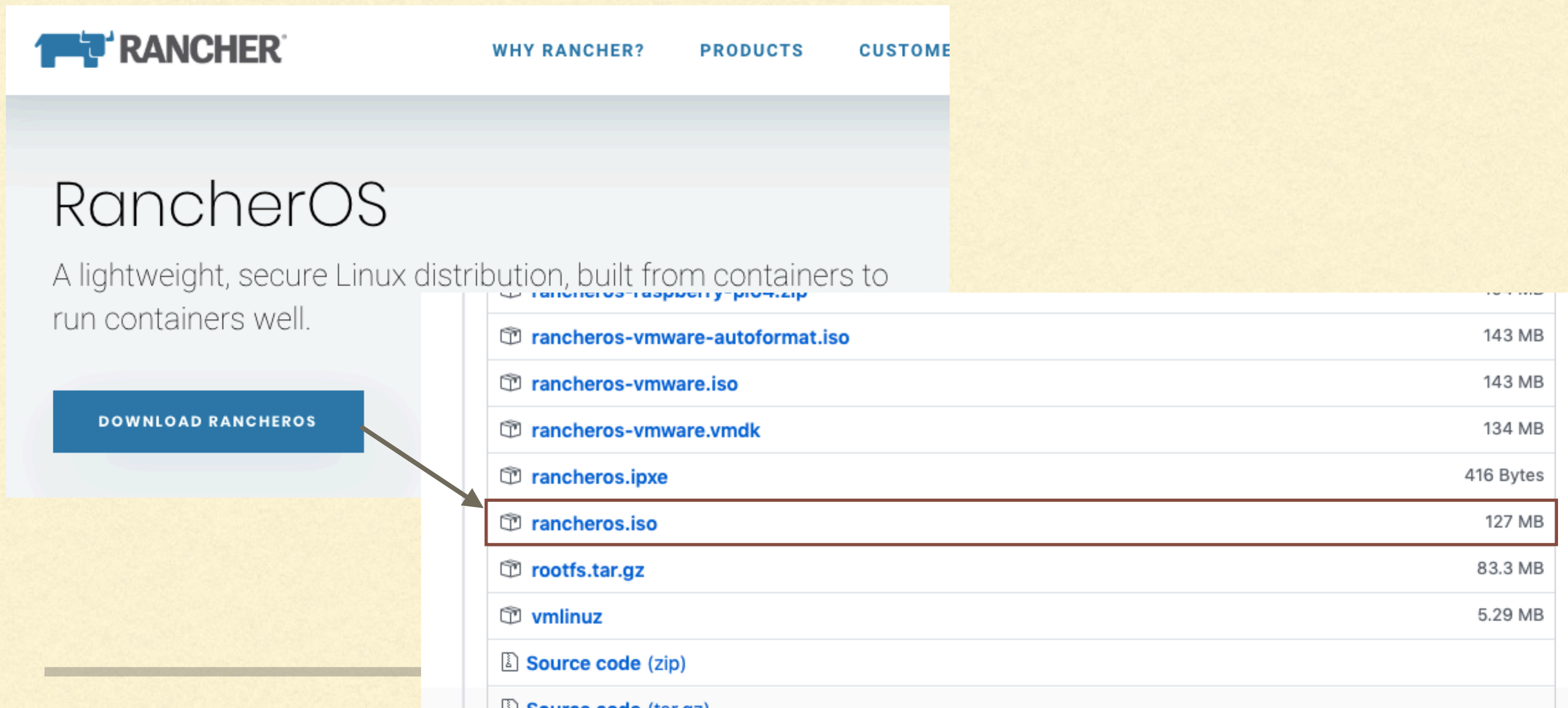
VirtualBox 6.0.8 platform packages

- [Windows hosts](#)
- [OS X hosts](#)
- [Linux distributions](#)
- [Solaris hosts](#)

[About](#)
[Screenshots](#)
[Downloads](#)
[Documentation](#)
 [End-user docs](#)
 [Technical docs](#)
[Contribute](#)
[Community](#)

RancherOSのダウンロード

- githubの方から、”rancheros.iso”をダウンロード。



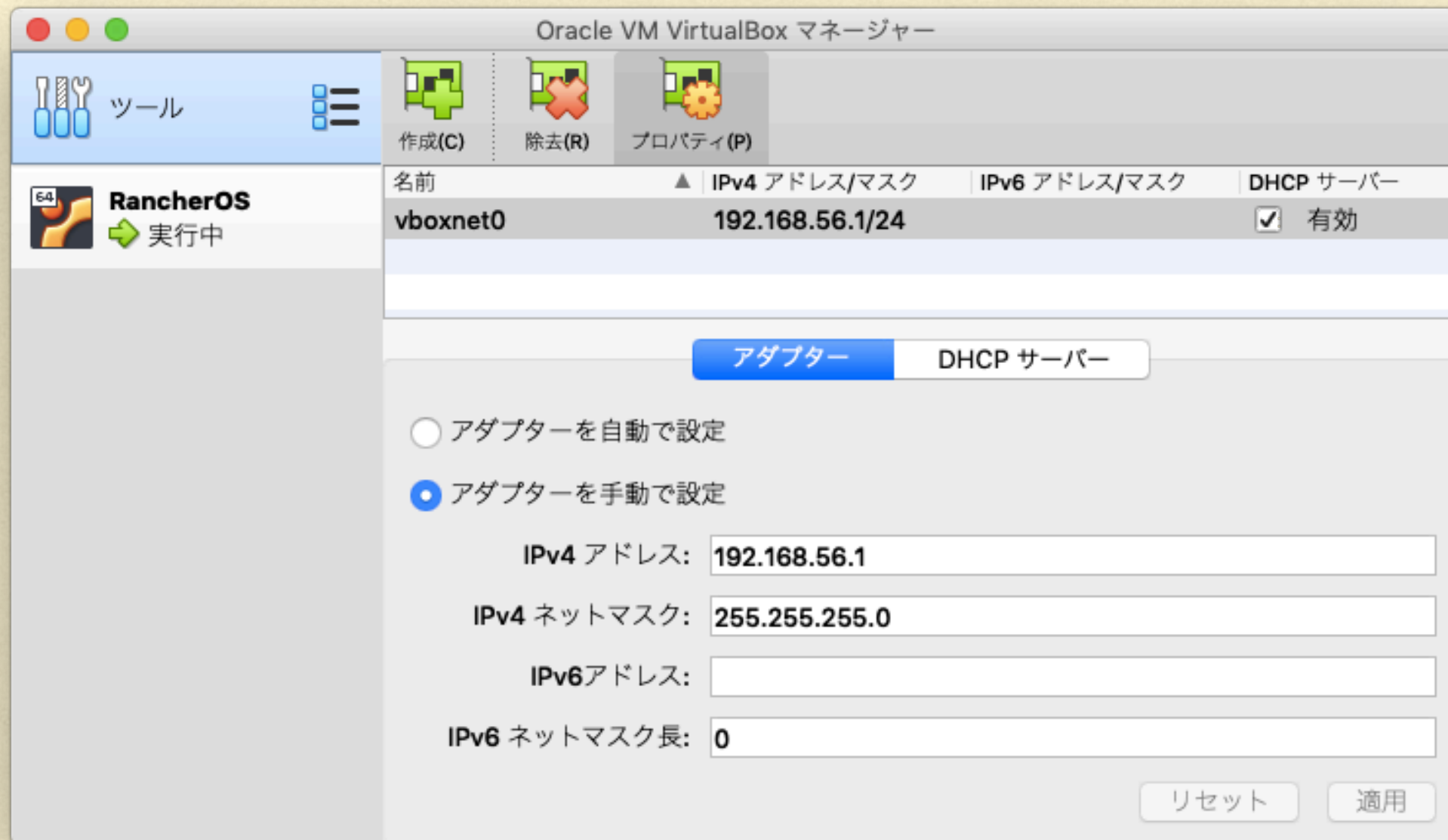
The screenshot shows the Rancher website with the following elements:

- Header:** RANCHER logo, navigation links: WHY RANCHER?, PRODUCTS, CUSTOMER.
- Hero Section:** "RancherOS" title, subtitle "A lightweight, secure Linux distribution, built from containers to run containers well.", and a blue "DOWNLOAD RANCHEROS" button.
- Download List:** A table of available download files.

File Name	Size
rancheros-raspberry-pi04.zip	10.4 MB
rancheros-vmware-autoformat.iso	143 MB
rancheros-vmware.iso	143 MB
rancheros-vmware.vmdk	134 MB
rancheros.ipxe	416 Bytes
rancheros.iso	127 MB
rootfs.tar.gz	83.3 MB
vmlinuz	5.29 MB
Source code (zip)	
Source code (tar.gz)	

VirtualBox上での設定

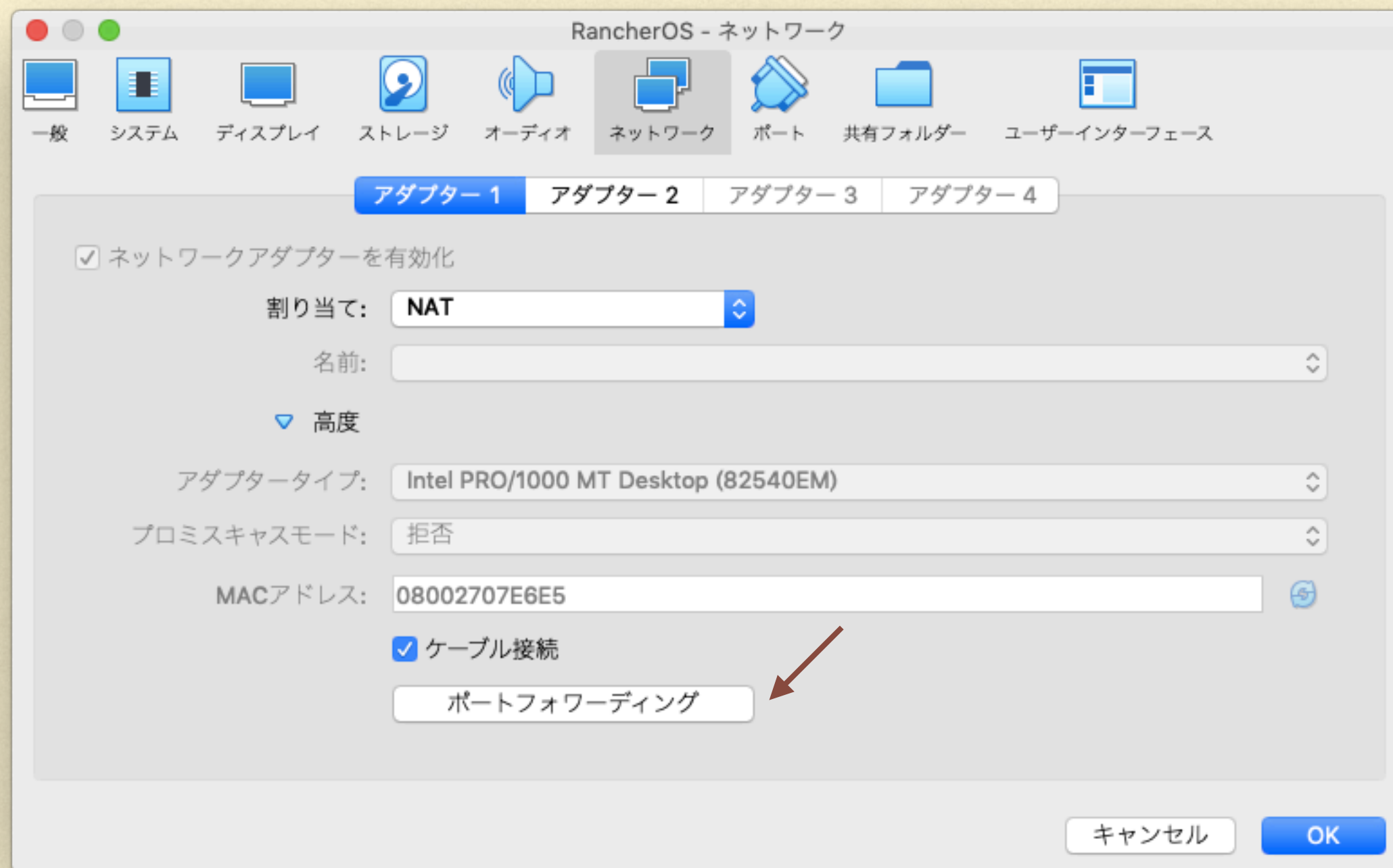
- 仮想マシンを作る前に、ネットワークの設定をする。



仮想マシンの設定

- VDIで、20GByte確保
 - MEMは、2048MByte
 - rancheros.isoをマウント
 - ネットワークのアダプターを、”NAT+ホストオンリーアダプター”にする。
-

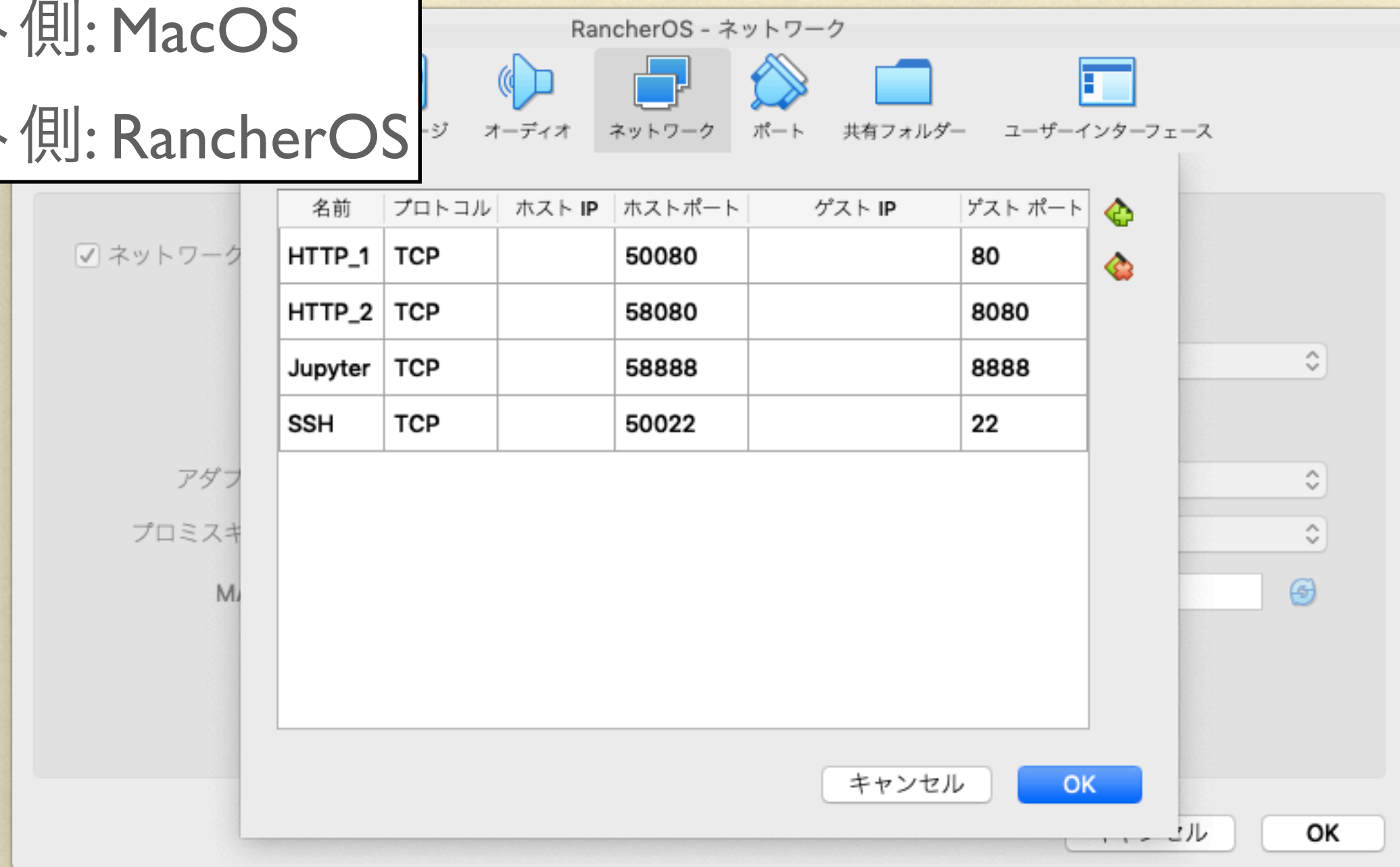
ネットワークの設定(eth0)



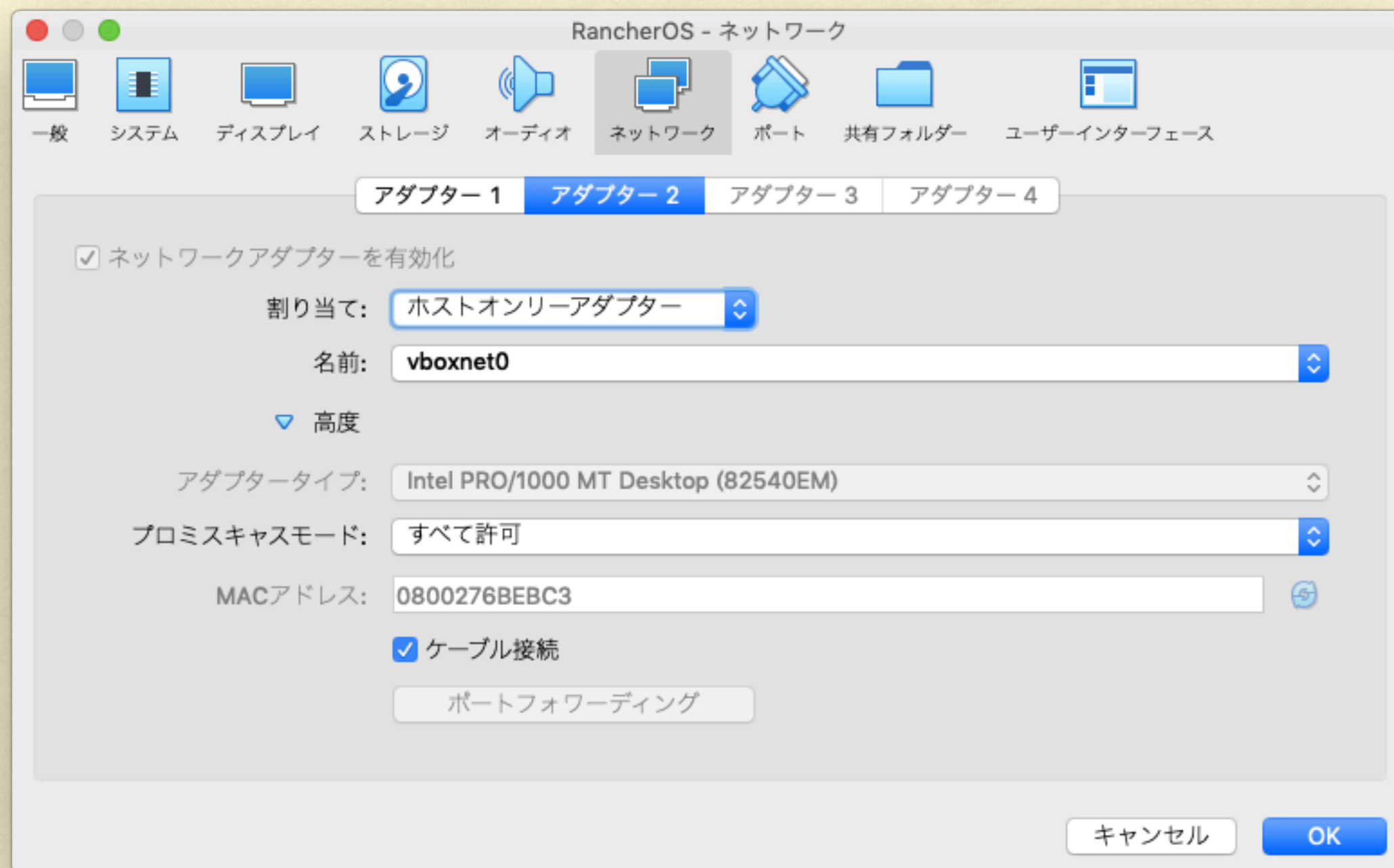
ポートフォワード

ホスト側: MacOS

ゲスト側: RancherOS



ネットワークの設定(eth1)




RancherOSのインストール

- 仮想マシンを起動し、インストールする前に、cloud-config.ymlファイルを作成。内容は、次ページに記載。

cloud-config.yml

```
#cloud-config.yml
rancher:
  console: ubuntu
  network:
    interfaces:
      eth0:
        #address: 192.168.88.1/24
        #gateway: 255.255.255.0
        mtu: 1500
        dhcp: true
```



cloud-config.ymlの内容確認 & インストール

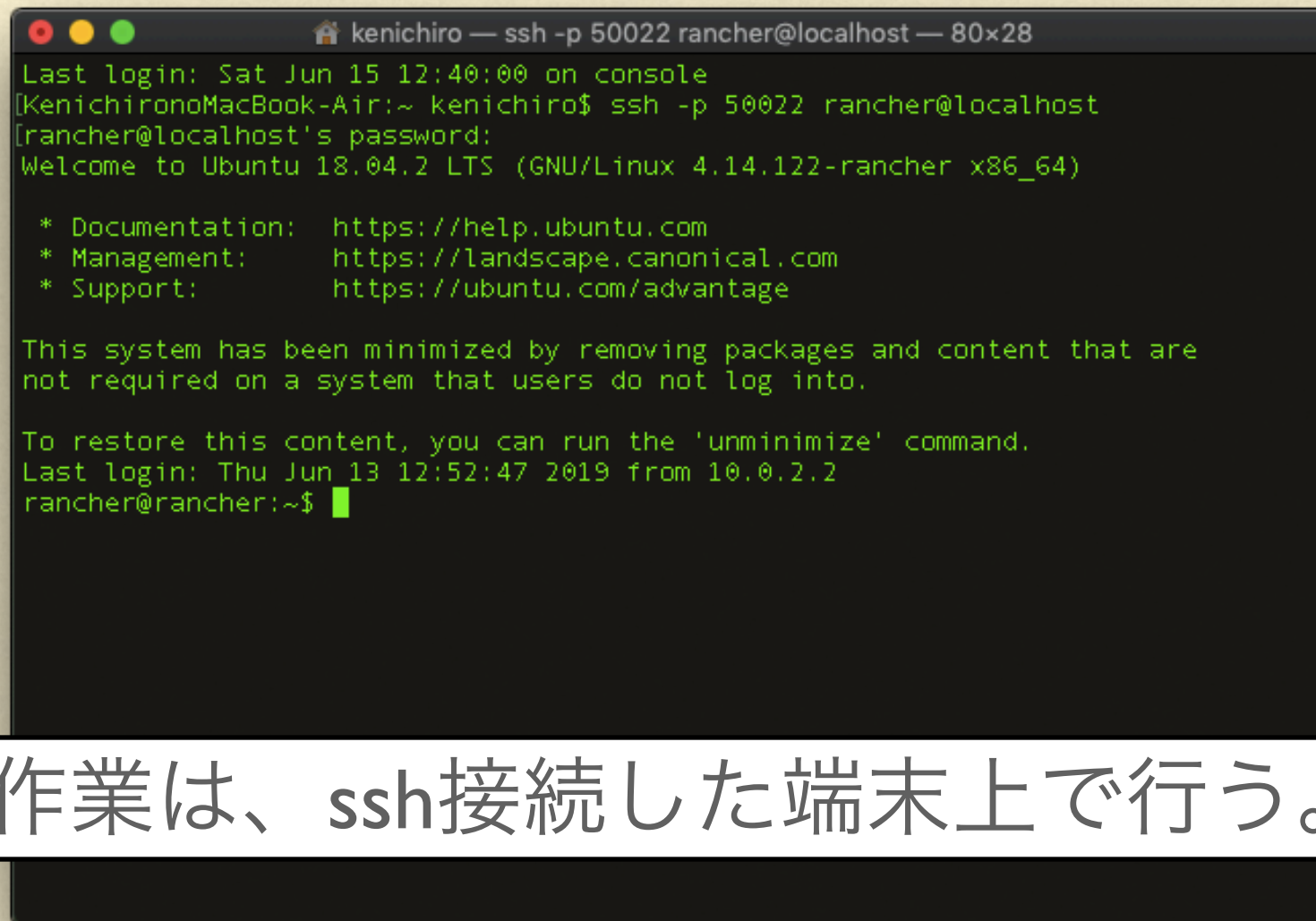
```
$ sudo ros config validate -i cloud-config.yml  
$ sudo ros install -c cloud-config.yml -d /dev/sda --append "rancher.password=rancher"
```

- ユーザ名: rancher
- パスワード: rancher で、RancherOSをインストール

インストール後の接続確認

- bashなどの端末から、sshコマンドで、RancherOSと接続

```
$ ssh -p 50022 rancher@localhost
```

A terminal window titled 'kenichiro — ssh -p 50022 rancher@localhost — 80x28'. The output shows a successful SSH connection to a RancherOS instance. The prompt changes from 'kenichiro\$' to 'rancher@rancher:~\$'. The terminal text includes: 'Last login: Sat Jun 15 12:40:00 on console', '[KenichironoMacBook-Air:~ kenichiro\$ ssh -p 50022 rancher@localhost]', '[rancher@localhost's password:', 'Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.14.122-rancher x86_64)', a list of links for documentation, management, and support, a message about system minimization, and instructions to run 'unminimize' to restore content. The final prompt is 'rancher@rancher:~\$' with a green cursor.

```
kenichiro — ssh -p 50022 rancher@localhost — 80x28
Last login: Sat Jun 15 12:40:00 on console
[KenichironoMacBook-Air:~ kenichiro$ ssh -p 50022 rancher@localhost
[rancher@localhost's password:
Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.14.122-rancher x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Thu Jun 13 12:52:47 2019 from 10.0.2.2
rancher@rancher:~$
```

これ以降の作業は、ssh接続した端末上で行う。

RancherOS側のコンソール切り替え

```
$ sudo ros console list  
$ sudo ros console switch ubuntu
```

- 必要に応じて、使いやすいコンソールに切り替える。

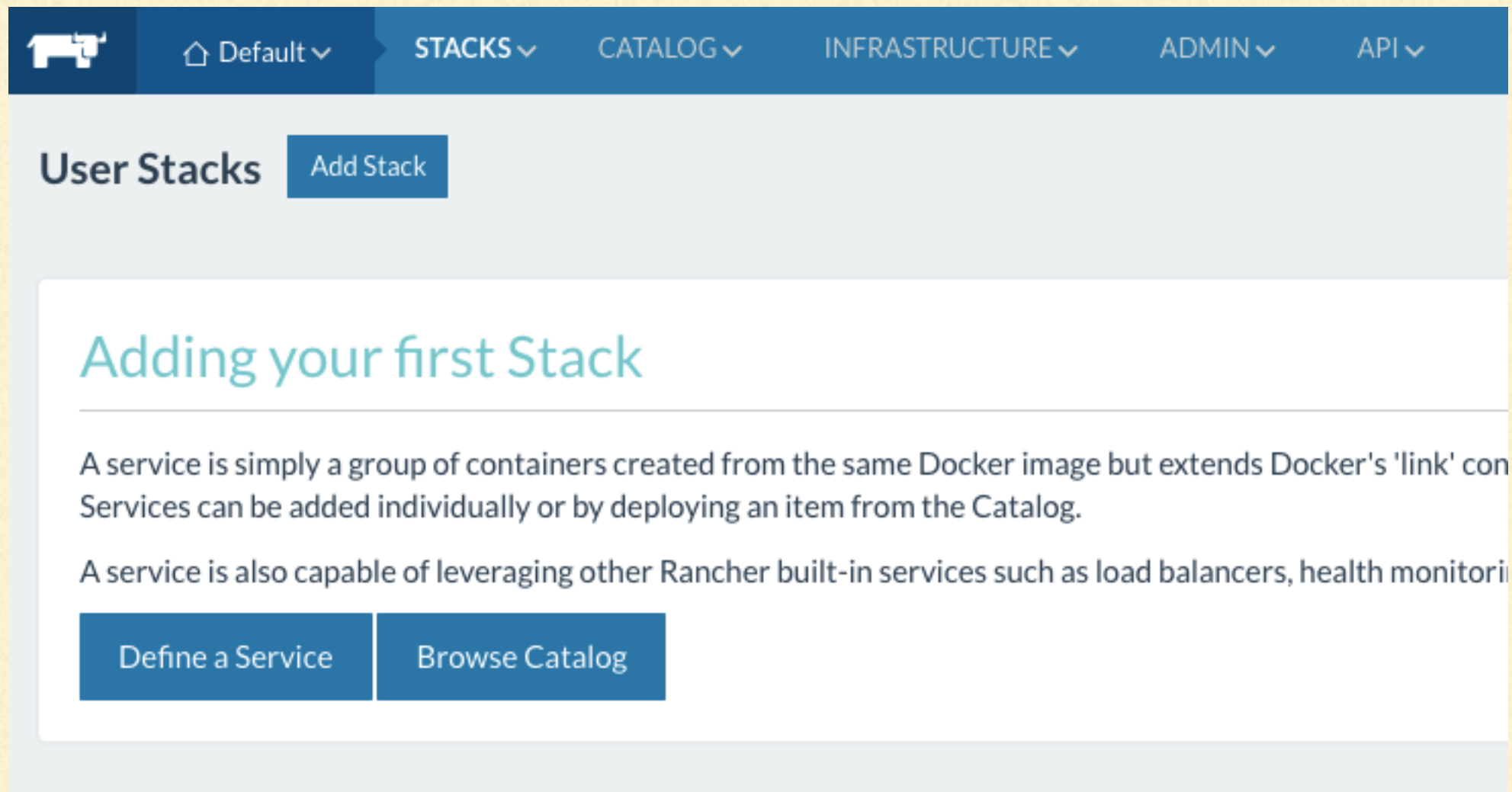
管理用のコンテナ(rancher)をインストール

```
$ docker run -d --restart=always -p 8080:8080 rancher/server
```

- しばらく待った後、Webブラウザ上で、
"http://192.168.56.1:58080/" にアクセス。

アクセス先のIPアドレスは、設定に応じて変更すること。

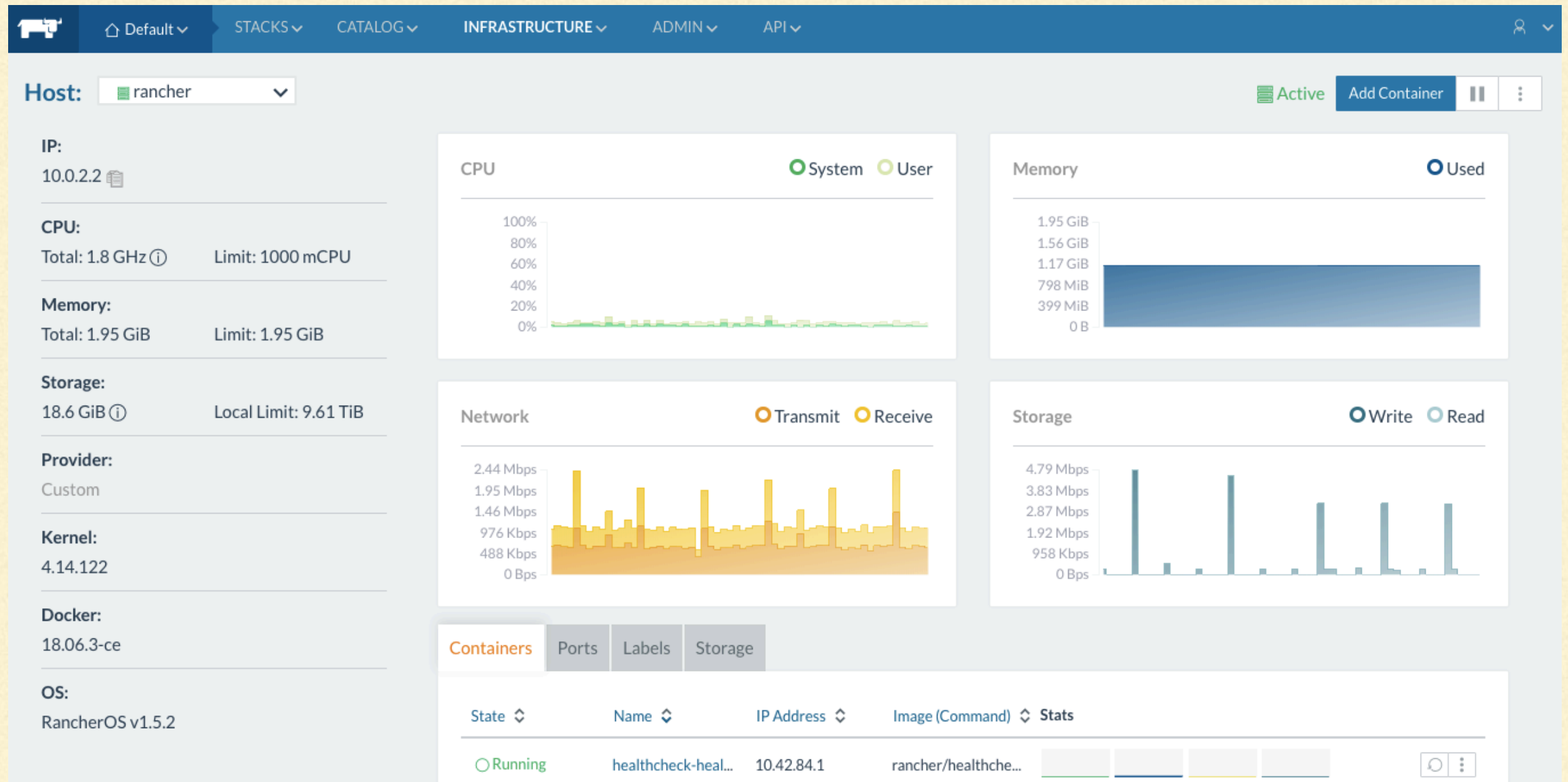
管理画面



rancher側の設定

- ADMIN -> Settingsで、サイトのアドレス設定をする。
- INFRASTRUCTURE -> Add Hostで、指示に従ってhostを追加する。

動作画面のキャプチャー



Jupyter Notebookコンテナのインストール&起動

```
$ docker pull jupyter/datascience-notebook  
$ docker images jupyter/datascience-notebook  
$ docker run -d --name notebook -p 8888:8888 jupyter/datascience-notebook
```

- Jupyter Notebookのコンテナが、容量が大きいいため、ダウンロードが完了するまでしばらく待つ。

Jupyter Notebookの起動画面

- “<http://192.168.56.1:58888/>”にアクセスし、コンテナ上のJupyter Notebookを開く。初回起動の場合は、Tokenを聞かれてくるので、設定を行う。

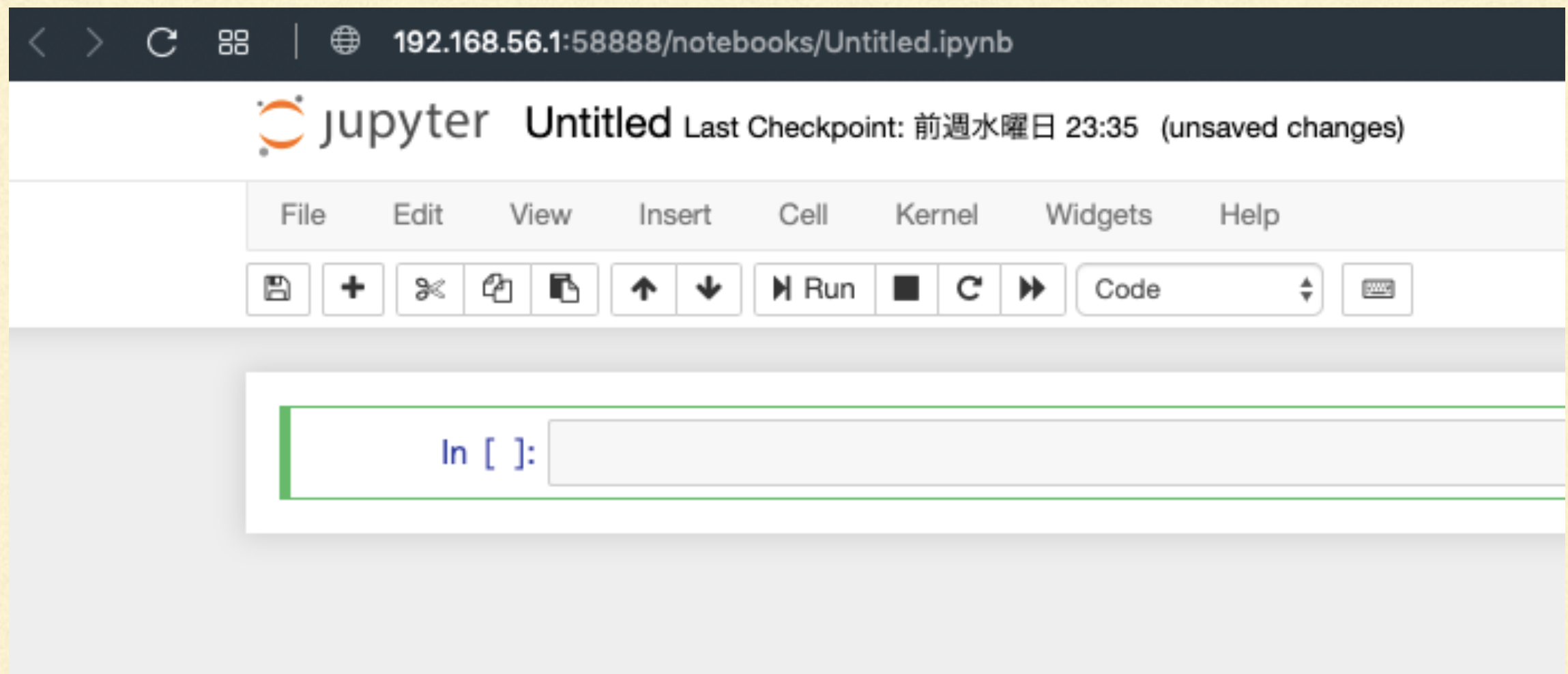
Jupyter Notebookの初期設定

- 初回の起動だと、Tokenを聞かれてくるので、Jupyter Notebookコンテナ内のbashにアクセスし、Tokenを拾ってくる。

```
$ docker exec -it notebook /bin/bash
jovyan@d9fd6a558d96:~$ jupyter notebook list
Currently running servers:
http://
0.0.0.0:8888/?token=59fc2938f4fdb740177e71982dd0517264ae3b6688b6a328 :
: /home/jovyan
```

■

Jupyter Notebookの起動画面



とりあえずは、動いたので...

- ベンチマークしてみました。流体解析の計算とのこと。
- <https://qiita.com/buffetonset/items/49f52a3dc0cdad6011a0>

@buffetonset 2018年06月18日に更新

姫野ベンチマーク(himenoBMT)をPythonで実行し、[numpy, numba]で高速化。

Python ベンチマーク numpy Numba 差分法

姫野ベンチマークをPythonで実行し、[numpy, numba]で高速化。

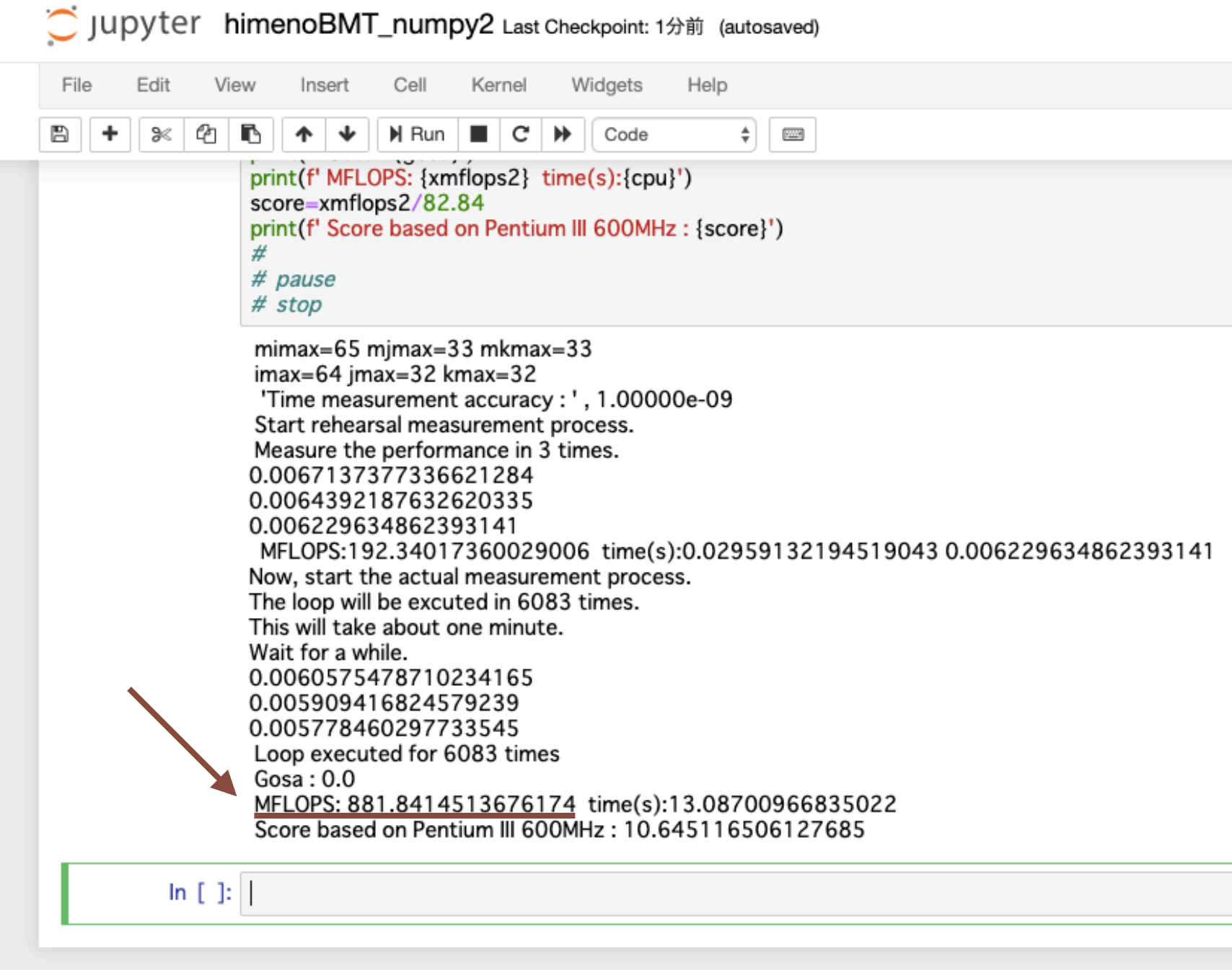
姫野ベンチは、3次元ポアソン方程式をヤコビ法で解く反復処理の速度を図るベンチマークです。

本家のホームページは→[こちら](#) Fortran90とCの公式版があります。

20数年前からある古いベンチマークですが、流体解析に必要なメモリ律速のベンチマークに向いています。

Pythonを始めるにあたり、どの程度の速度がでるか興味があって始めました。

動作デモ



jupyter himenoBMT_numpy2 Last Checkpoint: 1分前 (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run

```
print(f' MFLOPS: {xmflops2} time(s):{cpu}')
score=xmflops2/82.84
print(f' Score based on Pentium III 600MHz : {score}')
```


pause
stop

mimax=65 mjmax=33 mkmax=33
imax=64 jmax=32 kmax=32
'Time measurement accuracy : ', 1.00000e-09
Start rehearsal measurement process.
Measure the performance in 3 times.
0.0067137377336621284
0.0064392187632620335
0.006229634862393141
MFLOPS:192.34017360029006 time(s):0.02959132194519043 0.006229634862393141
Now, start the actual measurement process.
The loop will be excuted in 6083 times.
This will take about one minute.
Wait for a while.
0.0060575478710234165
0.005909416824579239
0.005778460297733545
Loop executed for 6083 times
Gosa : 0.0
MFLOPS: 881.8414513676174 time(s):13.08700966835022
Score based on Pentium III 600MHz : 10.645116506127685

In []:

ベンチマーク結果

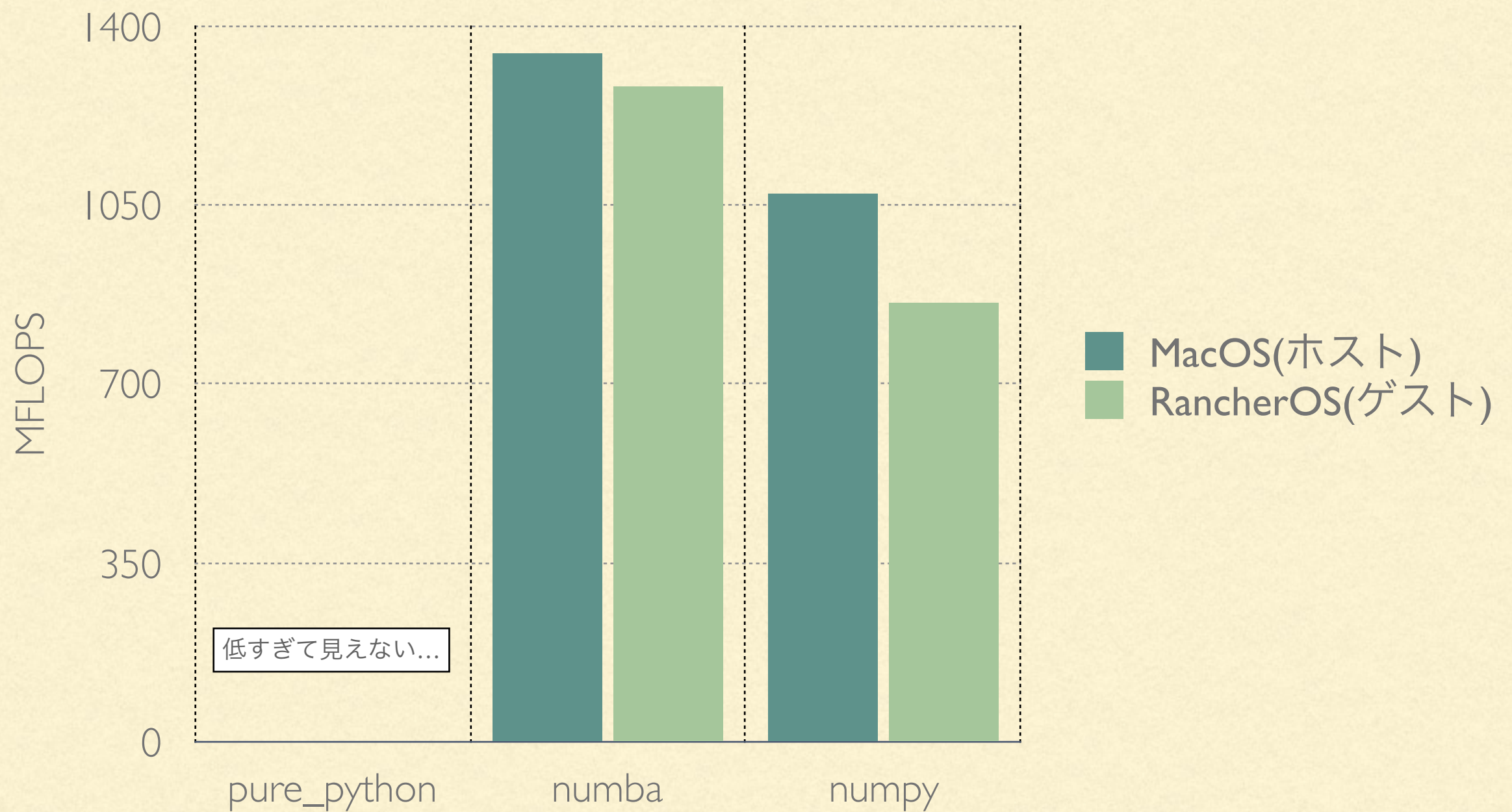
- サイズ: “xs”
- ベンチマーク結果(MFLOPSの値)は、下表のとおり。

単位[MFLOPS]	MacOS(ホスト)	RancherOS(ゲスト)
pure_python	1.94	0.96
numba	1347.43	1282.28
numpy	1073.28	859.14

VirtualBox上でも、意外とパフォーマンスが良い。

numbaの成績が良いのは、JITコンパイラを使っているため？

ベンチマーク結果



おわり
