tcconfig Documentation

Release 0.9.0

Tsuyoshi Hombashi

Apr 12, 2017

Table of Contents

CHAPTER 1

tcconfig

Summary

A Simple tc command wrapper tool. Easy to set up traffic control of network bandwidth/latency/packet-loss to a network interface.

Traffic control features

Network

Traffic control can be specified network to apply to:

- Outgoing/Incoming packets
- Certain IP address/network and port

Available parameters

The following parameters can be set to network interfaces.

- Network bandwidth rate [G/M/K bps]
- Network latency [milliseconds]
- Packet loss rate [%]
- Packet corruption rate [%]

CHAPTER 2

Installation

Installing from PyPI

`tcconfig` can be installed from PyPI via pip (Python package manager) command.

```
sudo pip install tcconfig
```

Installing from binary

`tcconfig` can be installed environments which cannot access to PyPI directly:

1. `https://github.com/thombashi/tcconfig/releases/download/v0.7.0/tcconfig_wheel.tar.gz`
2. `tar xvf tcconfig_wheel.tar.gz`
3. `cd tcconfig_wheel/`
4. `./install.sh`

CHAPTER 3

Dependencies

Linux packages

- iproute2 (mandatory: required for tc command)
- iptables (optional: required to when you use `--iptables` option)

Linux kernel module

- sch_netem

Python packages

Dependency python packages are automatically installed during `tcconfig` installation via pip.

- DataPropery
- ipaddress
- logbook
- pyparsing
- six
- subprocrunner
- typepy
- voluptuous

Optional

- **netifaces**
  - Suppress excessive error messages if this package is installed

Test dependencies

- allpairspy
- pingparsing
- pytest
- pytest-runner
- tox

CHAPTER 4

Usage

Set traffic control (`tcset` command)

`tcset` is a command to add traffic control rule to a network interface (device).

You can delete rule(s) from a network interface by *Delete traffic control (tcdel command)*.

`tcset` command help

usage: tcset [-h] [–version] [–tc-command | –tc-script] [–debug | –quiet] (–device DEVICE | -f CONFIG_FILE) [–overwrite | –add] [–direction {outgoing,incoming}] [–rate BANDWIDTH_RATE] [–delay NETWORK_LATENCY] [–delay-distro LATENCY_DISTRO_MS] [–loss PACKET_LOSS_RATE] [–corrupt CORRUPTION_RATE] [–network NETWORK] [–port PORT] [–ipv6] [–shaping-algo {tbf,htb}] [–iptables] [–src-network SRC_NETWORK]

optional arguments:

-h, --help	show this help message and exit
--version	show program's version number and exit
--tc-command	display tc commands to be executed and exit. commands are not actually executed.
--tc-script	generate a script file that described tc commands to be executed by this command.
--debug	for debug print.
--quiet	suppress execution log messages.
--device DEVICE	network device name (e.g. eth0)
-f CONFIG_FILE, --config-file CONFIG_FILE	setting traffic controls from a configuration file. output file of the tcshow.
--overwrite	overwrite existing settings

--add add a traffic shaping rule in addition to existing rules.

Traffic Control:

–direction {outgoing,incoming} the direction of network communication that impose traffic control. `incoming` requires Linux kernel version 2.6.20 or later. (default = `outgoing`)

--rate BANDWIDTH_RATE network bandwidth rate [K|M|G bit per second]

--delay NETWORK_LATENCY round trip network delay [ms]. the valid range is 0 to 3600000. (default=0)

--delay-distro LATENCY_DISTRO_MS distribution of network latency becomes X +- Y [ms] (normal distribution). Here X is the value of –delay option and Y is the value of –delay-dist option). network latency distribution will be uniform without this option.

--loss PACKET_LOSS_RATE round trip packet loss rate [%]. the valid range is 0 to 100. (default=0)

--corrupt CORRUPTION_RATE packet corruption rate [%]. the valid range is 0 to 100. packet corruption means single bit error at a random offset in the packet. (default=0)

--network NETWORK target IP address/network to control traffic

--port PORT target port number to control traffic.

--ipv6 apply traffic control to IPv6 packets rather than IPv4.

–shaping-algo {tbf,htb} shaping algorithm. defaults to htb (recommended).

Routing:

--iptables use iptables to traffic shaping.

--src-network SRC_NETWORK set traffic shaping rule to a specific packets that routed from –src-network to –network. This option required to execute with the –iptables option. the shaping rule only affect to outgoing packets (no effect to if you execute with "–direction incoming" option)

Basic usage

Examples of outgoing packet traffic control settings are as follows.

e.g. Set a limit on bandwidth up to 100Kbps

```
# tcset --device eth0 --rate 100k
```

e.g. Set 100ms network latency

```
# tcset --device eth0 --delay 100
```

e.g. Set 0.1% packet loss

```
# tcset --device eth0 --loss 0.1
```

e.g. All of the above at once

```
# tcset --device eth0 --rate 100k --delay 100 --loss 0.1
```

e.g. Specify the IP address of traffic control

```
# tcset --device eth0 --delay 100 --network 192.168.0.10
```

e.g. Specify the IP network and port of traffic control

```
# tcset --device eth0 --delay 100 --network 192.168.0.0/24 --port 80
```

Advanced usage

Traffic control of incoming packets

Execute `tcset` command with `--direction incoming` option to set incoming traffic control. Other options are the same as in the case of the basic usage.

e.g. Set traffic control both incoming and outgoing network

```
# tcset --device eth0 --direction outgoing --rate 200K --network 192.168.0.0/24
# tcset --device eth0 --direction incoming --rate 1M --network 192.168.0.0/24
```

Requirements

Incoming packet traffic control requires additional `ifb` module, Which need to the following conditions:

- Equal or later than Linux kernel version **2.6.20**
- Equal or later than `iproute2` package version **20070313**

Set latency destribution

Latency setting by `--delay` will be uniform-distribution. If you using `--delay-distro` option, latency will be decided by normal distribution.

e.g. Set 100ms +- 20ms network latency with normal distribution

```
# tcset --device eth0 --delay 100 --delay-distro 20
```

Multiple traffic shaping rules per interface

You can set multiple shaping rules to a network interface with `--add` option.

```
tcset --device eth0 --rate 500M --network 192.168.2.0/24
tcset --device eth0 --rate 100M --network 192.168.0.0/24 --add
```

Using IPv6

IPv6 addresses can be used at `tcset`/`tcshow` commands with `--ipv6` option.

```
# tcset --device eth0 --delay 100 --network 2001:db00::0/24 --ipv6
# tcshow --device eth0 --ipv6
{
    "eth0": {
        "outgoing": {
            "network=2001:db00::/24, protocol=ipv6": {
                "delay": "100.0",
                "rate": "1G"
            }
        },
        "incoming": {}
    }
}
```

Get `tc` commands

You can get `tc` commands to be executed by `tcconfing` commands by executing with `--tc-command` option (displayed commands are not actually executed).

```
# tcset --device eth0 --delay 10 --tc-command
tc qdisc add dev eth0 root handle 1f87: htb default 1
tc class add dev eth0 parent 1f87: classid 1f87:1 htb rate 1000000kbit
tc class add dev eth0 parent 1f87: classid 1f87:2 htb rate 1000000Kbit ceil␣
↪1000000Kbit
tc qdisc add dev eth0 parent 1f87:2 handle 2007: netem delay 10.0ms
tc filter add dev eth0 protocol ip parent 1f87: prio 1 u32 match ip dst 0.0.0.0/0␣
↪flowid 1f87:2
```

Generate a `tc` script file

You can generate a script file that described `tc` commands to be executed by `tcconfig` commands with `--tc-script` option. Created script can execute at other hosts where tcconfig is not installed but tc command is available.

```
# tcset --device eth0 --delay 10 --tc-script
[INFO] tcconfig: written a tc script to 'tcset_eth0.sh'
# ./tcset_eth0.sh
```

Delete traffic control (`tcdel` command)

`tcdel` is a command to delete traffic shaping rules from a network interface (device).

Note: `tcdel` will be deleted mangle tables in `iptables`. (any other tables are not affected).

`tcdel` command help

usage: tcdel [-h] [–version] [–tc-command | –tc-script] [–debug | –quiet] –device DEVICE

optional arguments:

-h, --help	show this help message and exit
--version	show program's version number and exit
--tc-command	display tc commands to be executed and exit. commands are not actually executed.
--tc-script	generate a script file that described tc commands to be executed by this command.
--debug	for debug print.
--quiet	suppress execution log messages.

Traffic Control:

--device DEVICE	network device name (e.g. eth0)

e.g. Delete traffic control of `eth0`

```
# tcdel --device eth0
```

Display traffic control configurations (`tcshow` command)

`tcshow` is a command to display traffic control to network interface(s).

Note: Scope of `tcshow` command is limited to parameters that can be set with tcset (i.e. `tcshow` is not a general purpose tool to display all of the parameters of the tc command).

`tcshow` command help

usage: tcshow [-h] [–version] [–tc-command | –tc-script] [–debug | –quiet] –device DEVICE [–ipv6]

optional arguments:

-h, --help	show this help message and exit
--version	show program's version number and exit

--tc-command	display tc commands to be executed and exit. commands are not actually executed.
--tc-script	generate a script file that described tc commands to be executed by this command.
--debug	for debug print.
--quiet	suppress execution log messages.

Traffic Control:

--device DEVICE	network device name (e.g. eth0)
--ipv6	Display IPv6 shaping rules. Defaults to show IPv4 shaping rules.

Example

```
# tcset --device eth0 --delay 10 --delay-distro 2  --loss 0.01 --rate 0.25M --network␣
↪192.168.0.10 --port 8080
# tcset --device eth0 --delay 1 --loss 0.02 --rate 500K --direction incoming
# tcshow --device eth0
{
    "eth0": {
        "outgoing": {
            "network=192.168.0.10/32, port=8080": {
                "delay": "10.0",
                "loss": "0.01",
                "rate": "250K",
                "delay-distro": "2.0"
            },
            "network=0.0.0.0/0": {}
        },
        "incoming": {
            "network=0.0.0.0/0": {
                "delay": "1.0",
                "loss": "0.02",
                "rate": "500K"
            }
        }
    }
}
```

Backup and restore traffic control configurations

`tcshow` command output can be used as a backup, and `tcset` command can restore configurations from a backup.

e.g. Backup configurations

```
# tcset --device eth0 --delay 10 --delay-distro 2  --loss 0.01 --rate 0.25M --network␣
↪192.168.0.10 --port 8080
# tcset --device eth0 --delay 1 --loss 0.02 --rate 500K --direction incoming
# tcset --device eth1 --delay 2.5 --delay-distro 1.2 --loss 0.01 --rate 0.25M --port␣
↪80
# tcset --device eth1 --corrupt 0.02 --rate 1.5M --direction incoming --network 192.
↪168.10.0/24
```

Redirect configurations to the `tcconfig.json` file.

```
# tcshow --device eth0 --device eth1 > tcconfig.json
```

e.g. Restore configurations

Before restore

```
# tcshow --device eth0 --device eth1
{
    "eth1": {
        "outgoing": {},
        "incoming": {}
    },
    "eth0": {
        "outgoing": {},
        "incoming": {}
    }
}
```

Restore from a configuration file (`tcconfig.json`).

```
# tcset -f tcconfig.json
```

After restore

```
# tcshow --device eth0 --device eth1
{
    "eth1": {
        "outgoing": {
            "port=80": {
                "delay": "2.5",
                "loss": "0.01",
                "rate": "250K",
                "delay-distro": "1.2"
            },
            "network=0.0.0.0/0": {}
        },
        "incoming": {
            "network=192.168.10.0/24": {
                "corrupt": "0.02",
                "rate": "1500K"
            },
            "network=0.0.0.0/0": {}
        }
    },
    "eth0": {
        "outgoing": {
            "network=192.168.0.10/32, port=8080": {
                "delay": "10.0",
                "loss": "0.01",
                "rate": "250K",
                "delay-distro": "2.0"
            },
            "network=0.0.0.0/0": {}
```

```
        },
        "incoming": {
            "network=0.0.0.0/0": {
                "delay": "1.0",
                "loss": "0.02",
                "rate": "500K"
            }
        }
    }
}
```

CHAPTER 5

Troubleshooting

Phenomenon

`tcset` command failed with an error message *RTNETLINK answers: No such file or directory*.

Solutions

The cause of this error is `sch_netem` kernel module is not loaded in your system. Execute the following command to solve this problem:

```
# modprobe sch_netem
```

The command is loading the `sch_netem` module. If the command failed with below message, you need to install additional kernel module.

```
# modprobe: FATAL: Module sch_netem not found in directory /lib/modules/xxxxxx
```

Execute the following command to install kernel modules (includes the *sch_netem* module).

```
# dnf install kernel-modules-extra
```

(in the case of *RHEL*/*CentOS*/*Fedora*). After that, re-execute *modprobe sch_netem* command.

```
# modprobe sch_netem
#
```

Phenomenon

`tcset` command with `--direction incoming` failed with an error message *RTNETLINK answers: Operation not supported*.

Solutions

The cause may be some mandatory kernel configurations are disabled. Following configurations are needed to be enabled to use `--direction incoming` option.

- CONFIG_IP_ADVANCED_ROUTER
- CONFIG_IP_MULTIPLE_TABLES
- CONFIG_NETFILTER_NETLINK
- CONFIG_NETFILTER_NETLINK_QUEUE
- CONFIG_NETFILTER_NETLINK_LOG
- CONFIG_NF_CT_NETLINK
- CONFIG_NETFILTER_XT_TARGET_MARK
- CONFIG_NET_SCHED
- CONFIG_NET_SCH_INGRESS
- CONFIG_SCSI_NETLINK

e.g. Display kernel configurations that enabled the above configurations (Debian)

```
cat /boot/config-3.16.0-4-amd64 | egrep "NETFILTER_NETLINK=|NETFILTER_NETLINK_
↪QUEUE=|NETFILTER_NETLINK_LOG=|NF_CT_NETLINK=|SCSI_NETLINK=|IP_ADVANCED_ROUTER=|NET_
↪SCH_INGRESS=|NET_SCHED=|IP_MULTIPLE_TABLES=|NETFILTER_XT_TARGET_MARK="
CONFIG_IP_ADVANCED_ROUTER=y
CONFIG_IP_MULTIPLE_TABLES=y
CONFIG_NETFILTER_NETLINK=m
CONFIG_NETFILTER_NETLINK_QUEUE=m
CONFIG_NETFILTER_NETLINK_LOG=m
CONFIG_NF_CT_NETLINK=m
CONFIG_NETFILTER_XT_TARGET_MARK=m
CONFIG_NET_SCHED=y
CONFIG_NET_SCH_INGRESS=m
CONFIG_SCSI_NETLINK=y
```

These configurations need to either `y` or `m`. If some of the configurations are disabled, you need to enable the configurations and recompile the kernel.

Note: Name of the */boot/config-3.16.0-4-amd64* will be changed depends on environment.

CHAPTER 6

Indices and tables

- genindex

CHAPTER 7

Links

- pip: A tool for installing python packages
- GitHub repository
- Issue tracker
- PyPI

CHAPTER 8

Indices and tables

- genindex