Simulating terrible network conditions

If you're an app developer reading this, can you tell me, off the top of your head, how your app behaves on a link with 40 kbps available bandwidth, 1,000 ms latency, occasional jitter of up to 2,000 ms, packet loss of 10%, and a complete 15-second connectivity dropout every few minutes?
— https://brr.fyi/posts/engineering-for-slow-internet

https://chatgpt.com/share/13e444bc-f16e-4ce7-841f-e8362f366da4

macOS

1. Create a configuration file for pfctl

```
sudo nano /etc/pf.conf:
dummynet-anchor "shaping"
anchor "shaping"
```

2. Create the dummynet rules

```
sudo nano /etc/pf.anchors/shaping:
dummynet out proto tcp from any to any pipe 1
dummynet out proto udp from any to any pipe 1

# Define the pipe with bandwidth, delay, and packet
loss
pipe 1 config bw 40Kbit/s delay 1000ms plr 0.1 queue
50
```

3. Enable pfctl and load the configuration

```
sudo pfctl -e
sudo pfctl -f /etc/pf.conf
sudo dnctl -f flush
sudo dnctl pipe 1 config bw 40Kbit/s delay 1000ms
plr 0.1
```

4. Simulate jitter and periodic connectivity dropouts

To simulate jitter and dropouts, you can use a looped script that periodically changes the conditions. Create a

script **network_simulation.sh**:

```
#!/usr/bin/env bash

while true; do
  # Simulate jitter
  sudo dnctl pipe 1 config bw 40Kbit/s delay $((1000
+ RANDOM % 2000))ms plr 0.1
  sleep 30

  # Simulate connectivity dropout
  sudo dnctl pipe 1 config bw 40Kbit/s delay 1000ms
plr 1
  sleep 15

  # Restore normal conditions
  sudo dnctl pipe 1 config bw 40Kbit/s delay 1000ms
plr 0.1
  sleep $(( RANDOM % 120 + 60 ))
done
```

5. Make the script executable and run it

```
chmod +x network_simulation.sh
./network_simulation.sh
```

This setup will simulate the specified network conditions, including bandwidth limitations, latency, jitter, packet loss, and periodic connectivity dropouts.

6. Restoring network conditions

```
#!/usr/bin/env bash

# Flush all dummynet configurations
sudo dnctl -f flush

# Disable pfctl
sudo pfctl -d

echo "Network conditions restored to normal."
```

Linux

To achieve the same network conditions on Linux, you can use **tc** (Traffic Control) along with netem for network emulation. Here's how to set it up:

1. Install necessary tools

```
sudo apt-get update
sudo apt-get install iproute2
```

2. Set up the basic network conditions

```
sudo tc qdisc add dev eth0 root handle 1: htb
default 30
sudo tc class add dev eth0 parent 1: classid 1:1 htb
rate 40kbps
sudo tc qdisc add dev eth0 parent 1:1 handle 10:
netem delay 1000ms loss 10%
Replace eth0 with your network interface name, which
you can find using ifconfig or ip a.
```

3. Simulate jitter and periodic connectivity dropouts

Create a script **network_simulation.sh**:

```
#!/usr/bin/env bash

while true; do
  # Simulate jitter
  sudo tc qdisc change dev eth0 root handle 1: htb
default 30
  sudo tc class change dev eth0 parent 1: classid
1:1 htb rate 40kbps
  sudo tc qdisc change dev eth0 parent 1:1 handle
10: netem delay $((1000 + RANDOM % 2000))ms loss 10%
  sleep 30

  # Simulate connectivity dropout
  sudo tc qdisc change dev eth0 root handle 1: htb
default 30
  sudo tc class change dev eth0 parent 1:1 htb rate
40kbps
```

```
  sudo tc qdisc change dev eth0 parent 1:1 handle
10: netem delay 1000ms loss 100%
  sleep 15

  # Restore normal conditions
  sudo tc qdisc change dev eth0 root handle 1: htb
default 30
  sudo tc class change dev eth0 parent 1:1 htb rate
40kbps
  sudo tc qdisc change dev eth0 parent 1:1 handle
10: netem delay 1000ms loss 10%
  sleep $(( RANDOM % 120 + 60 ))
done
```

Replace **eth0** with your network interface name, which you can find using **ifconfig** or **ip a**.

4. Make the script executable and run it

```
chmod +x network_simulation.sh
./network_simulation.sh
```

This script periodically changes the network conditions to include jitter and simulate complete connectivity dropouts as specified.

5. Restoring network conditions

```
#!/usr/bin/env bash

# Delete all tc configurations
sudo tc qdisc del dev eth0 root

echo "Network conditions restored to normal."
```

Replace **eth0** with your network interface name, which you can find using **ifconfig** or **ip a**.