Vultr Docs (/Docs/)

Search articles   🔍

You could earn up to **$300** by adding new articles!

Get Started Now → **(/docs/submit/)**

How to Deploy Google BBR on CentOS 7

Published on: Thu, Jan 5, 2017 at 6:34 pm EST

CentOS (/docs/category/centos/)   **Linux Guides (/docs/category/linux-guides/)**

Networking (/docs/category/networking/)

System Admin (/docs/category/system-admin/)

BBR (Bottleneck Bandwidth and RTT) is a new congestion control algorithm which is contributed to the Linux kernel TCP stack by Google. With BBR in place, a Linux server can get significantly increased throughput and reduced latency for connections. Besides, it's easy to deploy BBR because this algorithm requires only updates on the sender side, not in the network or on the receiver side.

In this article, I will show you how to deploy BBR on a Vultr CentOS 7 KVM server instance.

Prerequisites

- A Vultr CentOS 7 x64 server instance.
- A sudo user (https://www.vultr.com/docs/how-to-use-sudo-on-debian-centos-and-freebsd).

Step 1: Upgrade the kernel using the ELRepo RPM repository

In order to use BBR, you need to upgrade the kernel of your CentOS 7 machine to 4.9.0. You can easily get that done using the ELRepo RPM repository.

Before the upgrade, you can take a look at the current kernel:

```
uname -r
```

This command should output a string which resembles:

```
3.10.0-514.2.2.el7.x86_64
```

As you see, the current kernel is 3.10.0.

Install the ELRepo repo:

```
sudo rpm --import https://www.elrepo.org/RPM-GPG-KEY-elrepo.org
sudo rpm -Uvh http://www.elrepo.org/elrepo-release-7.0-2.el7.elrepo.noarch.rpm
```

Install the 4.9.0 kernel using the ELRepo repo:

```
sudo yum --enablerepo=elrepo-kernel install kernel-ml -y
```

Confirm the result:

```
rpm -qa | grep kernel
```

If the installation is successful, you should see `kernel-ml-4.9.0-1.el7.elrepo.x86_64` among the output list:

```
kernel-ml-4.9.0-1.el7.elrepo.x86_64
kernel-3.10.0-514.el7.x86_64
kernel-tools-libs-3.10.0-514.2.2.el7.x86_64
kernel-tools-3.10.0-514.2.2.el7.x86_64
kernel-3.10.0-514.2.2.el7.x86_64
```

Now, you need to enable the 4.9.0 kernel by setting up the default grub2 boot entry.

Show all entries in the grub2 menu:

```
sudo egrep ^menuentry /etc/grub2.cfg | cut -f 2 -d \'
```

The result should resemble:

```
CentOS Linux 7 Rescue a0cbf86a6ef1416a8812657bb4f2b860 (4.9.0-1.el7.elrepo.x86_64)
CentOS Linux (4.9.0-1.el7.elrepo.x86_64) 7 (Core)
CentOS Linux (3.10.0-514.2.2.el7.x86_64) 7 (Core)
CentOS Linux (3.10.0-514.el7.x86_64) 7 (Core)
CentOS Linux (0-rescue-bf94f46c6bd04792a6a42c91bae645f7) 7 (Core)
```

Since the line count starts at `0` and the 4.9.0 kernel entry is on the second line, set the default boot entry as `1`:

```
sudo grub2-set-default 1
```

Reboot the system:

```
sudo shutdown -r now
```

When the server is back online, log back in and rerun the uname command to confirm that you are using the correct Kernel:

```
uname -r
```

You should see the result as below:

```
4.9.0-1.el7.elrepo.x86_64
```

Step 2: Enable BBR

In order to enable the BBR algorithm, you need to modify the `sysctl` configuration as follows:

```
echo 'net.core.default_qdisc=fq' | sudo tee -a /etc/sysctl.conf
echo 'net.ipv4.tcp_congestion_control=bbr' | sudo tee -a /etc/sysctl.conf
sudo sysctl -p
```

Now, you can use the following commands to confirm that BBR is enabled:

```
sudo sysctl net.ipv4.tcp_available_congestion_control
```

The output should resemble:

```
net.ipv4.tcp_available_congestion_control = bbr cubic reno
```

Next, verify with:

```
sudo sysctl -n net.ipv4.tcp_congestion_control
```

The output should be:

```
bbr
```

Finally, check that the kernel module was loaded:

```
lsmod | grep bbr
```

The output will be similar to:

```
tcp_bbr                16384  0
```

Step 3 (optional): Test the network performance enhancement

In order to test BBR's network performance enhancement, you can create a file in the web server directory for download, and then test the download speed from a web browser on your desktop machine.

```
sudo yum install httpd -y
sudo systemctl start httpd.service
sudo firewall-cmd --zone=public --permanent --add-service=http
sudo firewall-cmd --reload
cd /var/www/html
sudo dd if=/dev/zero of=500mb.zip bs=1024k count=500
```

Finally, visit the URL `http://[your-server-IP]/500mb.zip` from a web browser on your desktop computer, and then evaluate download speed.

That's all. Thank you for reading.

Want to contribute ?

You could earn up to **$300** by adding new articles!

<div>

Submit your Article (/docs/submit/)

Suggest an Update (/docs/suggest/?title=hov

Request an Article (/docs/request/)

</div>

Get started in the SSD Cloud!

mgshuffle@gmail.com

··········

Create Account

Terms of Service (/legal/tos/)   AUP / DMCA (/legal/use_policy/)

Privacy Policy (/legal/privacy/)

f   **y**

(https://twitter.com/Vultr/)(https://www.facebook.com/choosevultr/)