

# TCの構築と運用方法

## 1. 背景と目的

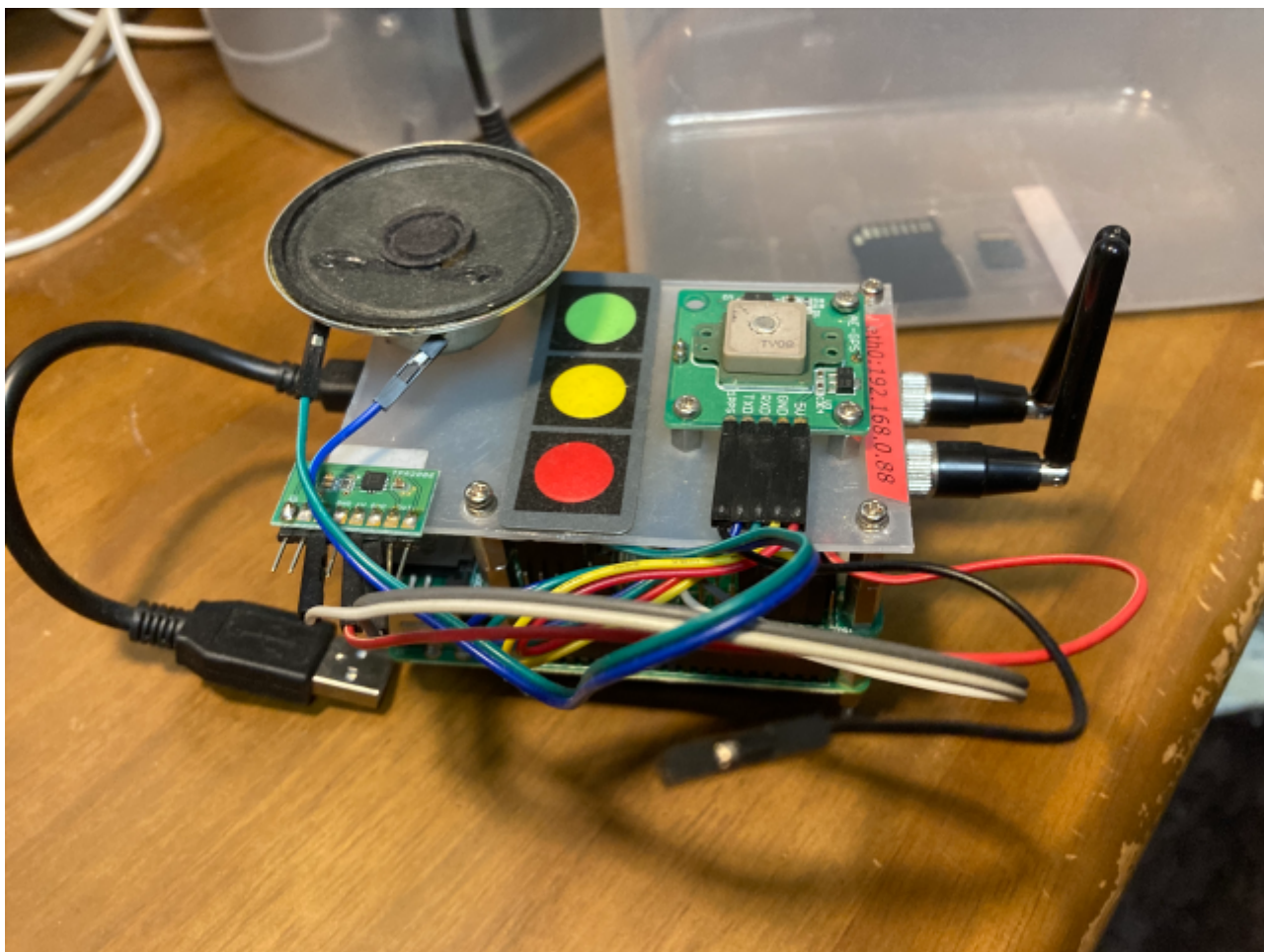
実験システム用にTCが2台必要となったので、2019年の実証実験で使っていたラズパイを改造してTCにすることにした。

TC (Traffic Control) は、ネットワークトラフィックの管理と制御を行うツールで、主に iproute2 パッケージに含まれている。TCはネットワークインターフェースのトラフィックに対して、帯域制限や遅延、パケット損失などのシミュレーションを行うための強力な機能を提供する。

## 2. 今回のターゲットとしたラズパイとその構成

本ラズパイのディストリビューションはRaspbianである。

```
Linux raspberrypi 4.19.66-v7+ #1253 SMP Thu Aug 15 11:49:46 BST 2019 armv7l  
GNU/Linux
```



ラズパイにはイーサポートが1つしかないなので、2つ目のポートは、USB-Ethを使う。

<https://amzn.to/3BKhXyv>

Cable Matters USB LAN変換アダプター 有線LANアダプター USB イーサネットアダプタ USB2.0 to RJ45



## 3. 古いラズパイの再構築

---

### 3.1. 状況の把握

古いラズパイを再利用するために、**sudo apt-get update**をすると、

```
E: Failed to fetch http://raspbian.raspberrypi.org/raspbian/pool/main/c/cyrus-sasl2/libsasl2-modules_2.1.27~101-g0780600+dfsg-3+deb9u1_armhf.deb 404 Not Found [IP: 2a00:1098:0:80:1000:75:0:3 80]
E: Failed to fetch
```

```
http://raspbian.raspberrypi.org/raspbian/pool/main/libv/libvpx/libvpx4_1.6.1-3+deb9u2_armhf.deb 404 Not Found [IP: 2a00:1098:0:80:1000:75:0:3 80]
```

という表示がでてくる。サポートは数年前に終わっているためである。

## 3.2. リポジトリの更新

この場合リポジトリのURLが古いので、`/etc/apt/sources.list`または`/etc/apt/sources.list.d/raspi.list`を編集し、最新のリポジトリURLを使用していることを確認する。

```
sudo nano /etc/apt/sources.list.d/raspi.list
```

`deb`行を以下のように変更する：

```
deb http://archive.raspbian.org/raspbian/ buster main contrib non-free rpi
```

---

## 3.3. パッケージリストを更新

リポジトリURLを変更した後、以下のコマンドを実行してパッケージリストを更新する。

```
sudo apt-get update
```

## 3.4. リリースのアップグレードを検討

Raspbianのバージョンが古い場合（例えば、`stretch`や`jessie`）、サポートされていない可能性がある。この場合、`buster`や`bullseye`などの新しいバージョンにアップグレードを検討する。

アップグレード手順の例：

```
sudo apt-get update
sudo apt-get upgrade
sudo apt-get dist-upgrade
sudo apt-get autoremove
```

その後、リリースをアップグレード：

```
sudo sed -i 's/stretch/buster/g' /etc/apt/sources.list
sudo sed -i 's/stretch/buster/g' /etc/apt/sources.list.d/raspi.list
sudo apt-get update
sudo apt-get upgrade
```

## 4. 設定するTCの環境

### 4.1. TCの2つのイーサポート

今回はeth0と、USB LAN変換アダプターによって作られたeth1の2つのポートを使用する。

```
pi@raspberrypi:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    ether b8:27:eb:69:5b:65 txqueuelen 1000 (Ethernet)
    RX packets 438 bytes 40624 (39.6 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 246 bytes 28578 (27.9 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    ether f4:4d:ad:04:49:d5 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 22 bytes 1372 (1.3 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 6 bytes 234 (234.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 6 bytes 234 (234.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::ba27:ebff:fe3c:e30 prefixlen 64 scopeid 0x20<link>
    ether b8:27:eb:3c:0e:30 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 18 bytes 2694 (2.6 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

本来TCには、IPアドレスを付与する必要はない。なぜならTCとは、有線を切断して、その間に挟んで、トラフィックを制御する装置であるからである。

但し、今回はSSHを使いたいので、eth0に固定のIPアドレスを付与することとする。

### 4.2. bridge-utils の確認とインストール

TCのパッケージである、`bridge-utils` がインストールされているか確認する。

```
dpkg -l | grep bridge-utils
```

出力がない場合、以下を実行してインストールする。

```
sudo apt update
sudo apt install bridge-utils -y
```

### 4.3. /etc/network/interfacesの設定設定

以下のように記載する。

```
# Include files from /etc/network/interfaces.d:
source-directory /etc/network/interfaces.d

# ブリッジ設定
auto br0
iface br0 inet static
    address 192.168.0.88
    netmask 255.255.255.0
    gateway 192.168.0.1
    bridge_ports eth0 eth1
```

修正後、以下のコマンドでネットワークサービスを再起動する。

```
sudo systemctl restart networking
```

上記によって、以下のbr0が作成されていることを確認する。

```
pi@raspberrypi:~ $ ifconfig
br0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.88 netmask 255.255.255.0 broadcast 192.168.0.255
    inet6 fe80::ba27:ebff:fe85:6108 prefixlen 64 scopeid 0x20<link>
    inet6 240f:30:a9ce:1:ba27:ebff:fe85:6108 prefixlen 64 scopeid
0x0<global>
    ether b8:27:eb:85:61:08 txqueuelen 1000 (Ethernet)
    RX packets 77691 bytes 5352437 (5.1 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 5414 bytes 391872 (382.6 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    ether b8:27:eb:85:61:08 txqueuelen 1000 (Ethernet)
    RX packets 81368 bytes 7178197 (6.8 MiB)
    RX errors 0 dropped 2 overruns 0 frame 0
```

```
TX packets 7924 bytes 686585 (670.4 KiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

(以下省略)

## 4.4. dhcpd.conf を編集編集

dhcpd を使用している場合は、以下の手順を実行する。

1. dhcpd.conf を編集する。

```
sudo nano /etc/dhcpd.conf
```

2. 以下を追加する。

```
interface eth0
static ip_address=192.168.0.88/24
static routers=192.168.0.1
static domain_name_servers=8.8.8.8
```

3. dhcpd を再起動する。

```
sudo systemctl restart dhcpd
```

## 5. TCの設定の恒久化

---

TCは手動でも設定できるが、スクリプトを作成してデフォルトの設定をしておく。

### 5.1. スクリプトの作成作成

/usr/local/bin/setup\_tc.sh に以下の内容を記載する。

```
#!/bin/bash
# TC 設定スクリプト

# Reset existing settings
tc qdisc del dev br0 root 2>/dev/null

# Add HTB qdisc
tc qdisc add dev br0 root handle 1: htb default 10

# Define root class
tc class add dev br0 parent 1: classid 1:1 htb rate 100mbit
```

```
# Define subclasses for traffic control
tc class add dev br0 parent 1:1 classid 1:10 htb rate 50mbit ceil 100mbit
tc class add dev br0 parent 1:1 classid 1:20 htb rate 30mbit ceil 50mbit

# Add a filter to classify all traffic to default class 1:10
tc filter add dev br0 protocol ip parent 1:0 prio 1 u32 match u32 0 0 flowid 1:10

# Add artificial delay (optional)
tc qdisc add dev br0 parent 1:10 handle 10: netem delay 50ms
```

### 5.1.1. スクリプトの内容説明説明

- `tc qdisc del`: 既存の設定をリセットします。
- `tc qdisc add`: HTB を使用して帯域幅制御を設定します。
- `tc class add`: 帯域幅と優先度をクラスごとに設定します。
  - `1:10`: 50 Mbps の帯域幅を割り当て。
  - `1:20`: 30 Mbps の帯域幅を割り当て（必要に応じて追加）。
- `tc filter add`: 全トラフィックをクラス `1:10` に割り当てます。
- `netem delay`: 遅延を 50ms に設定（必要に応じて変更）。

---

## 5.2. スクリプトに実行権限を付与

作成したスクリプトに実行権限を付与する。

```
sudo chmod +x /usr/local/bin/setup_tc.sh
```

---

## 5.3. スクリプトの起動時の自動実行

以下のスクリプトの作成と設定を行う。

1. `/etc/rc.local` に追加 `/etc/rc.local` を編集する。

```
sudo nano /etc/rc.local
```

以下を `exit 0` の直前に追加する。

```
/usr/local/bin/setup_tc.sh
```

保存後のファイル例:

```
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# By default this script does nothing.

/usr/local/bin/setup_tc.sh

exit 0
```

## 2. rc.local に実行権限を付与

```
sudo chmod +x /etc/rc.local
```

---

## 5.4. 設定のテスト

設定が起動時に正しく反映されることを確認する。

### 1. スクリプトを手動実行して確認

```
sudo /usr/local/bin/setup_tc.sh
```

設定後、以下のコマンドで確認します。

```
tc qdisc show dev br0
tc class show dev br0
tc filter show dev br0
```

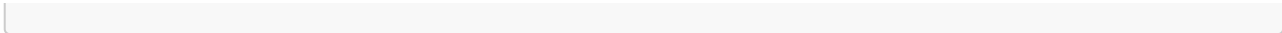
### 2. システムを再起動して確認

```
sudo reboot
```

再起動後、tc 設定が反映されているか再度確認する

```
tc qdisc show dev br0
tc class show dev br0
tc filter show dev br0
```





以上