

RaspberryPi活用8①

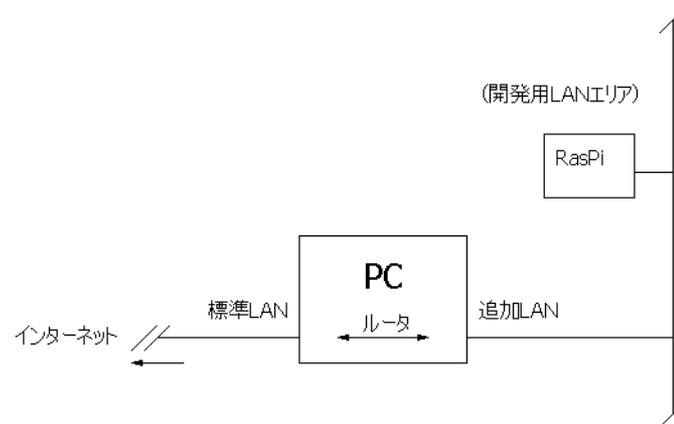
RaspberryPiOSの起動

- ・支援ツールとしてWindowsマシン上にLinuxマシンをVMwareWorkstationを介したubuntuを用いる。(「付録」のVMwareWorkstationとubuntuのインストール参照)
- ・RaspberryPiに使用するOSはRaspberryPiOS(Raspbianといった表現は見つからなくなった)とする。
- ・RaspberryPiハードウェアはRaspberryPi4 ModelB
(電池駆動には不利だか、より省電力のRaspberryPi3以前との違いについては都度注釈を入れる)



RaspberryPiOSの起動

ここではRaspberryPiはPCに接続されたLAN(グローバル側)とは別の以下のような追加LAN環境(ローカル側)で操作する。



RaspberryPiOSの起動

<https://www.raspberrypi.org/software/operating-systems/>

からRaspberryPiImagerをダウンロードする。



Hardware

Software

Books & magazines

Learn

Teach

About us

Operating system images

Many operating systems are available for Raspberry Pi, including Raspberry Pi OS, our official supported operating system, and operating systems from other organisations.

[Raspberry Pi Imager](#) is the quick and easy way to install an operating system to a microSD card ready to use with your Raspberry Pi. Alternatively, choose from the operating systems below, available to download and install manually.

Download:

[Raspberry Pi OS \(32-bit\)](#)

[Raspberry Pi Desktop](#)

[Third-Party operating systems](#)

[Raspberry Pi Imager](#) is the
system to a microSD card

RaspberryPiOSの起動

Windowsマシンの場合、Windows版をダウンロード

Install Raspberry Pi OS using Raspberry Pi Imager

Raspberry Pi Imager is the quick and easy way to install Raspberry Pi OS and other operating systems to a microSD card, ready to use with your Raspberry Pi. [Watch our 45-second video](#) to learn how to install an operating system using Raspberry Pi Imager.

Download and install Raspberry Pi Imager to a computer with an SD card reader. Put the SD card you'll use with your Raspberry Pi into the reader and run Raspberry Pi Imager.

[Download for Windows](#)

[Download for macOS](#)

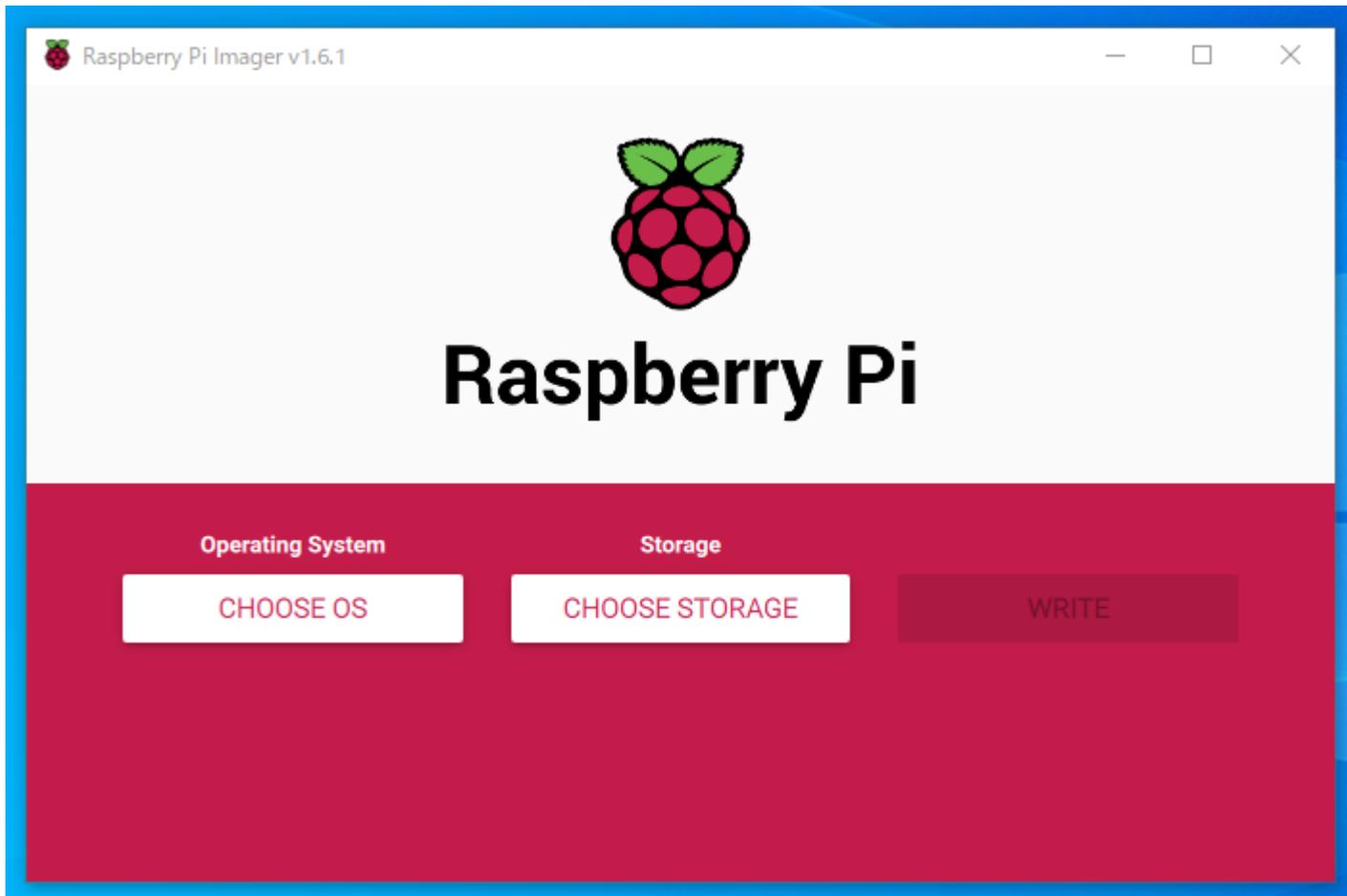
[Download for Ubuntu for x86](#)



[Download for Windows](#)

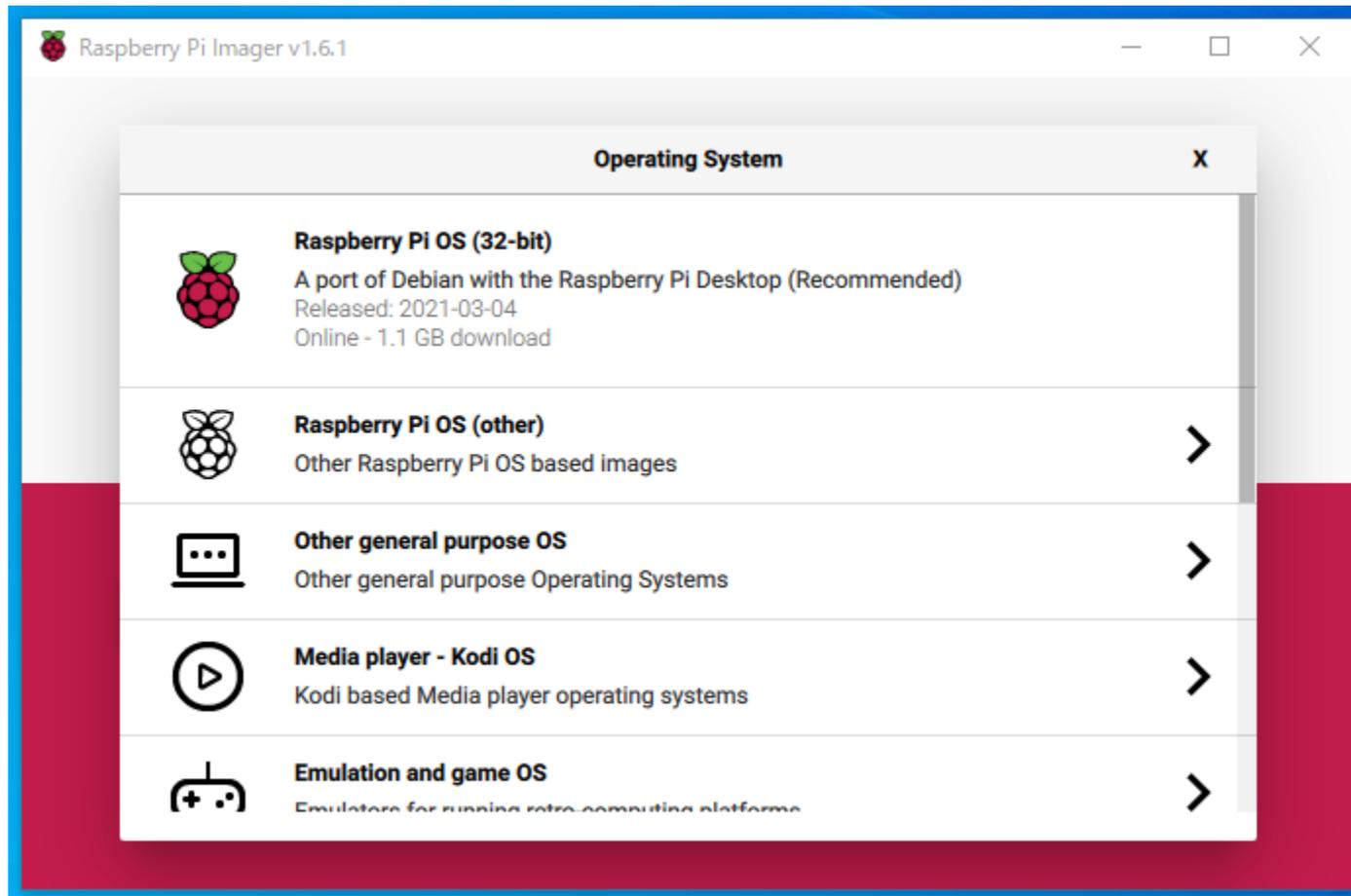
RaspberryPiOSの起動

インストール後、Raspberry Pi Imagerを起動する。



RaspberryPiOSの起動

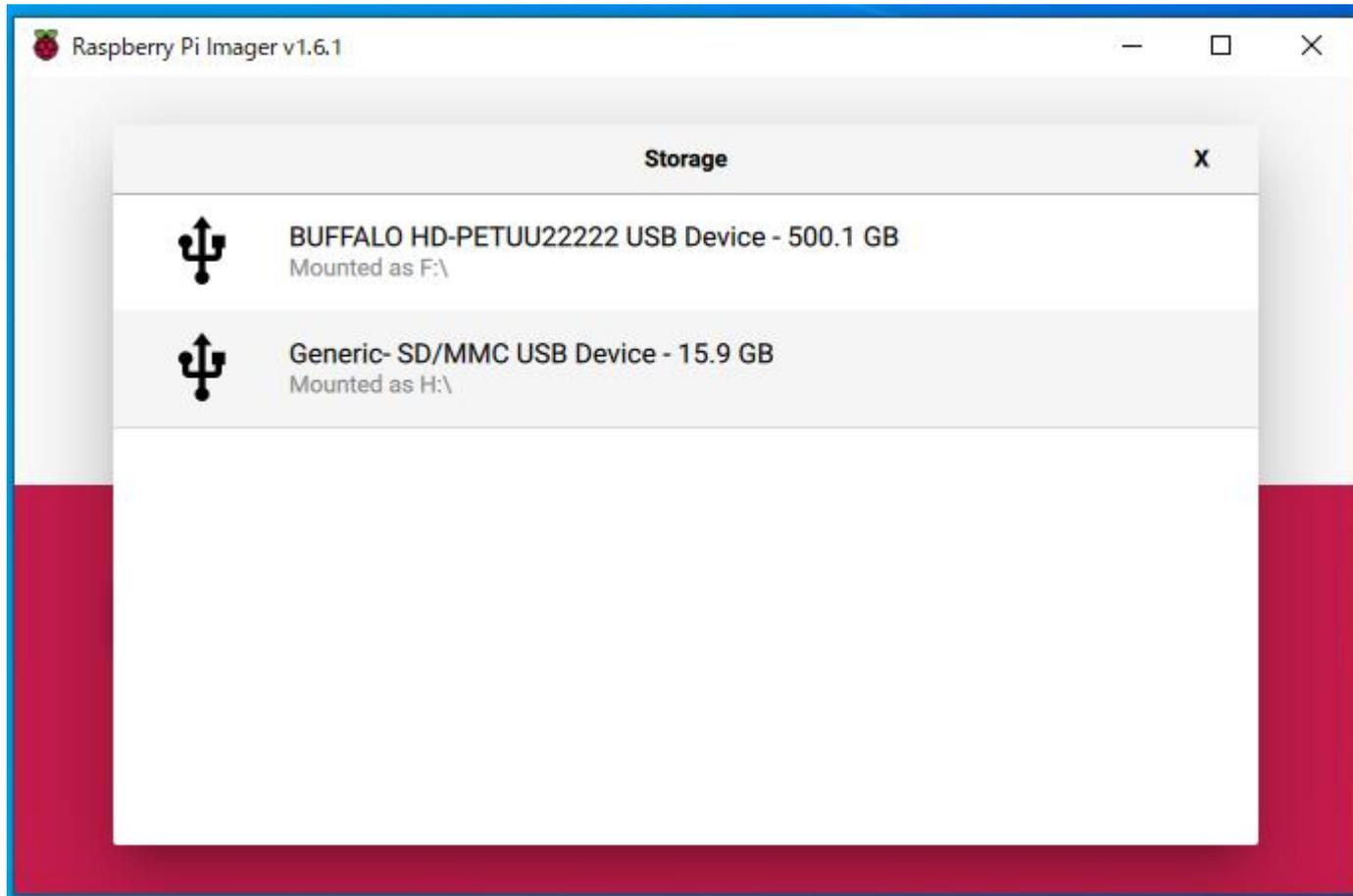
CHOOSE OSでインストールするOSをリストから選ぶ。



RaspberryPiOSの起動

CHOOSE STRAGEでインストールするSDカードを選ぶ。

インターネットに直接つながった環境なら起動SDの完了を待つだけでよい。



RaspberryPiOSの起動

プロキシ環境の場合、<https://www.raspberrypi.org/software/operating-systems/>から必要なOSのイメージファイルをダウンロードする。

ここでは「Raspberry Pi OS with desktop and recommended software」をダウンロードしておく。



2021-03-04-raspios-buster-armhf-full.zip

Raspberry Pi OS

Compatible with:

[All Raspberry Pi models](#)



Raspberry Pi OS with desktop and recommended software

Release date: March 4th 2021

Kernel version: 5.10

Size: 2,868MB

[Show SHA256 file integrity hash:](#)

[Release notes](#)

[Download](#)

[Download torrent](#)

Raspberry Pi OS with desktop

Release date: March 4th 2021

Kernel version: 5.10

Size: 1,175MB

[Show SHA256 file integrity hash:](#)

[Release notes](#)

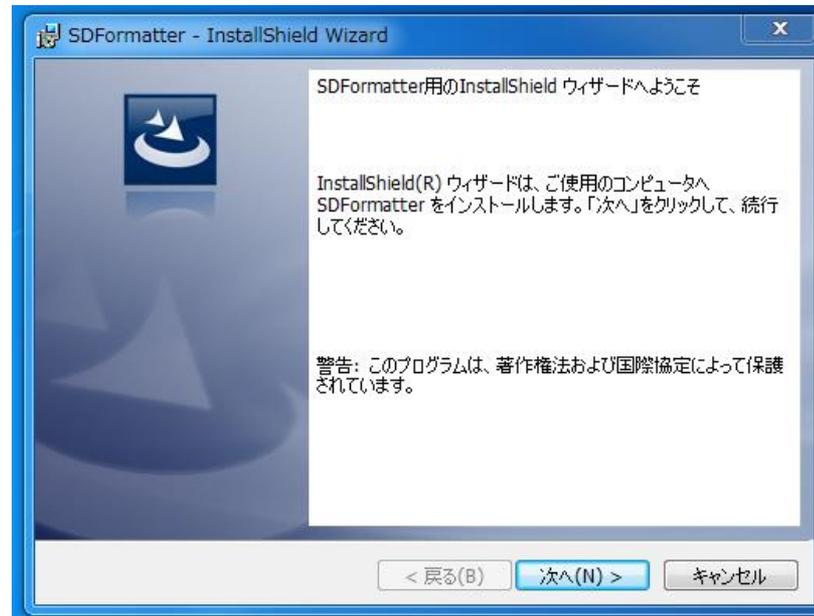
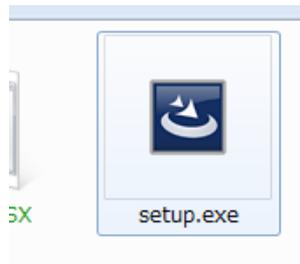
[Download](#)

[Download torrent](#)

RaspberryPiOSの起動

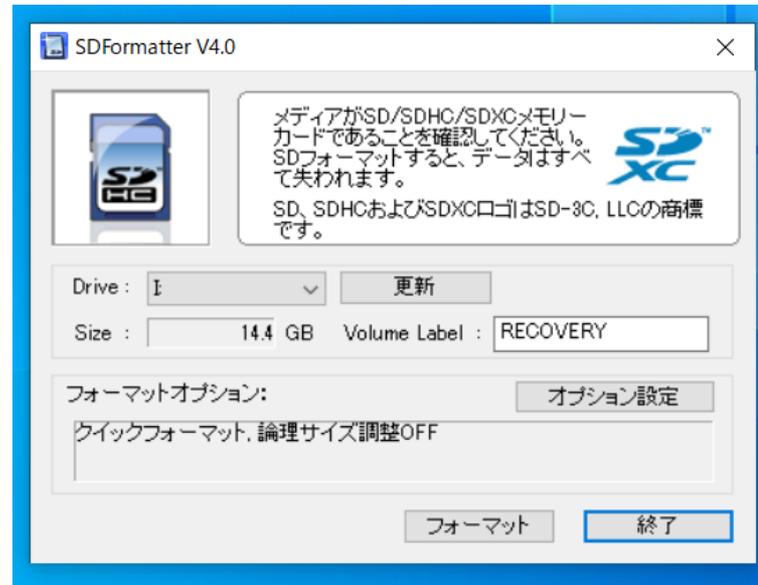
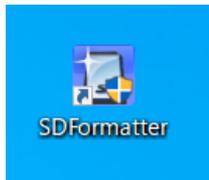
次に、購入後のSDカードまたは手持ちのSDカードをSDフォーマッタでフォーマットしカードを準備する。

フリーのSDフォーマッタについてはSDFormatter4exe.zipをダウンロードし、展開後、setupを実行する。(配布データに収録)



RaspberryPiOSの起動

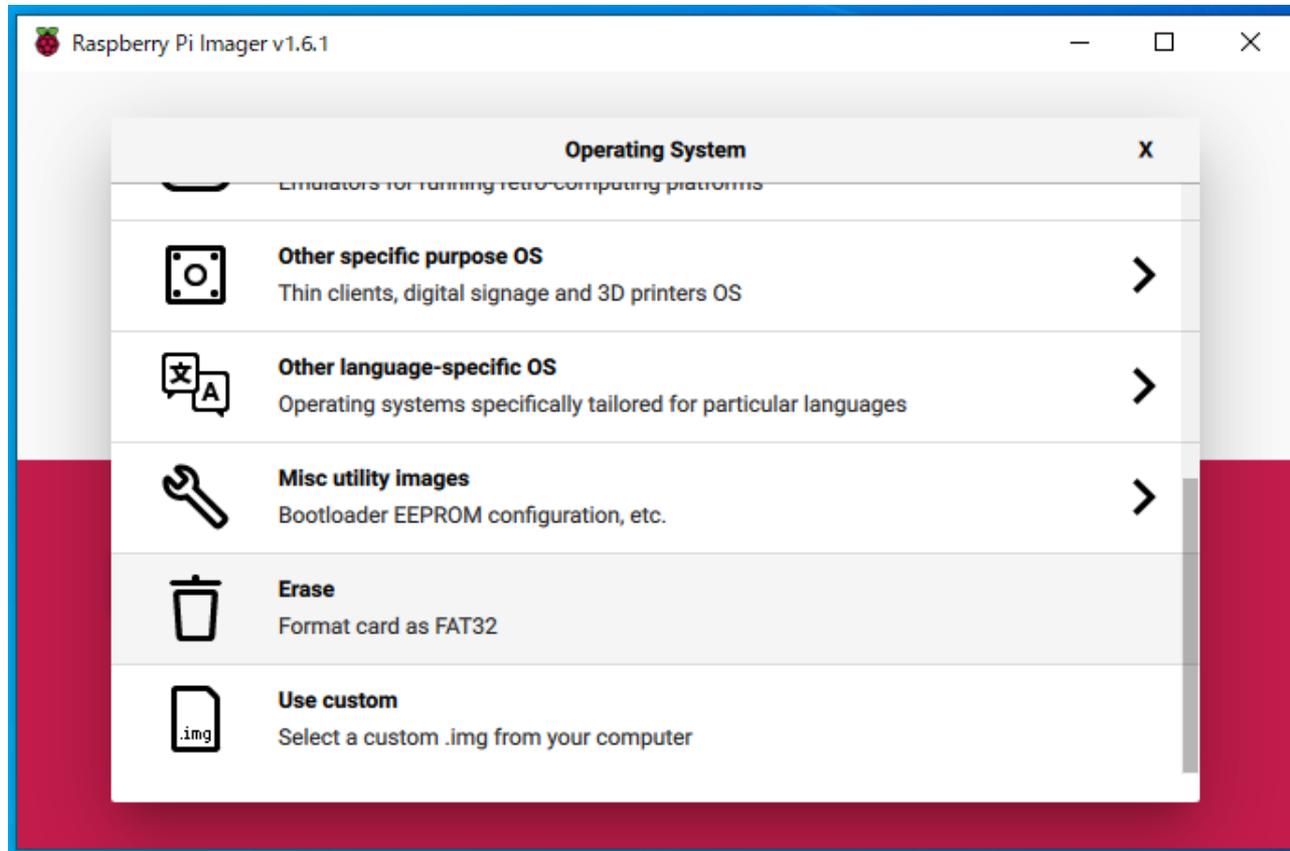
SDFormatterを起動し、フォーマッタでSDカードをフォーマットする。ここではSDカードが I:ドライブに当てられている。「フォーマット」を実行する。



~~このフォーマットでのSDカードはLinuxは認識しない。ddコマンドを使うなど、認識させたい場合はWindowsアプリでなく、dosアプリのDiskpartによる createparttion primary → active → フォーマット 処理が必要。~~
認識しなかったのはVolumeLabelがRECOVERYであったため。
他の名前にするとLinuxでも認識する。(原因不明)

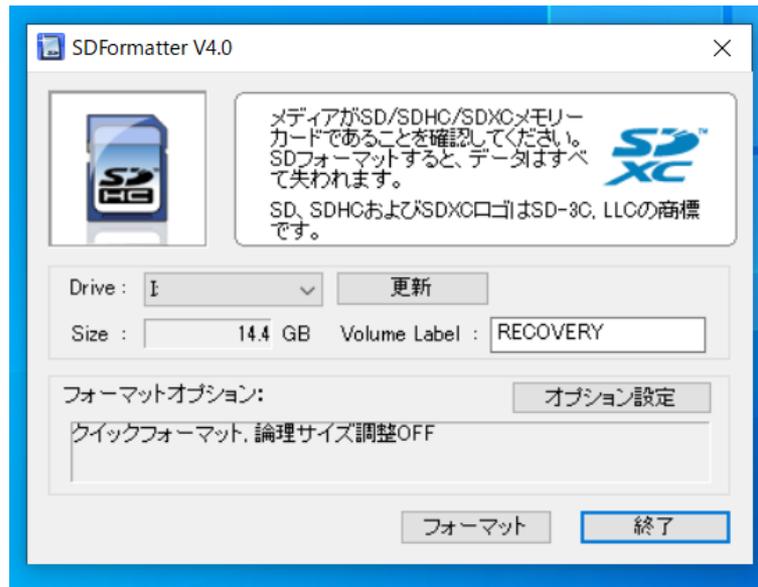
RaspberryPiOSの起動

Raspberry Pi Imagerを起動し、CHOOSE OS→「Erase」を選択するとフォーマットとEraseを実行する。これはプロキシ環境に関係なくSDカードの初期化処理として利用できる。エラーのときはSDカードを一旦取り外し、再接続、再認識させることでOKになることがある。



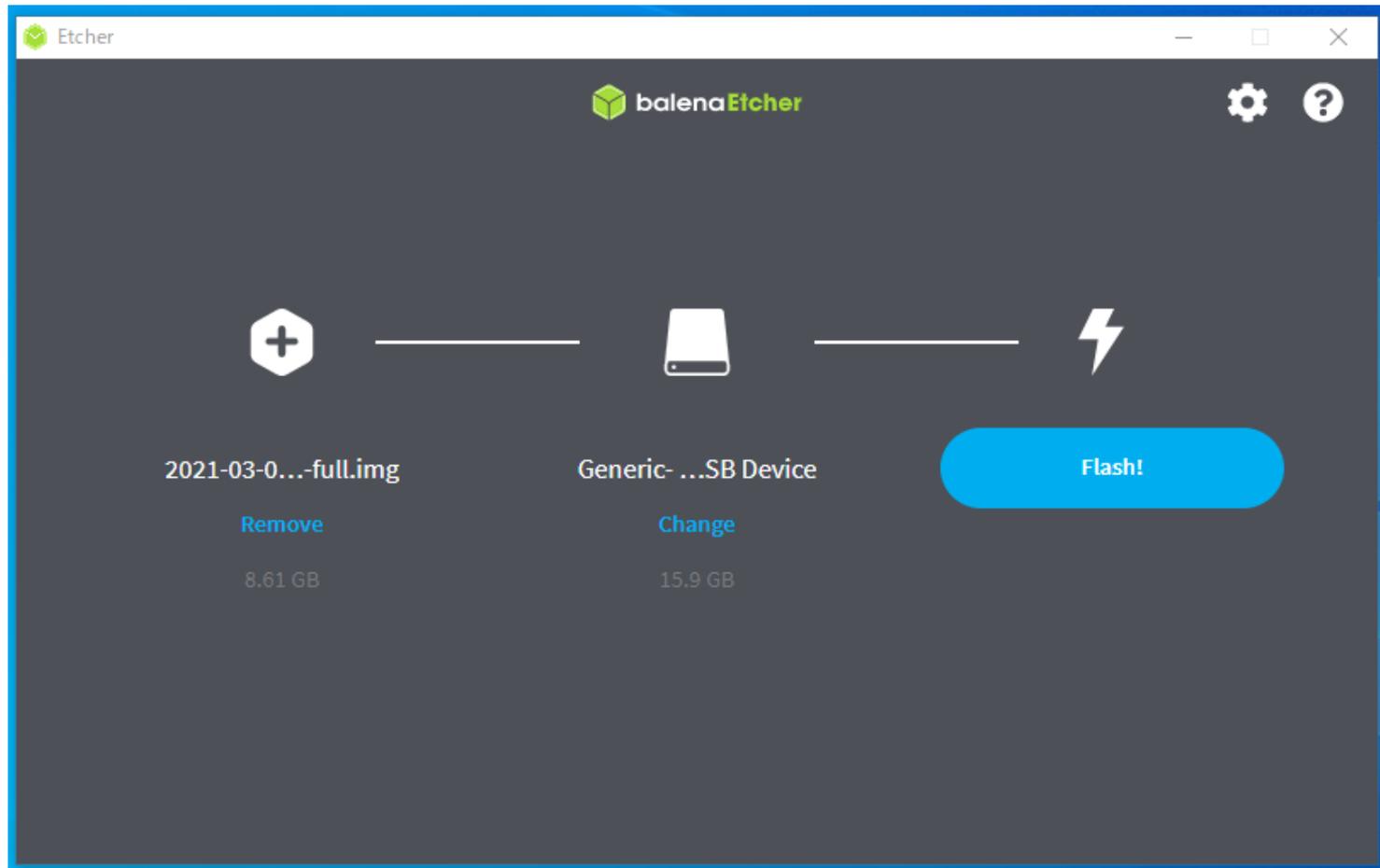
RaspberryPiOSの起動

あるいはRaspberry Pi Imagerを使用しなくても、SDフォーマッタでフォーマットすることでもErase可能。



RaspberryPiOSの起動

フリーのbalenaEtcher(配布データに収録)を起動し、Flash from file に解凍したイメージファイルを指定し、Select targetにSDカード選択を確認して、Flash！を実行する。



RaspberryPiOSの起動

起動用SDカードは検証したところ以下のような使用制限があるようだ。

- RaspberryPi4で起動させた後、SDカードはRaspberryPi3以前のSDカードには使用できない。
- RaspberryPi3で起動させた後のSDカードはRaspberryPi Zero W でも使用できる。おそらくBCM2835チップのモデルには使用できるだろう。

大きく異なるのは

- これまでのRaspbianはパーティションが7つ作成される。
- RaspPiOSはパーティションが2つ作成される。

[Kernel building - Raspberry Pi Documentation](#)

If it's a NOOBS card, you should see something like this:

```
sdb
sdb1
sdb2
sdb5
sdb6
sdb7
```

```
sdb
sdb1
sdb2
```

with `sdb1` being the FAT (boot) partition, and `sdb2` being the ext4 filesystem (root) partition.

RaspberryPiOSの起動

再起動後は、バージョンを重ねるうち初期画面で初期設定ができるようになった。



Set Country はJapan,Japanese,Tokyo を選択



RaspberryPiOSの起動

Change Passwordここでは変更せず。記述にあるようにデフォルトのアカウント「pi」、パスワード「raspberrry」となる。

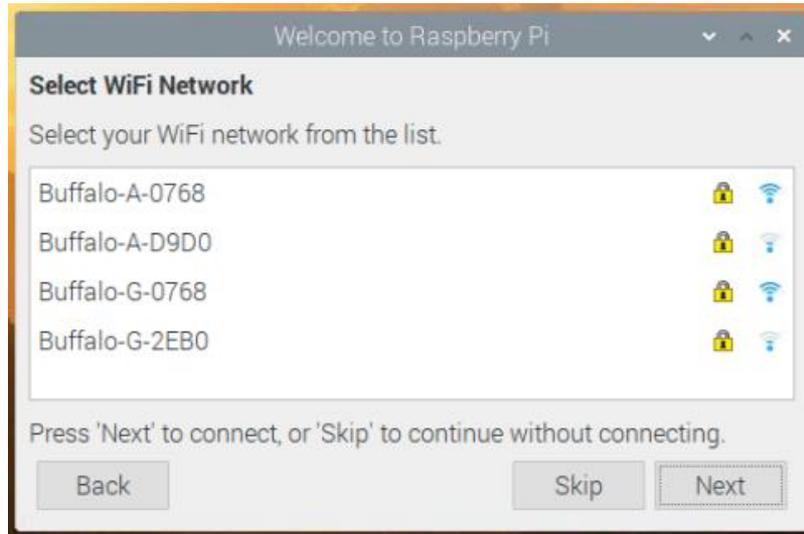


Set Up Screen はそのまま「Next」



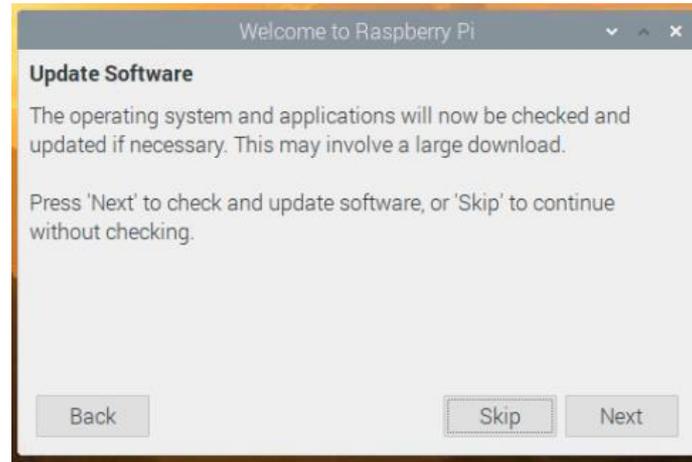
RaspberryPiOSの起動

今回はWiFiは使用しないので「Skip」を選択



RaspberryPiOSの起動

Update Software はプロキシ環境ではこの段階で接続できないので「Skip」を選択。



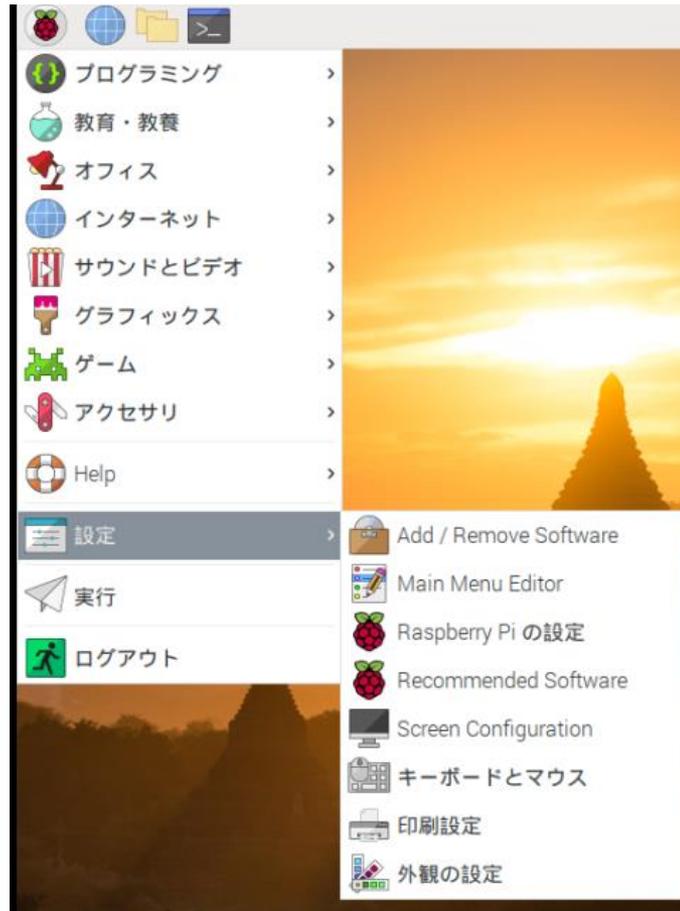
Setup Complete は「Done」を選択。



RaspberryPiOSの起動

Reboot後、日本語表記に変わる。

[設定] → [Raspberry Piの設定] を選択



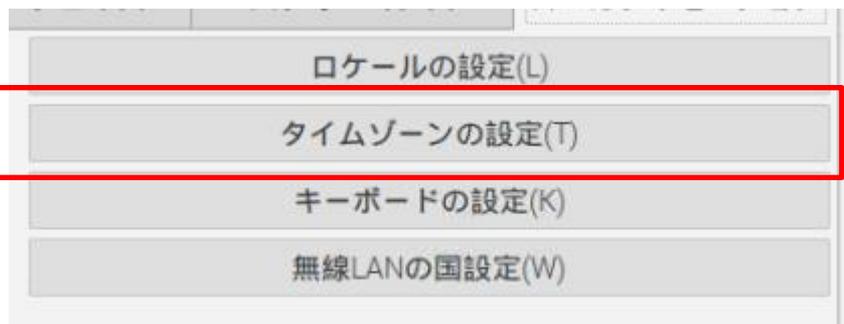
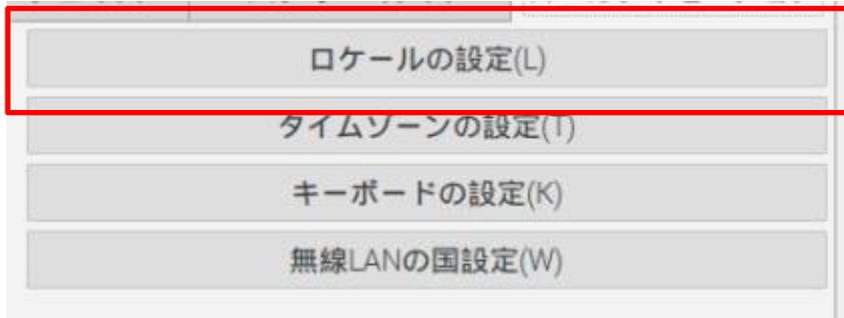
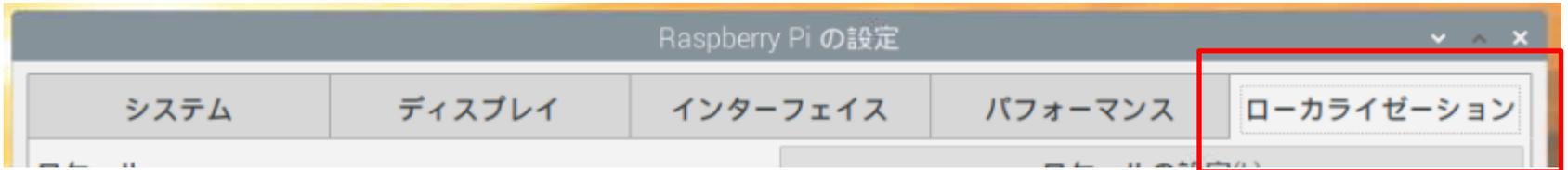
RaspberryPiOSの起動

ここでは[インターフェイス] から「SSH」、「VNC」を有効にする。



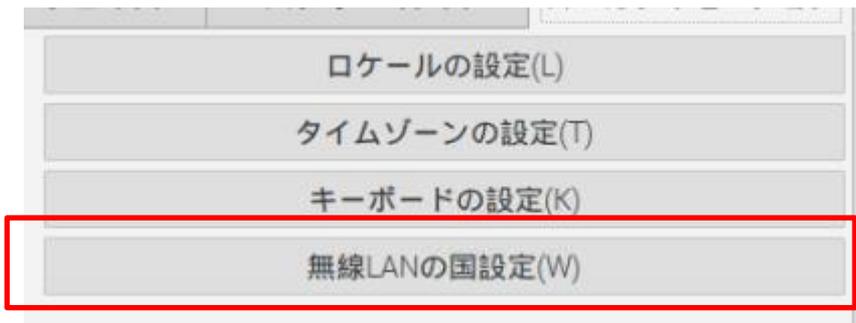
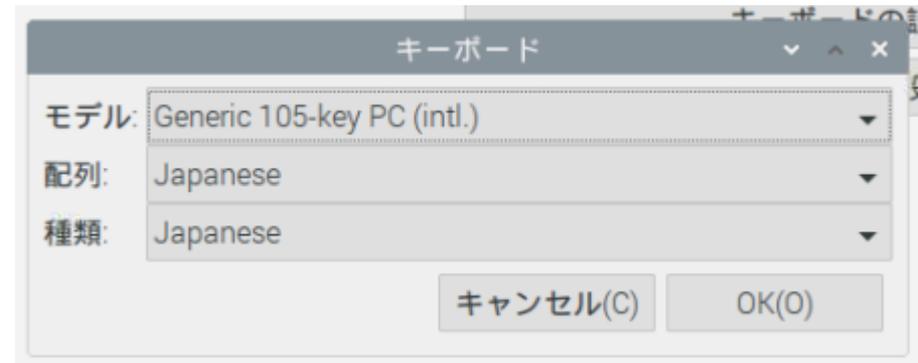
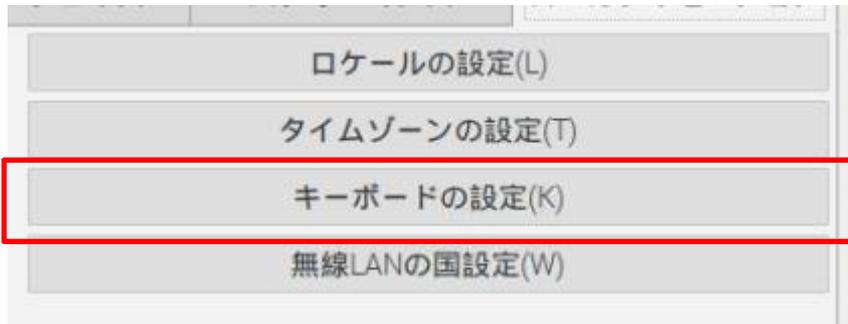
RaspberryPiOSの起動

「ローカライゼーション」の設定



RaspberryPiOSの起動

「ローカライゼーション」の設定



RaspberryPiOSの起動

ネットワークは有線とし、ここでは192.168.137.55/24に設定する。
またIPV4のみの手動アドレスで使用する。

上部「↑ ↓」を右クリック。

「Wireless & Wired Network Settings」を選択。ルータとDNSはルータアドレスに設定する。



RaspberryPiOSの起動

設定内容は /etc/dhcpd.conf に反映されている。

```
pi@raspberrypi:~ $ ls -l /etc/dhcpd.conf
-rw-rw-r-- 1 root netdev 1929  3月  5 09:43 /etc/dhcpd.conf
pi@raspberrypi:~ $ sudo nano /etc/dhcpd.conf
```

行末に以下の部分が追記されている。

```
#interface eth0
#fallback static_eth0

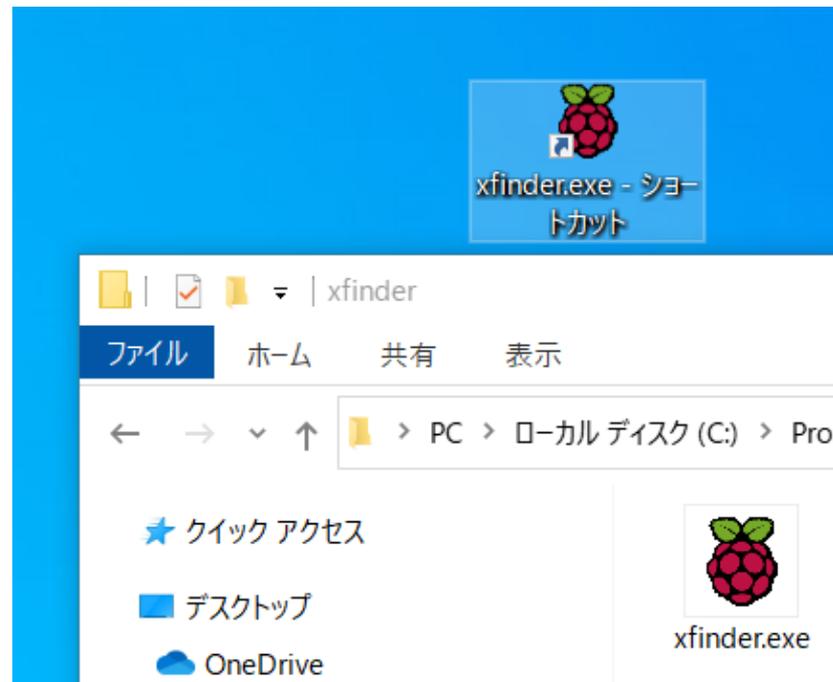
interface eth0
static ip_address=192.168.137.55/24
static routers=192.168.137.1
static domain_name_servers=192.168.137.1
static domain_search=
noipv6
```

RaspberryPiOSの起動

SSHが使える状態になっているので、Teratermで制御可能で、このときXfinderが使用できると便利である。

xfinder.exeはネットからダウンロードで入手できる。(配布データに収録)

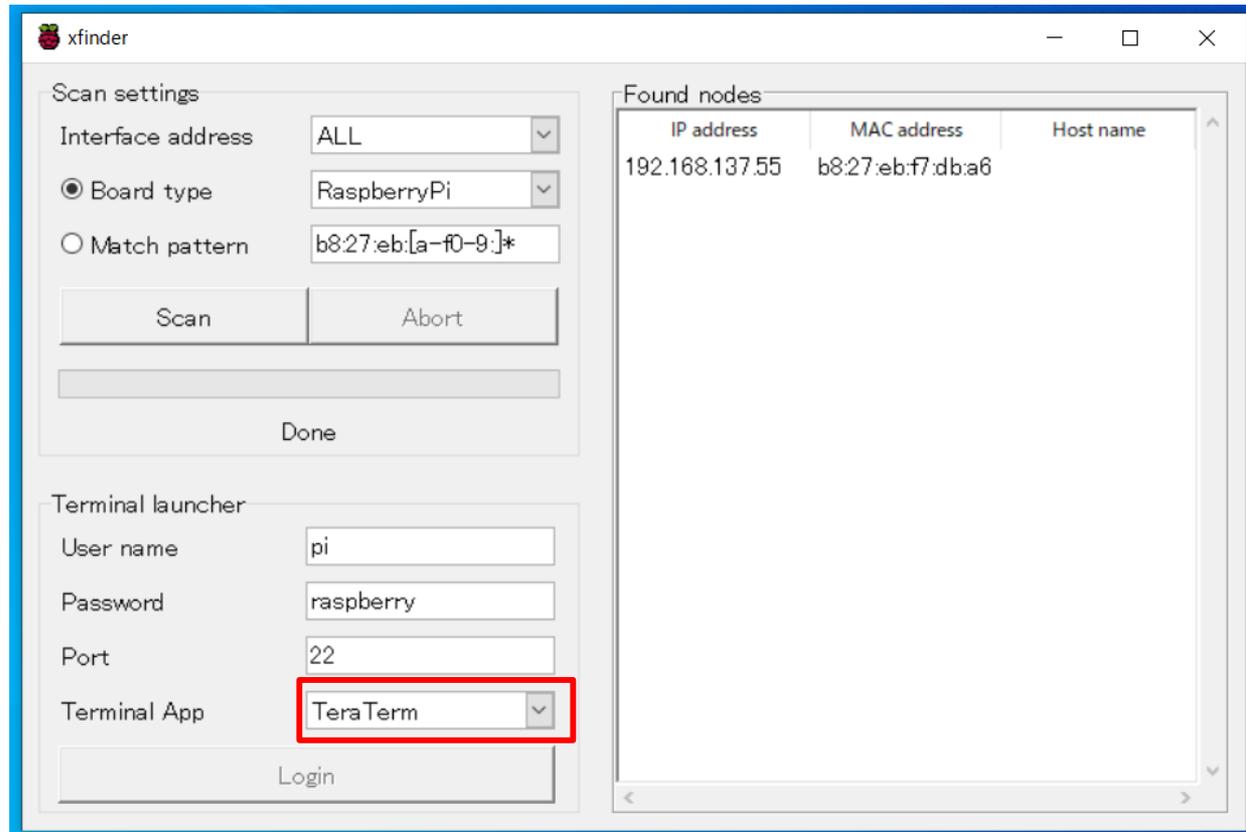
このアプリにはインストーラはないので適当なフォルダに保存し、このショートカットをデスクトップに作成する。



RaspberryPiOSの起動

RaspberryPiをLAN接続し、xfinder.exeを起動する。

デフォルトのターミナルアプリ(TerminalApp)は空白で好みのエディタを指定する。ここではTeraTermを選択する。



RaspberryPiOSの起動

RasPi4の8GBバージョンを入手する機会を得たところ、同じRasPi4でもRasPi4 4GBのMACアドレスとはベンダIDが異なることがわかった。

1GB、2GBは未入手なので、4GBと同じと考えていたが不明とした。

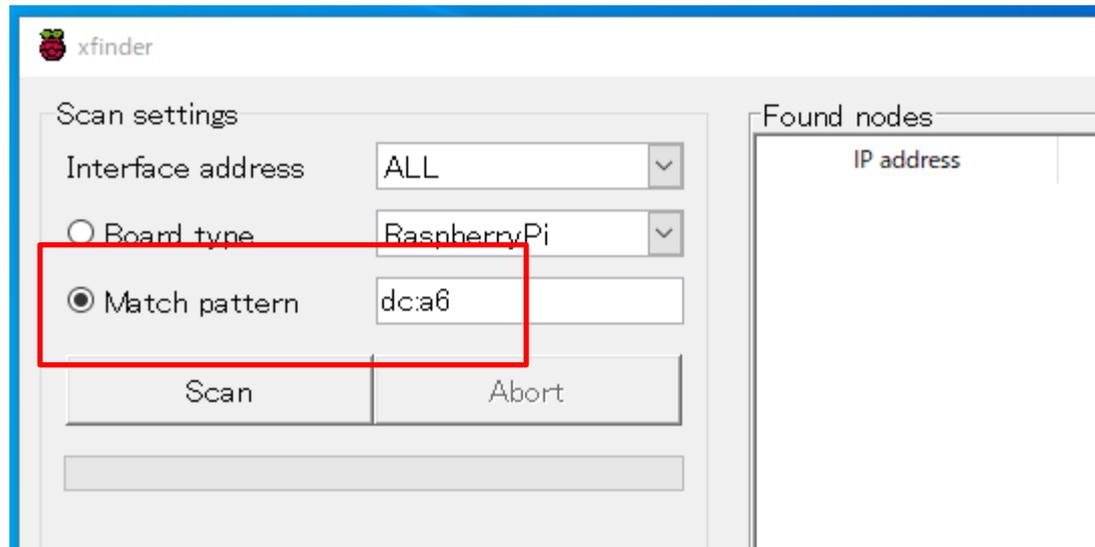
以下にNICのベンダIDを掲示しておく。

xfinderを使用する場合、Pi4はMatch patternでベンダIDの一部を記述することで使用が可能。

Pi4 (8GB)	e4:5f:01
Pi4 (4GB)	dc:a6:32
Pi4 (1GB,2GB)	不明
Pi3 + 以前(少なくともPi3,Pi2)	B8:27:eb

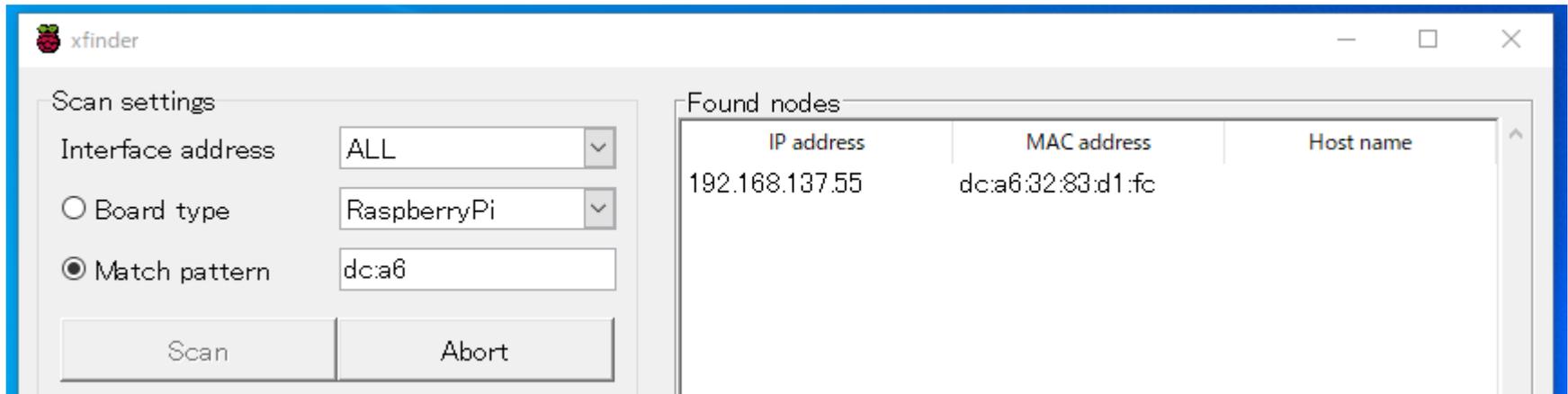
RaspberryPiOSの起動

RasPi4 4GBの検索で、Xfinderを利用する場合、「Match pertern」側で「dc:a6」とするとよい。



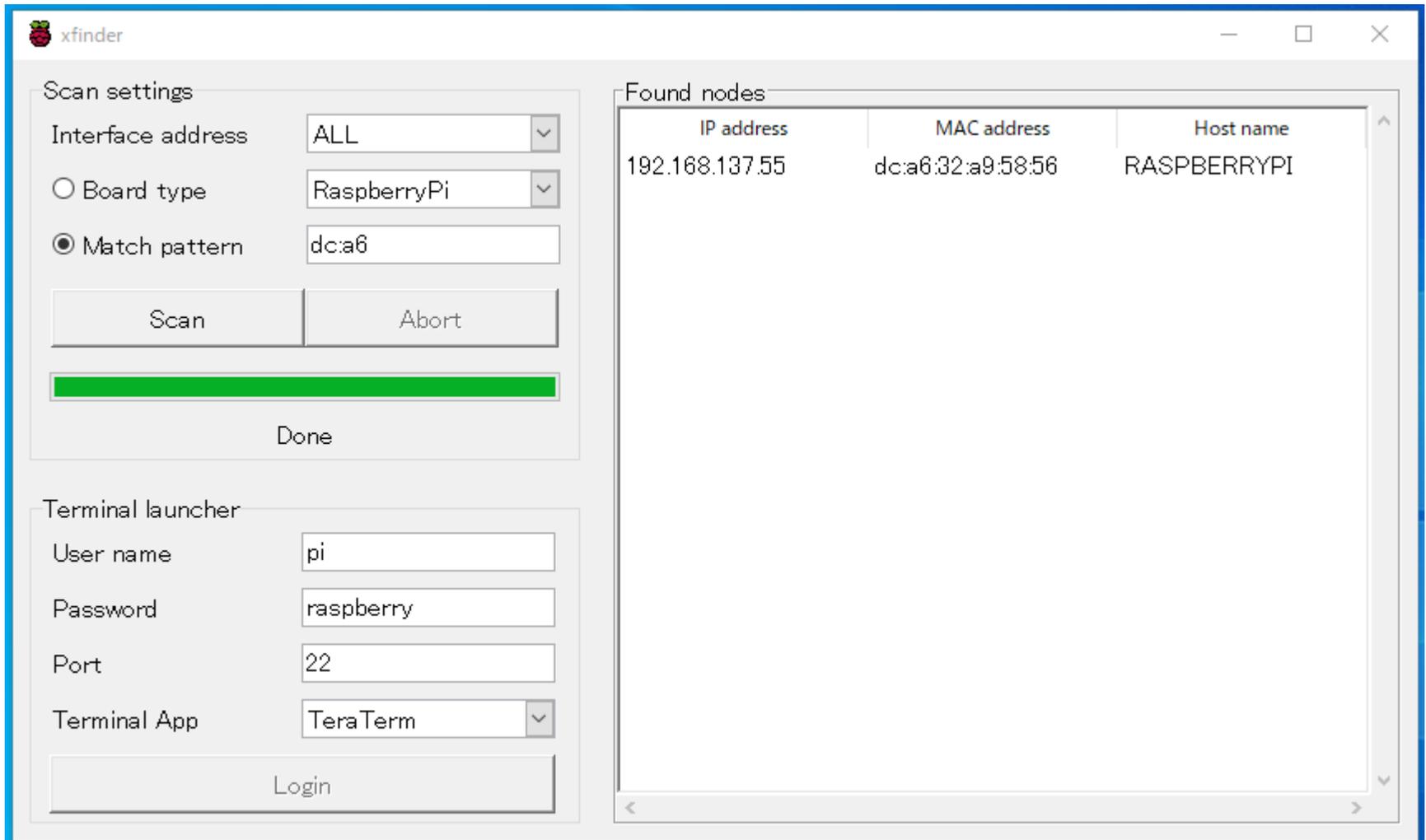
RaspberryPiOSの起動

「Scan」をクリックすると、ローカルネットワークにRaspberrypiを見つけることができる。ネットワーク上にRasPiが複数ある場合は、識別のためにHost nameを変えておけばよい。



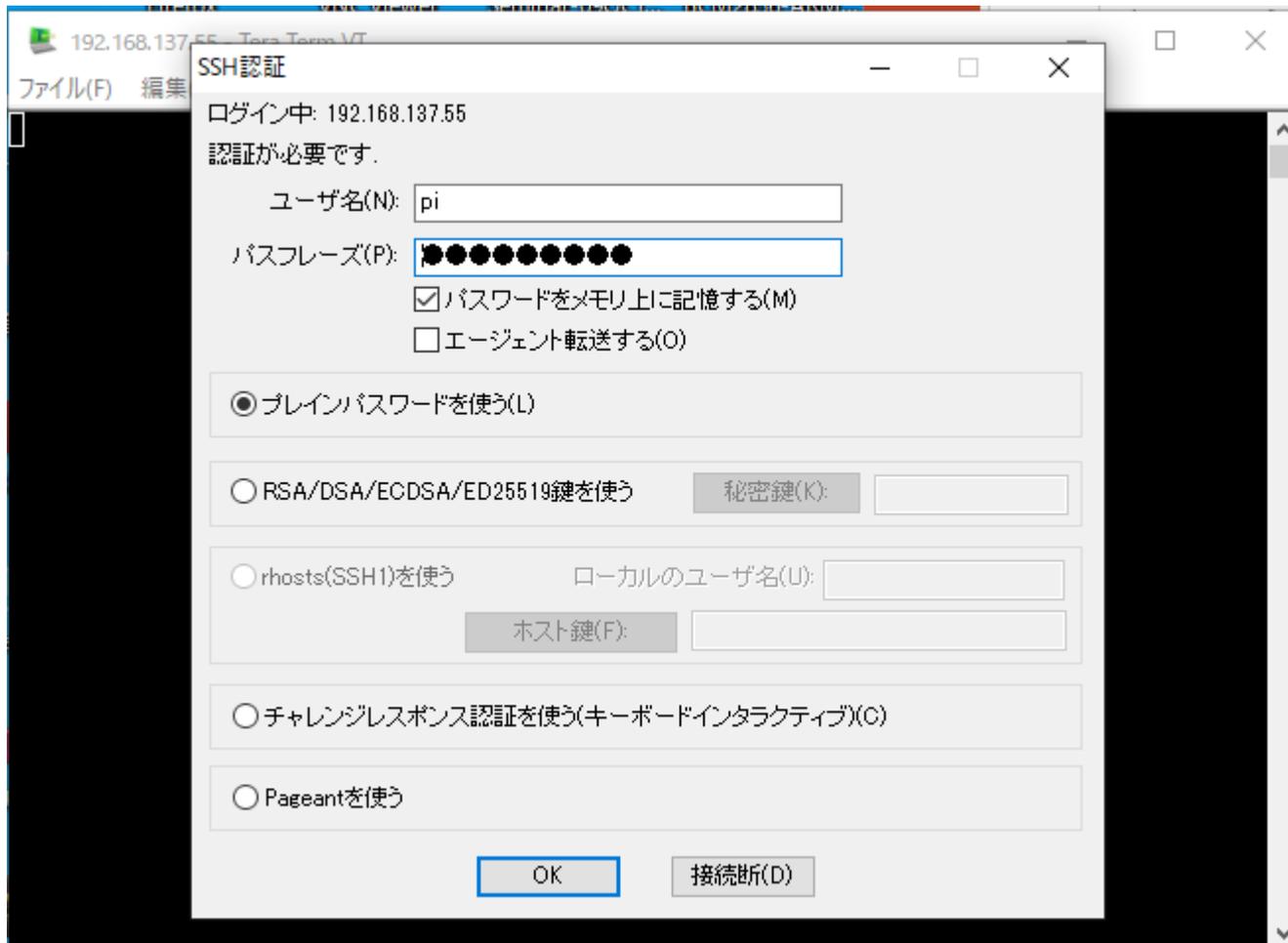
RaspberryPiOSの起動

対象のIPアドレスを選んで「Login」をクリック。



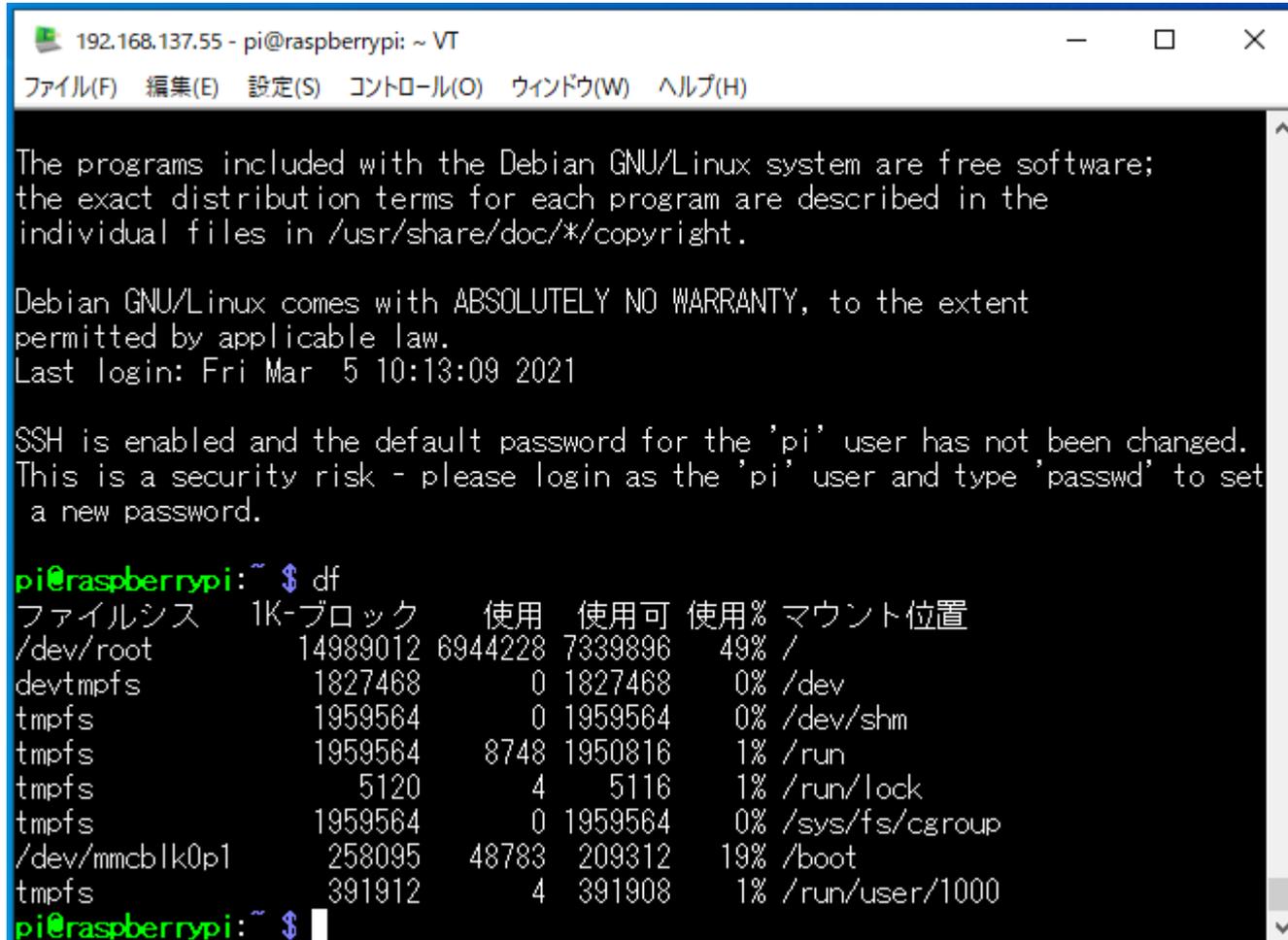
RaspberryPiOSの起動

SSHを有効にしているので、セキュリティ警告の「続行」を選択後、以下のようなTeraTerm起動が開く。「OK」を選択すると見慣れたターミナルが現れる。



RaspberryPiOSの起動

\$dfにてファイルシステムの確認。ubuntuPCで見たファイルブロック (/SHETTING, /boot ちなみに /rootは現在の自身)がある。



```
192.168.137.55 - pi@raspberrypi: ~ VT
ファイル(F) 編集(E) 設定(S) コントロール(O) ウィンドウ(W) ヘルプ(H)

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Mar  5 10:13:09 2021

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
a new password.

pi@raspberrypi:~ $ df
ファイルシス 1K-ブロック 使用 使用可 使用% マウント位置
/dev/root    14989012 6944228 7339896  49% /
devtmpfs    1827468  0 1827468  0% /dev
tmpfs       1959564  0 1959564  0% /dev/shm
tmpfs       1959564  8748 1950816  1% /run
tmpfs        5120  4  5116  1% /run/lock
tmpfs       1959564  0 1959564  0% /sys/fs/cgroup
/dev/mmcblk0p1 258095 48783 209312 19% /boot
tmpfs       391912  4 391908  1% /run/user/1000
pi@raspberrypi:~ $
```

RaspberryPiOSの起動

ntpクライアントにsystemd-timesyncdが使われており、インターネットに接続(ポートNo123が伝わる)ではそのままデフォルトでインターネット上のサーバと同期がとられるが、当施設のプロキシ環境では以下の状態となる。

なお、以下の内容は同期がとれない場合の対処であって、同期時間が図れる環境はスルーしてよい。(備忘録的情報)

```
$ timedatectl status
```

```
pi@raspberrypi:~ $ timedatectl status
      Local time: 金 2021-03-05 10:26:50 JST
      Universal time: 金 2021-03-05 01:26:50 UTC
      RTC time: n/a
      Time zone: Asia/Tokyo (JST, +0900)
System clock synchronized: no
      NTP service: active
      RTC in local TZ: no
pi@raspberrypi:~ $
```

RaspberryPiOSの起動

/etc/systemd/timesyncd.conf を編集する。

バックアップをとって編集する。

```
$ sudo cp timesyncd.conf timesyncd.conf.org
```

```
pi@raspberrypi:~ $ cd /etc/systemd
pi@raspberrypi:/etc/systemd $ sudo cp timesyncd.conf timesyncd.conf.org
```

```
$ sudo nano timesyncd.conf
```

```
pi@raspberrypi:/etc/systemd $ sudo nano timesyncd.conf
```

以下のコメントをはずし、

NTP= 以降にntpサーバを登録する。ここでは172.16.0.1(LAN内のタイムサーバ)を登録する。

```
[Time]
NTP=172.16.0.1
#FallbackNTP=0.debian.pool.ntp.org 1
#RootDistanceMaxSec=5
#PollIntervalMinSec=32
```

RaspberryPiOSの起動

systemd-timesyncdを再起動する。

```
pi@raspberrypi:/etc/systemd $ sudo systemctl restart systemd-timesyncd
```

同期履歴を調べる

```
$ sudo systemctl status systemd-timesyncd(.service)
```

```
pi@raspberrypi:~ $ sudo systemctl status systemd-timesyncd.service
Warning: The unit file, source configuration file or drop-ins of systemd-timesyncd.service ch
• systemd-timesyncd.service - Network Time Synchronization
  Loaded: loaded (/lib/systemd/system/systemd-timesyncd.service; enabled; vendor preset: ena
  Drop-In: /usr/lib/systemd/system/systemd-timesyncd.service.d
           └─disable-with-time-daemon.conf
  Active: active (running) since Thu 2021-05-20 10:24:05 JST; 10min ago
  Docs: man:systemd-timesyncd.service(8)
  Main PID: 323 (systemd-timesyn)
  Status: "Synchronized to time server for the first time 172.16.0.1:123 (172.16.0.1)."
```

Tasks: 2 (limit: 4915)
CGroup: /system.slice/systemd-timesyncd.service
└─323 /lib/systemd/systemd-timesyncd

```
5月 20 10:24:04 raspberrypi systemd[1]: Starting Network Time Synchronization...
5月 20 10:24:05 raspberrypi systemd[1]: Started Network Time Synchronization.
5月 20 10:24:53 raspberrypi systemd-timesyncd[323]: Synchronized to time server for the first
```

:q で終了

RaspberryPiOSの起動

ntpdateによる時刻設定は「付録2」に提示する。

RaspberruyPiOSの起動

時刻設定済後、ファイルのアップデートをする。

プロキシ環境で作業する場合は以下の書式でコマンド処理する。

```
$sudo http_proxy=http://(プロキシサーバIP):(ポート) apt-get update
```

```
pi@raspberrypi:~ $ sudo http_proxy=http://172.16.0.8:8080 apt-get update
```

このように apt-get コマンドで毎回、プロキシサーバ指定するのは大変なので /etc/apt/apt.conf ファイルを新規作成し、保存する。

ここではubuntu系エディタ”nano”で編集する。(行末のセミコロンを忘れずに)

```
pi@raspberrypi:~ $ sudo nano /etc/apt/apt.conf
```

```
Acquire::http::proxy "http://172.16.0.8:8080";  
Acquire::https::proxy "https://172.16.0.8:8080";  
Acquire::ftp::proxy "ftp://172.16.0.8:8080";  
Acquire::socks::proxy "socks://172.16.0.8:8080";
```

Ctrl+O→「enter」、Ctrl+Xで保存と終了。

再起動で .conf ファイルを読み込み。

RaspberryPiOSの起動

updateには本施設環境で約2分ほど要する。

次に、ファイルのアップグレードをする。

/etc/apt/apt.conf を作成したので

\$sudo apt-get upgrade として プロキシサーバとポート番号の付加は省略できる。

```
pi@raspberrypi:~ $ sudo apt-get upgrade
```

約40分ほどにてプロキシ環境では完全に終了しないことがある。

```
- SHA1:954071acae80f700d252ff54735dd33b2778d5c5 [weak]
- MD5Sum:c3382b0082c94d505edc594d14d58ed5 [weak]
- Filesize:2728708 [weak]
E: いくつかのアーカイブを取得できません。apt-get update を実行するか --fix-missing ください。
pi@raspberrypi:~ $
```

~~update~~ ⇔ upgrade を何度か繰り返すとメッセージは少なくなる。

RaspberryPiOSの起動

日本語環境をインストールする。ここでは日本語入力ソフト「ibus-anthy」と「fonts-takao」をインストールする。

```
$ sudo apt-get install -y ibus-anthy (約3分の処理)
```

```
$ sudo apt-get install -y fonts-takao (1分以内の処理)
```

```
pi@raspberrypi:~ $ sudo apt-get install -y ibus-anthy
```

```
pi@raspberrypi:~ $ sudo apt-get install -y fonts-takao
```

Proxy環境では「エラーメッセージ」が出るが何度か繰り返すとメッセージが少なくなる。

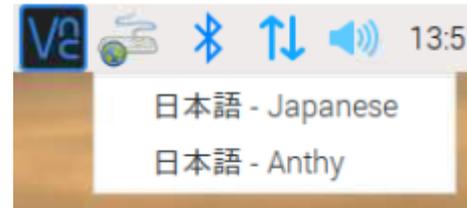
再起動後、iBusの設定ができるようになる。

また、ibus-anthyは日本語フォントが使える状況で、かな漢が使用できるようになる。コマンドラインでの日本語入力、Webでの日本語使用を考えている場合はibus-anthyのインストールをしておく。

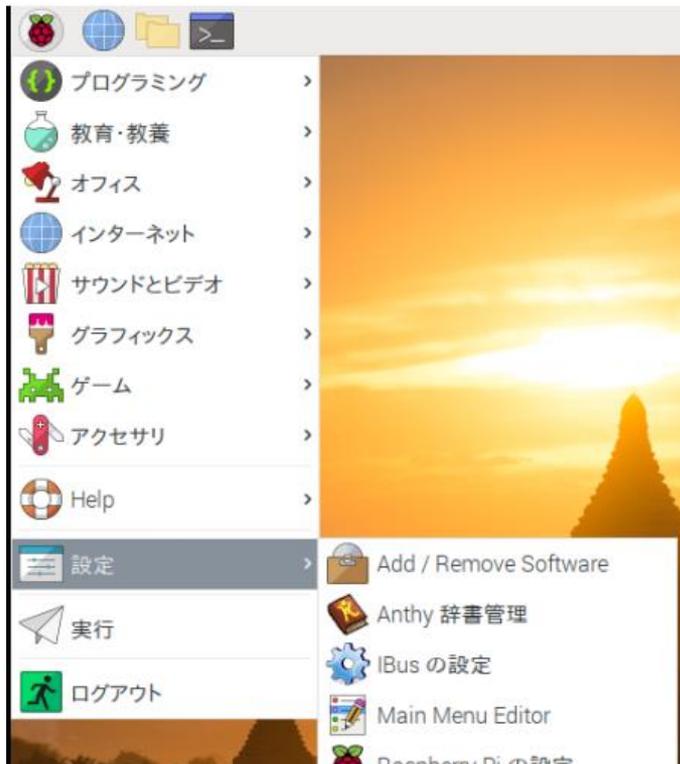
RaspberryPiOSの起動

リブート後、ibus-anthyのインストールで右上の表示に「あ」が現れる。

```
pi@raspberrypi:~$ sudo reboot
```



「設定」→「iBusの設定」メニューが追加される。



RaspberryPiOSの起動

WEBブラウザにはchromiumが標準で装備されているが、proxy環境では接続できない。そのため、/etc/profile.d にproxy.shファイルを準備する。

```
pi@raspberrypi:~ $ sudo nano /etc/profile.d/proxy.sh
```

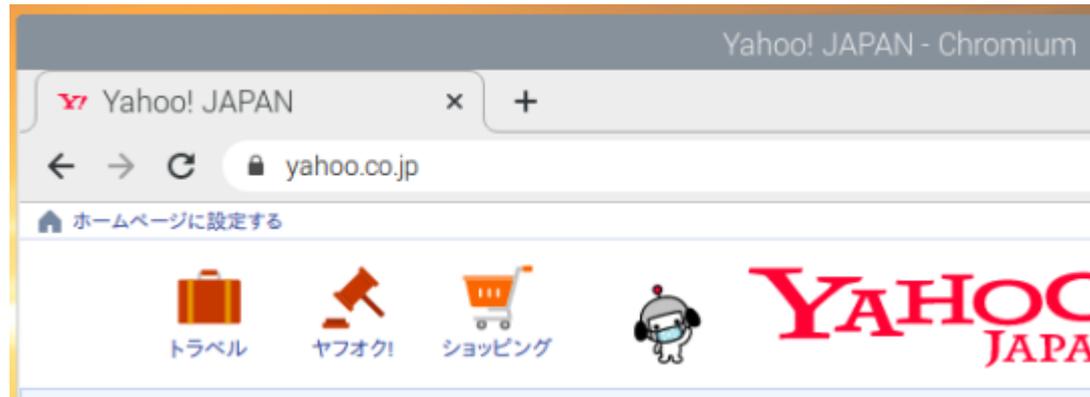
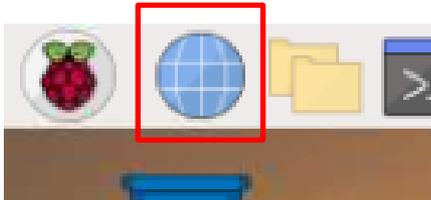
```
GNU nano 3.2 /etc/profile.  
  
export http_proxy=http://172.16.0.8:8080/  
export https_proxy=http://172.16.0.8:8080/  
export ftp_proxy=http://172.16.0.8:8080/  
export HTTP_PROXY=http://172.16.0.8:8080/  
export HTTPS_PROXY=http://172.16.0.8:8080/  
export FTP_PROXY=http://172.16.0.8:8080/
```

Proxy.sh を読み込ませるため再起動する。

(ネットワーク機能の再起動ではproxy.sh内容は反映されない)。

RaspberryPiOSの起動

WEBブラウザを起動する。



RaspberryPiOSの起動

VNC機能が標準装備となって久しい。

一方、VNCサーバにはtightvncserverがあり、この機能も捨て難いのでこの2つ接続方法を示す。

tightvncserverは標準VNCと違い、個別にターミナルを複数生成することができる。

また、RaspbianOSでは標準VNCをリモートで使用する場合は、Pi4をモニタにつないだ状態か、あるいはモニタ接続を疑似できるアナログ出力変換器をつないだ状態にしておく必要があったが、RaspberryPiOSではアナログ変換器の疑似接続もできず、モニタ接続が必須になっている。

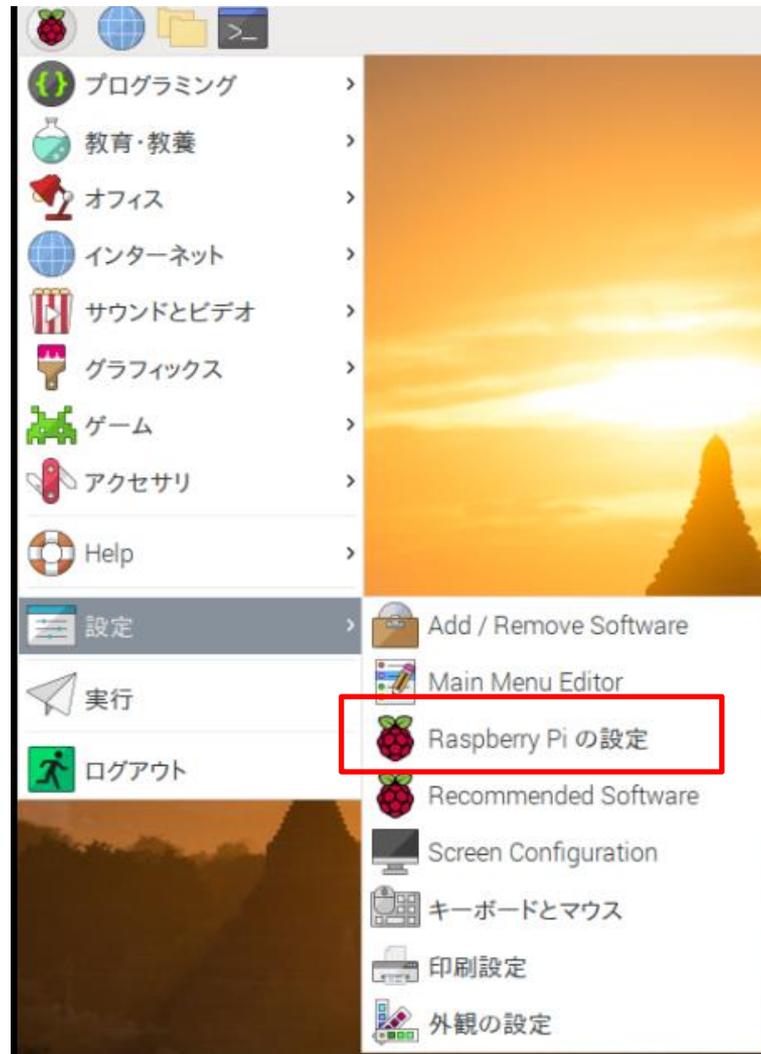
モニタなしで制御するにはssh接続にてリモートするほかない。

モニタなしでGUIでのリモート操作する場合はtightvncserverを選択する。

また、かつてはtightvncserverから標準VNCに戻すのが容易だったが、バージョンを重ねるうち、元に戻すのが困難な状態になっているのでVNCサーバーはどちらか一方で使用するもの考えるべきものになっている。

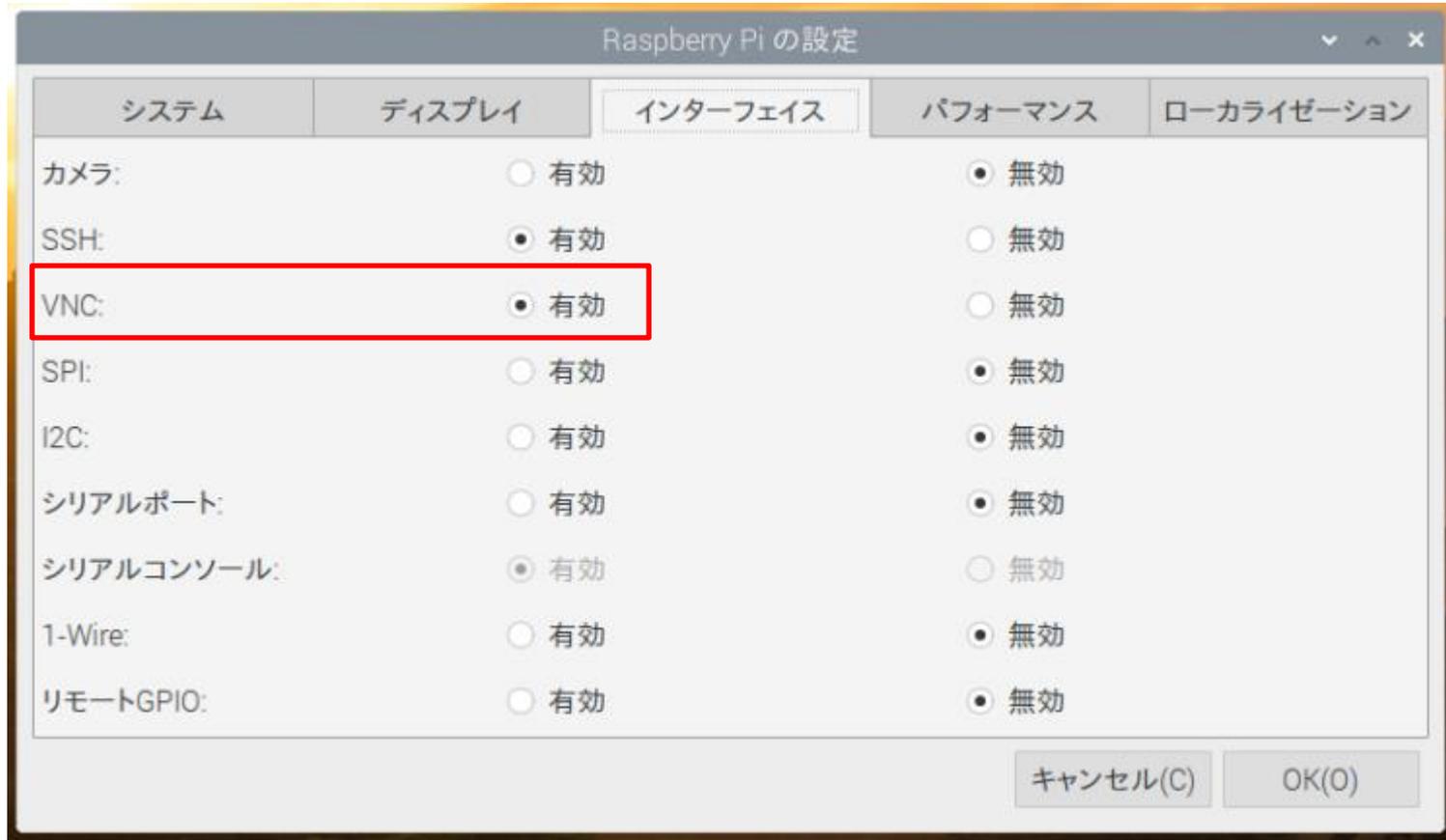
RaspberryPiOSの起動

まずは標準VNCの起動について。「設定」→「RaspberryPiの設定」



RaspberryPiOSの起動

「インターフェイス」を選択し「VNC」を有効を確認する。

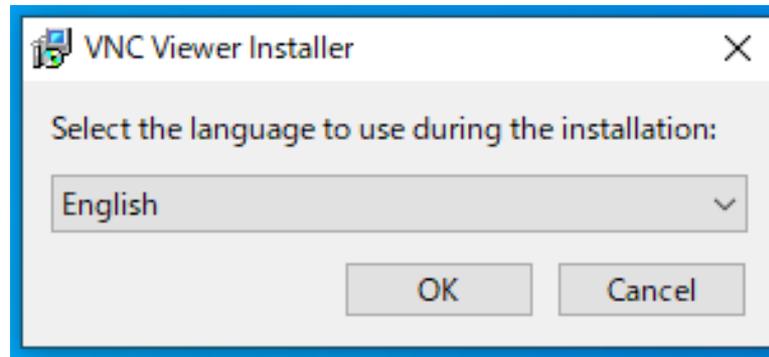


RaspberryPiOSの起動

VNCサーバに対し、クライアント側はWindowsで動作するVNC Viewerを使用する。2021.5現在VNC-Viewer-6.21.406。

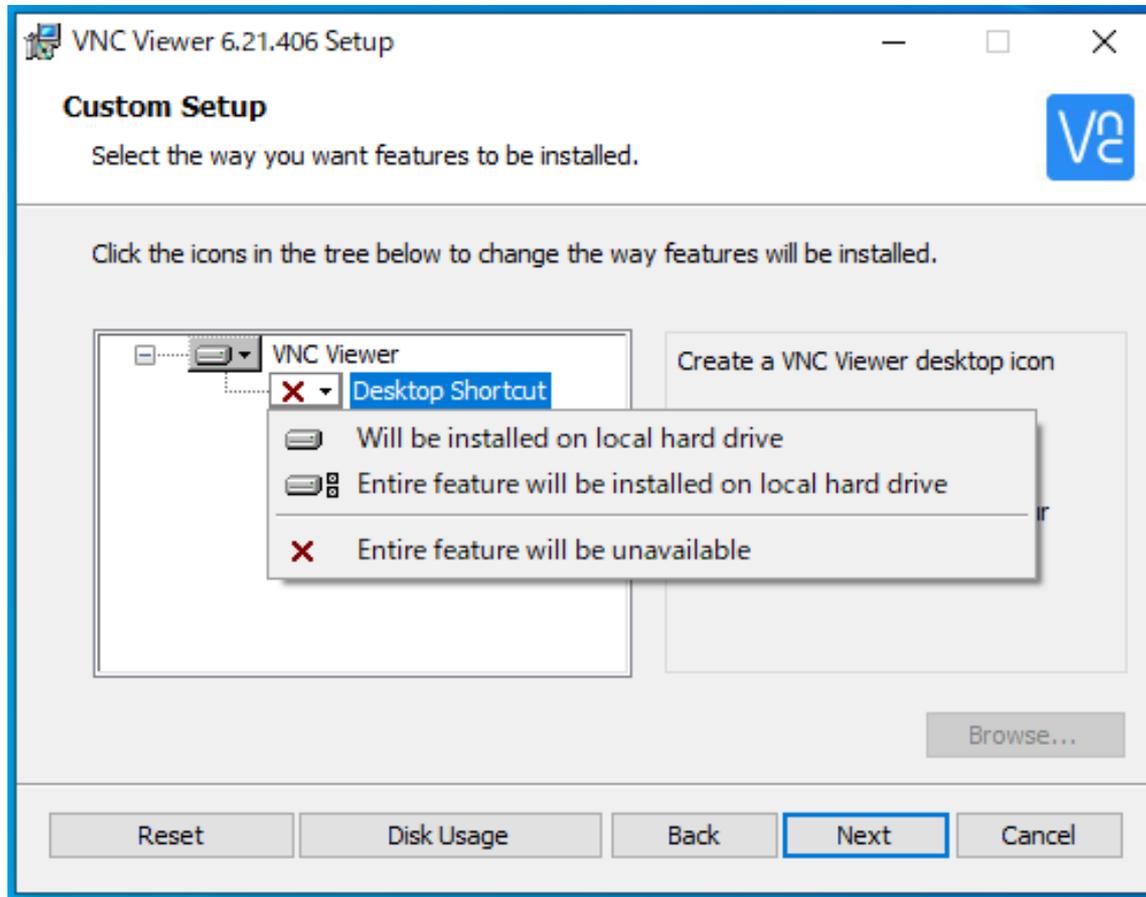
VNC Viewerのインストールはネットからダウンロードできる。
(配布データに収録)

VNC-Viewer-6.21.406-Windows.exeを実行する。



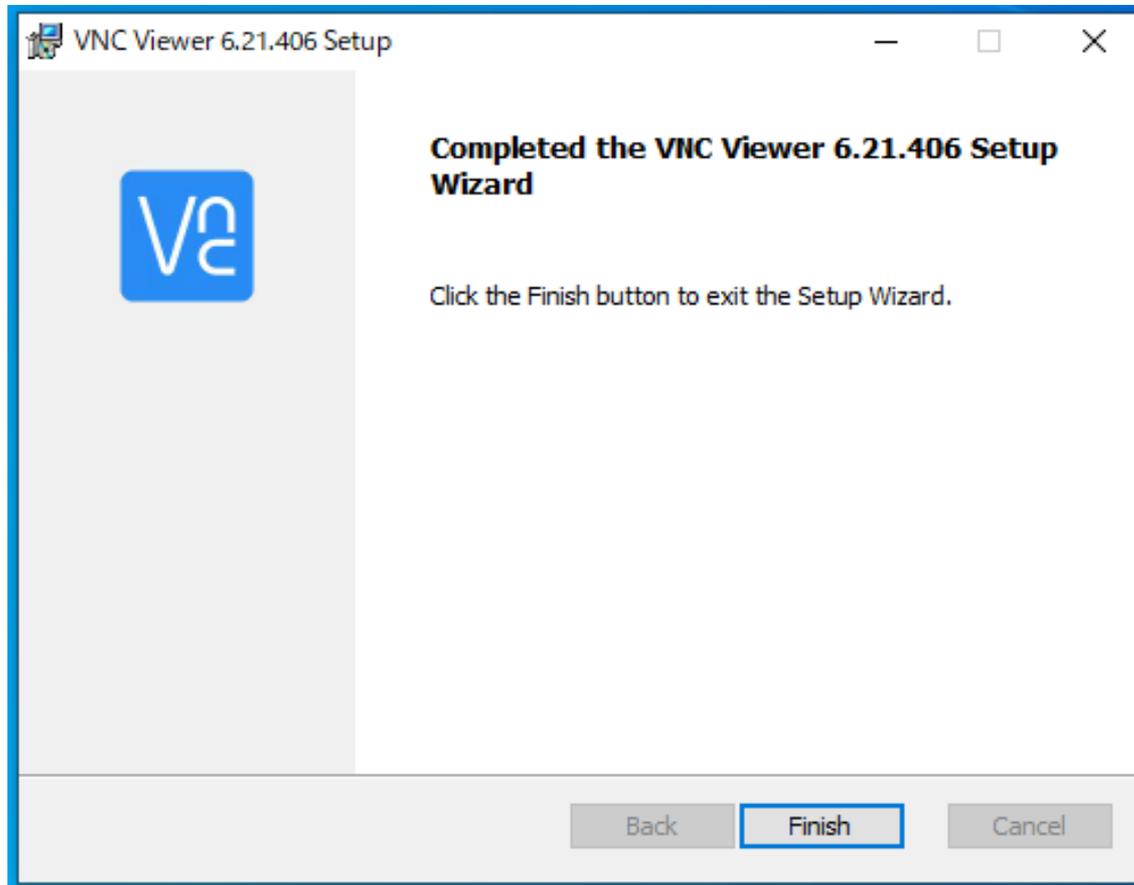
RaspberryPiOSの起動

Desktop Shortcutを作成する。



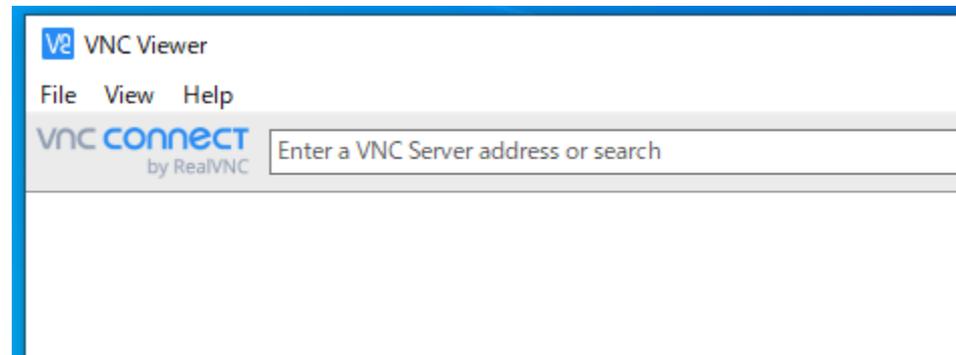
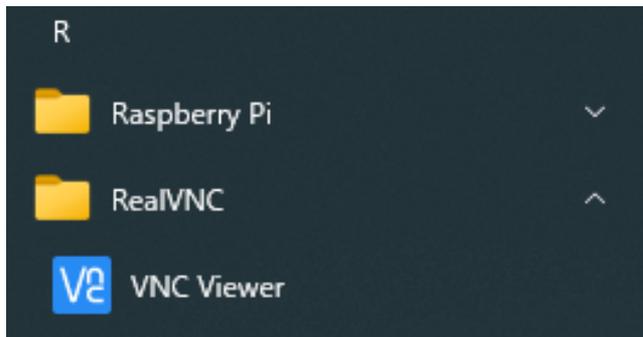
RaspberryPiOSの起動

「Finish」。



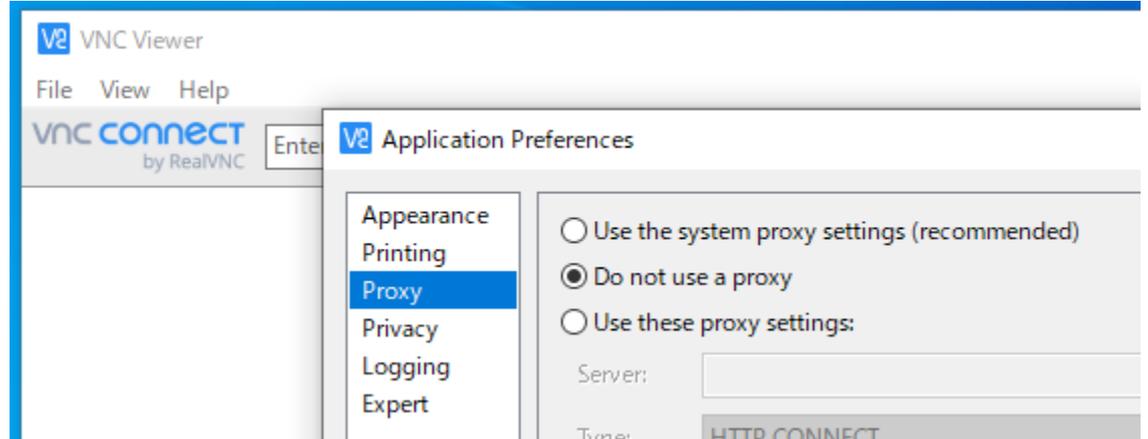
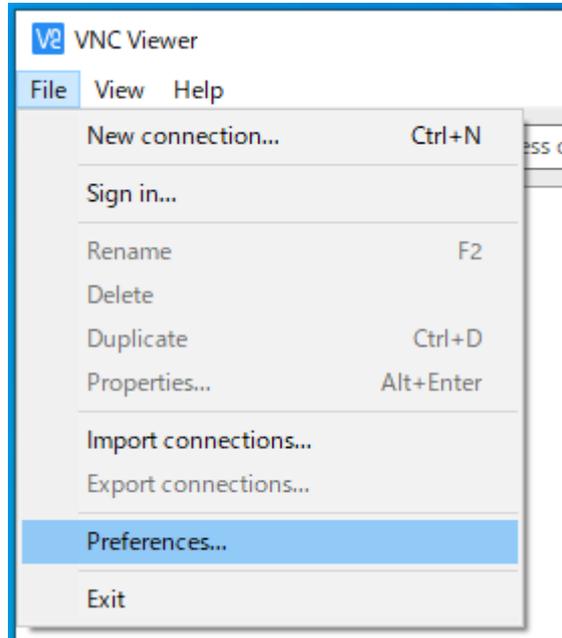
RaspberryPiOSの起動

RealVNC → VNC Viewer で起動する。

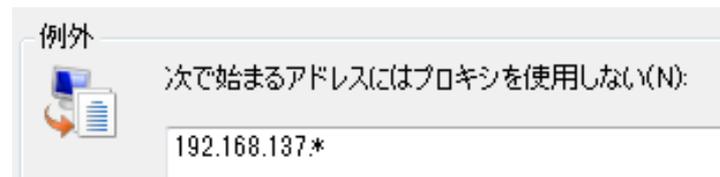


RaspberryPiOSの起動

プロキシ環境の場合、File → Preferences.. から Proxy → 「Do not use a proxy」を選択する。

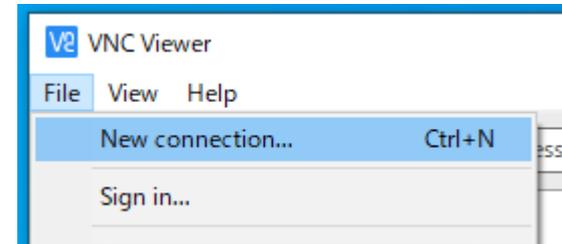


また、あらかじめPCで以下のようにproxyから除外している場合はデフォルト設定でもOK

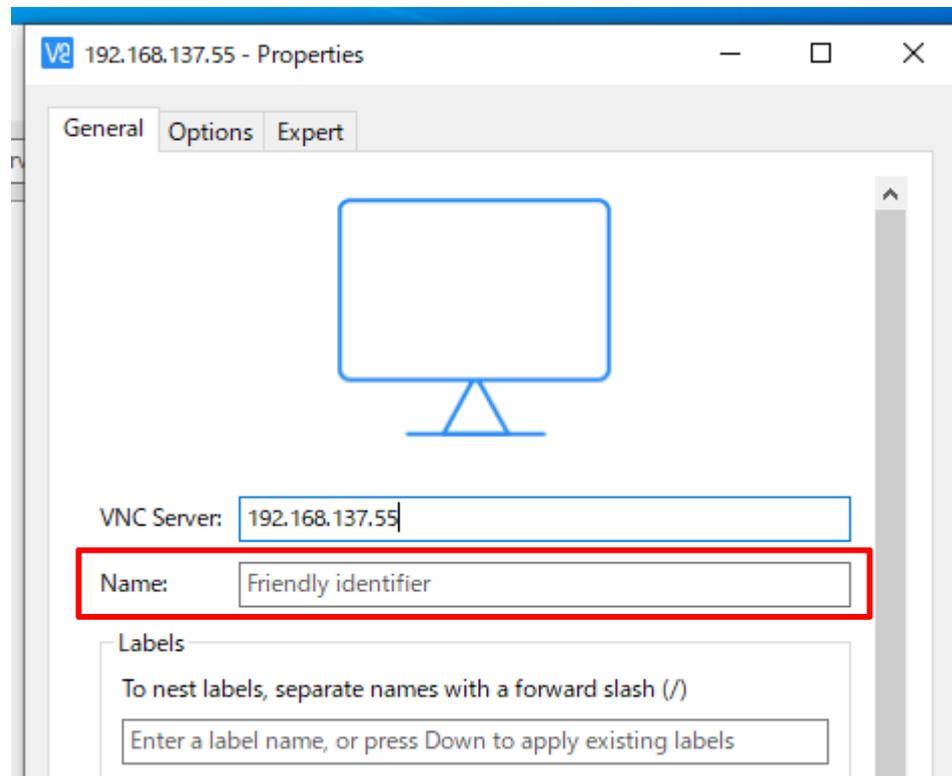


RaspberryPiOSの起動

VNC Viewerにて「File」→「New connection」。

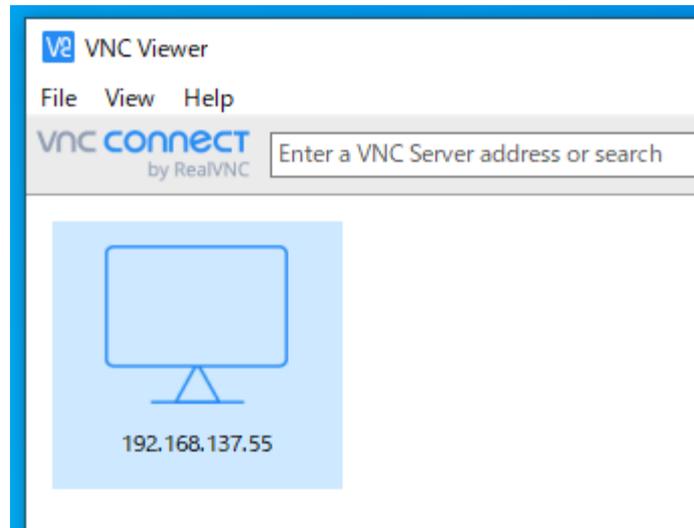


IPアドレスと次回からの利便性のため自由なサーバー名を「Name」に登録できる。



RaspberryPiOSの起動

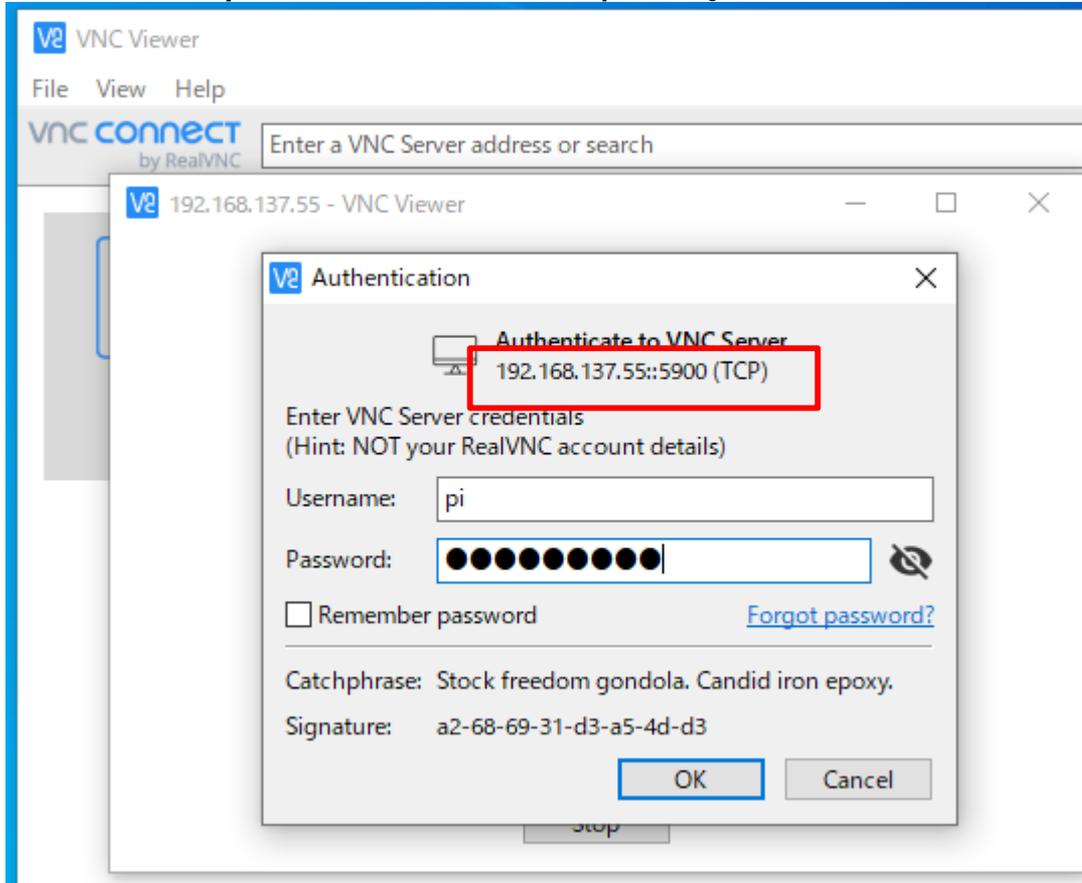
Nameを空白にすると、VNC ServerのIPアドレスが表示される。



対象サーバをダブルクリック

RaspberryPiOSの起動

Contunueの後、サーバのIPアドレスとポート番号(5900)が確認できる。
デフォルトは Username「pi」、Password「raspberrypi」



(なお、tightvncserverのポート番号は5901)

RaspberryPiOSの起動

Raspiウィンドウが現れる。

モニタ出力と同じ解像度で現れる。Skype画面も本モニタで確認できる。

