10?GbE Network Tuning on Debian

Introduction

This document describes a small, focused kernel network tuning for Debian systems connected via 10 GbE, using a custom `/etc/sysctl.d/10g.conf` file and the standard `sysctl` mechanism to apply the settings. These values follow common recommendations for high-bandwidth Linux hosts by increasing TCP buffer limits and using modern congestion control and queuing disciplines.

Purpose of 10g.conf

The goal of `10g.conf` is to allow the TCP stack to efficiently fill a 10 Gb/s link for protocols like SMB and NFS while remaining conservative enough for general-purpose servers. It does this by:

- Increasing the maximum socket buffer sizes for receive and transmit.
- Raising the autotuning ceiling for TCP read and write buffers.
- Improving backlog and connection queue limits for busy hosts.
- Enabling modern congestion control (`htcp`) and fair queuing (`fq`).

10g.conf Content

Create `/etc/sysctl.d/10g.conf` with the following content:

```
# /etc/sysctl.d/10g.conf
# Basic 10 GbE TCP tuning for Debian / Linux.
# Focus: higher throughput for SMB/NFS and other bulk transfers over 10G.

# Allow larger TCP socket buffers for high-bandwidth links
net.core.rmem_max = 33554432
net.core.wmem_max = 33554432
net.core.rmem_default = 262144
net.core.wmem_default = 262144
```

```
# TCP autotuning limits: min, default, max
net.ipv4.tcp_rmem = 4096 87380 33554432
net.ipv4.tcp_wmem = 4096 65536 33554432

# Optional but often helpful for busy 10GbE hosts
net.core.netdev_max_backlog = 250000
net.core.somaxconn = 4096

# Use modern congestion control and fair queueing
net.ipv4.tcp_congestion_control = htcp
net.core.default_qdisc = fq
```

Parameter Notes

- `net.core.rmem_max` / `wmem_max`: Maximum per-socket buffer size; 32 MB is typical for 10 GbE hosts and is widely recommended.
- `tcp_rmem` / `tcp_wmem`: Trio of values (min, default, max) used by TCP autotuning; the higher max allows large windows on clean high-latency or high-bandwidth paths.
- `netdev_max_backlog`: Maximum number of packets that can be queued when the kernel receives them faster than it can process; 250 000 is a common safe value on modern hardware.
- `tcp_congestion_control = htcp`: Uses HighSpeed TCP Congestion Control, designed for fast long-fat-pipe links.
- `default_qdisc = fq`: Fair Queuing with pacing, often recommended for servers with modern kernels.

Applying the Configuration

Immediate Application

After saving `/etc/sysctl.d/10g.conf`, apply the settings to the running kernel:

```
sudo sysctl --system
```

This command reloads all configuration files under `/etc/sysctl.d`, `/run/sysctl.d` and `/usr/lib/sysctl.d`, and applies all settings without requiring a reboot.

Note: The settings take effect immediately for new connections. Existing long-lived TCP sessions may continue using their previous buffer sizes until they are re-established.

Verification

You can verify that the values have been applied by querying a few key parameters:

```
sysctl net.core.rmem_max
sysctl net.core.wmem_max
sysctl net.ipv4.tcp_rmem
sysctl net.ipv4.tcp_wmem
sysctl net.ipv4.tcp_congestion_control
sysctl net.core.default_qdisc
```

The output should reflect the values specified in `10g.conf`, confirming that the tuning is active.

Usage and Testing

Once the configuration is applied, you can re-test SMB/NFS throughput (for example, from your Nextcloud host or other Debian clients) using representative workloads or tools like `iperf3`, large file copies, or application-level benchmarks. The tuning primarily benefits sustained transfers where the network path was previously constrained by default TCP buffer limits rather than disk or CPU.

Revision #1
Created 2026-01-01 14:53:11 UTC
Updated 2026-01-01 14:55:59 UTC