Linux Tune Network Stack (Buffers Size) To Increase Networking Performance - nixCraft

I've two servers located in two different data center. Both server deals with a lot of concurrent large file transfers. But network performance is very poor for large files and performance degradation take place with a large files. How do I tune TCP under Linux to solve this problem?

By default the Linux network stack is not configured for high speed large file transfer across WAN links. This is done to save memory resources. You can easily tune Linux network stack by increasing network buffers size for high-speed networks that connect server systems to handle more network packets.

The default maximum Linux TCP buffer sizes are way too small. TCP memory is calculated automatically based on system memory; you can find the actual values by typing the following commands:

```
$ cat /proc/sys/net/ipv4/tcp_mem
```

The default and maximum amount for the receive socket memory:

```
$ cat /proc/sys/net/core/rmem_default
$ cat /proc/sys/net/core/rmem_max
```

The default and maximum amount for the send socket memory:

```
$ cat /proc/sys/net/core/wmem_default
$ cat /proc/sys/net/core/wmem_max
```

The maximum amount of option memory buffers:

```
$ cat /proc/sys/net/core/optmem_max
```

Tune values

Set the max OS send buffer size (wmem) and receive buffer size (rmem) to 12 MB for queues on all protocols. In other words set the amount of memory that is allocated for each TCP socket when it is opened or created while transferring files:

WARNING! The default value of rmem_max and wmem_max is about 128 KB in most Linux distributions, which may be enough for a low-latency general purpose network environment or for apps such as DNS / Web server. However, if the

latency is large, the default size might be too small. Please note that the following settings going to increase memory usage on your server.

```
# echo 'net.core.wmem_max=12582912' >> /etc/sysctl.conf
# echo 'net.core.rmem_max=12582912' >> /etc/sysctl.conf
```

You also need to set minimum size, initial size, and maximum size in bytes:

```
# echo 'net.ipv4.tcp_rmem= 10240 87380 12582912' >>
/etc/sysctl.conf
# echo 'net.ipv4.tcp_wmem= 10240 87380 12582912' >>
/etc/sysctl.conf
```

Turn on window scaling which can be an option to enlarge the transfer window:

```
# echo 'net.ipv4.tcp_window_scaling = 1' >> /etc/sysctl.conf
```

Enable timestamps as defined in RFC1323:

```
# echo 'net.ipv4.tcp_timestamps = 1' >> /etc/sysctl.conf
```

Enable select acknowledgments:

```
# echo 'net.ipv4.tcp_sack = 1' >> /etc/sysctl.conf
```

By default, TCP saves various connection metrics in the route cache when the connection closes, so that connections established in the near future can use these to set initial conditions. Usually, this increases overall performance, but may sometimes cause performance degradation. If set, TCP will not cache metrics on closing connections.

```
# echo 'net.ipv4.tcp_no_metrics_save = 1' >> /etc/sysctl.conf
```

Set maximum number of packets, queued on the INPUT side, when the interface receives packets faster than kernel can process them.

```
# echo 'net.core.netdev_max_backlog = 5000' >>
/etc/sysctl.conf
```

Now reload the changes:

```
# sysctl -p
```

Use tcpdump to view changes for eth0:

```
# tcpdump -ni eth0
```

Recommend readings:

- Please refer to kernel documentation in Documentation /networking/ip-sysctl.txt for more information.
- man page sysctl

The author is the creator of nixCraft and a seasoned sysadmin, DevOps engineer, and a trainer for the Linux operating system/Unix shell scripting. Get the **latest tutorials on SysAdmin, Linux/Unix and open source topics via RSS/XML feed** or weekly email newsletter.