Performance Tuning

CrossBox Server

Max Number of Workers

Increase the number of workers by setting Max Number of Workers to your total CPU count. This will allow CrossBox to utilize a multi-core processing. For example, if you have a server with 8 CPUs, set this to 8.

Remember that each worker also allocates 70-100 MB of RAM. Multiply this by a number of workers and you'll get the total amount of additional RAM which the change of this setting will require.

Linux Server

For a high-performance system trying to serve thousands of concurrent network clients, default Linux kernel parameters are often too low. Consider making following changes

- Increase max open files to 100,000 from the default (typically 1024). In Linux, every open network socket requires a file descriptor. Increasing this limit will ensure that lingering `TIME_WAIT` sockets and other consumers of file descriptors don’t impact our ability to handle lots of concurrent requests.
- Decrease the time that sockets stay in the `TIME_WAIT` state by lowering `tcp_fin_timeout` from its default of 60 seconds to 10. You can lower this even further, but too low, and you can run into socket close errors in networks with lots of jitter. We will also set `tcp_tw_reuse` to tell the kernel it can reuse sockets in the `TIME_WAIT` state.
- Increase the port range for ephemeral (outgoing) ports, by lowering the minimum port to 10000 (normally 32768), and raising the maximum port to 65000 (normally 61000). *Important:* This means you can’t have server software that attempts to bind to a port above 9999! If you need to bind to a higher port, say 10075, just modify this port range appropriately.
- Increase the read/write TCP buffers (`tcp_rmem` and `tcp_wmem`) to allow for larger window sizes. This enables more data to be transferred without ACKs, increasing throughput. We won’t tune the total TCP memory (`tcp_mem`), since this is automatically tuned based on available memory by Linux.
- Decrease the VM `swappiness` parameter, which discourages the kernel from swapping memory to disk. By default, Linux attempts to swap out idle processes fairly aggressively, which is counterproductive for long-running server processes that desire low latency.
- Increase the TCP congestion window, and disable reverting to TCP slow start after the connection is idle. By default, TCP starts with a single small segment, gradually increasing it by one each time. This results in unnecessary slowness that impacts the start of every request – which is especially bad for HTTP.

Kernel Parameters

To start, edit `/etc/sysctl.conf` and add these lines:

```
# /etc/sysctl.conf
# Increase system file descriptor limit
fs.file-max = 100000

# Discourage Linux from swapping idle processes to disk (default = 60)
vm.swappiness = 10

# Increase ephermeral IP ports
net.ipv4.ip_local_port_range = 10000 65000

# Increase Linux autotuning TCP buffer limits
# Set max to 16MB for 1GE and 32M (33554432) or 54M (56623104) for 10GE
# Don't set tcp_mem itself! Let the kernel scale it based on RAM.
net.core.rmem_max = 16777216
net.core.wmem_max = 16777216
net.core.rmem_default = 16777216
net.core.wmem_default = 16777216
net.core.optmem_max = 40960
net.ipv4.tcp_rmem = 4096 87380 16777216
net.ipv4.tcp_wmem = 4096 65536 16777216

# Make room for more TIME_WAIT sockets due to more clients,
# and allow them to be reused if we run out of sockets
```

```
# Also increase the max packet backlog
net.core.netdev_max_backlog = 50000
net.ipv4.tcp_max_syn_backlog = 30000
net.ipv4.tcp_max_tw_buckets = 2000000
net.ipv4.tcp_tw_reuse = 1
net.ipv4.tcp_fin_timeout = 10

# Disable TCP slow start on idle connections
net.ipv4.tcp_slow_start_after_idle = 0

# If your servers talk UDP, also up these limits
net.ipv4.udp_rmem_min = 8192
net.ipv4.udp_wmem_min = 8192

# Disable source routing and redirects
net.ipv4.conf.all.send_redirects = 0
net.ipv4.conf.all.accept_redirects = 0
net.ipv4.conf.all.accept_source_route = 0

# Log packets with impossible addresses for security
net.ipv4.conf.all.log_martians = 1
```

Since some of these settings can be cached by networking services, it's best to reboot to apply them properly (`sysctl -p` does not work reliably).

Open File Descriptors

In addition to the Linux `fs.file-max` kernel setting above, we need to edit a few more files to increase the file descriptor limits. The reason is the above just sets an absolute max, but we still need to tell the shell what our per-user session limits are.

So, first edit `/etc/security/limits.conf` to increase our session limits:

```
# /etc/security/limits.conf
# allow all users to open 100000 files
# alternatively, replace * with an explicit username
* soft nofile 100000
* hard nofile 100000
```

Next, `/etc/ssh/sshd_config` needs to make sure to use PAM:

```
# /etc/ssh/sshd_config
# ensure we consult pam
UsePAM yes
```

And finally, `/etc/pam.d/sshd` needs to load the modified `limits.conf`:

```
# /etc/pam.d/<g class="gr_ gr_808 gr-alert gr_spell gr_inline_cards gr_run_anim
ContextualSpelling ins-del multiReplace" id="808" data-gr-id="808">sshd</g>
# ensure pam includes our limits
session required pam_limits.so
```

You can confirm these settings have taken effect by opening a new ssh connection to the box and checking `ulimit`:

```
ulimit -n
100000
```

TCP Congestion Window

Finally, let's increase the TCP congestion window from 1 to 10 segments. This is done on the interface, which makes it a more manual process that our `sysctl` settings. First, use `ip route` to find the default route, shown in bold below:

```
route
default via 10.248.77.193 dev eth0 proto kernel
10.248.77.192/26 dev eth0  proto kernel  scope link  src 10.248.77.212
```

Copy that line, and paste it back to the `ip route change` command, adding `initcwnd 10` to the end to increase the congestion window:

```
route change default via 10.248.77.193 dev eth0 proto kernel initcwnd 10
```

To make this persistent across reboots, you'll need to add a few lines of bash like the following to a startup script somewhere. Often the easiest candidate is just pasting these lines into `/etc/rc.local`:

```
defrt=`ip route | grep "^default" | head -1`
ip route change $defrt initcwnd 10
```

Hardware

- More CPUs means more concurrency
- More than 4GB of RAM is often not required, even with high concurrency
- Faster Disk means faster IO, therefore we always recommend having an SSD

- Revision #3
- Created 8 years ago by Docs Admin
- Updated 8 years ago by Docs Admin