

Setting up a Linux cluster with Keepalived: Basic configuration

In this second of three Linux HA cluster articles, you'll explore the fundamentals of Keepalived installation and configuration.

Posted: March 25, 2020 | [Anthony Critelli](#) (Sudoer).



More Linux resources

- [Advanced Linux Commands Cheat Sheet for Developers](#)
- [Get Started with Red Hat Insights](#)
- [Download Now: Basic Linux Commands Cheat Sheet](#)
- [Linux System Administration Skills Assessment](#)

In the first article of this series, [Using Keepalived for managing simple failover in clusters](#), you learned about

Keepalived and the **VRRP** protocol for failing-over an IP address from one machine to another. Now it's time to get

We use cookies on our websites to deliver our online services. Details about how we use cookies and how you may disable them are set out in our [Privacy Statement](#). By using this website you agree to our use of cookies.



Keepalived installation

Keepalived is available within the standard package repositories and is easily installed using **yum**:

```
[root@server1 ~]# yum install -y keepalived

[root@server1 ~]# keepalived --version
Keepalived v2.0.10 (11/12,2018)

[root@server1 ~]# systemctl status keepalived
keepalived.service - LVS and VRRP High Availability Monitor
Loaded: loaded (/usr/lib/systemd/system/keepalived.service; disabled; vendor preset: disab
Active: inactive (dead)
```

You should also know how to compile **Keepalived** from source code. **Keepalived** is an actively maintained project, and it [regularly receives](#) new features and bug fixes that may not be in the package manager version when you need them. I even ran into bugs with the current version in the package repositories while writing this series, and I had to follow this exact procedure to use the latest version of **Keepalived**.

```
# Install prerequisites
yum install -y gcc openssl-devel

# Download the latest version of the code. Be sure to check the downloads page for the most
recent version:https://www.keepalived.org/download.html
[root@localhost ~]# wget https://www.keepalived.org/software/keepalived-2.0.20.tar.gz

# Extract the code
[root@localhost ~]# tar -xf keepalived-2.0.20.tar.gz

# Run the configure scripts
[root@localhost ~]# cd keepalived-2.0.20
[root@localhost keepalived-2.0.20]# ./configure

# Build and install keepalived
[root@localhost keepalived-2.0.20]# make
[root@localhost keepalived-2.0.20]# make install

# Test your installation
[root@localhost keepalived-2.0.20]# keepalived --version
Keepalived v2.0.20 (01/22,2020)
```

Basic configuration



server1 (VRRP Master)
eth0: 192.168.122.101
ens9: 192.168.122.15
eth0 VIP: 192.168.122.200



server2 (VRRP Backup)
ens9: 192.168.122.119
eth0: 192.168.122.102

Network symbols in the diagrams available via [VRT Network Equipment Extension](#), [CC BY-SA 3.0](#).

The configuration file for **Keepalived** is located at `/etc/keepalived/keepalived.conf`. As discussed in the previous article, **Keepalived** does more than just implement basic **VRRP**. This leads to a configuration file that might seem daunting if you look at the [Keepalived man page](#). However, a simple topology like the one above can be achieved with minimal configuration.

The most basic **Keepalived** configuration enables a shared IP address between two servers. In the above topology, server1 is the master, and server2 is the backup. The configuration for both servers is simple.

Server 1 configuration:

```
server1# cat /etc/keepalived/keepalived.conf
vrrp_instance VI_1 {
    state MASTER
    interface eth0
    virtual_router_id 51
    priority 255
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 12345
    }
    virtual_ipaddress {
        192.168.122.200/24
    }
}
```

Server 2 configuration:

```
server2# cat /etc/keepalived/keepalived.conf
vrrp_instance VI_1 {

    state BACKUP
    interface eth0
    virtual_router_id 51
    priority 254
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 12345
    }
    virtual_ipaddress {
        192.168.122.200/24
    }
}
```

The configuration directives should be obvious from their naming conventions, but I will walk through each one:

- **vrrp_instance** defines an individual instance of the **VRRP** protocol running on an interface. I have arbitrarily named this instance VI_1.
- **state** defines the initial state that the instance should start in.
- **interface** defines the interface that **VRRP** runs on.
- **virtual_router_id** is the unique identifier that you learned about in the first article of this series.
- **priority** is the advertised priority that you learned about in the first article of this series. As you will learn in the next article, priorities can be adjusted at runtime.
- **advert_int** specifies the frequency that advertisements are sent at (1 second, in this case).
- **authentication** specifies the information necessary for servers participating in **VRRP** to authenticate with each other. In this case, a simple password is defined.
- **virtual_ipaddress** defines the IP addresses (there can be multiple) that **VRRP** is responsible for.

If you're using a host-based firewall, such as **firewalld** or **iptables**, then you need to add the necessary rules to permit IP protocol 112 traffic. Otherwise, **Keepalived**'s advertisement method won't work. Configuring a host-based firewall is out of scope for this article, but be sure to check out some of Enable Sysadmin's other articles about [iptables](#) and [firewalld](#) for more information.

With the above configuration in place, you can start **Keepalived** on both servers using **systemctl start keepalived** and observe the IP addresses on each machine. Notice that server1 has started up as the **VRRP** master and owns the shared IP address (192.168.122.200), while server2's IP addresses remain unchanged:

```
server1# ip -brief address show
lo UNKNOWN 127.0.0.1/8 ::1/128
eth0 UP 192.168.122.101/24 192.168.122.200/24 fe80::5054:ff:fe82:d66e/64
```

```
server2# ip -br a
lo UNKNOWN 127.0.0.1/8 ::1/128
eth0 UP 192.168.122.102/24 fe80::5054:ff:fe04:2c5d/64
```

Once you've confirmed that **Keepalived** has started on both servers and server1 is the active master, you can test out failover functionality by "flipping" the VIP to the other server. By stopping **Keepalived** on server1, the active master stops sending out advertisements, and server2 takes over the VIP:

```
server1# systemctl stop keepalived
```

We use cookies on our websites to deliver our online services. Details about how we use cookies and how you may disable them are set out in our [Privacy Statement](#). By using this website you agree to our use of cookies.

```
lo UNKNOWN 127.0.0.1/8 ::1/128
```



```
eth0 UP 192.168.122.101/24 fe80::5054:ff:fe82:d66e/64

server2# ip -brief address show
lo UNKNOWN 127.0.0.1/8 ::1/128
eth0 UP 192.168.122.102/24 192.168.122.200/24 fe80::5054:ff:fe04:2c5d/64
```

And that’s it! You now have a basic pair of redundant servers.

Monitoring VRRP traffic

As discussed in the first article of the series, understanding the protocol-level behavior of **VRRP** is important so that you can properly configure and troubleshoot it. If you’ve read Enable Sysadmin’s previous articles about [analyzing network traffic](#), then you’re probably comfortable using **tcpdump**. Command line packet captures using **tcpdump** can reveal everything that you need to know about your **VRRP** configuration, including the VRID and priority of the active master:

```
server1# tcpdump proto 112
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode

listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes

16:51:01.353224 IP 192.168.122.102 > 224.0.0.18: VRRPv2, Advertisement, vrid 51, prio 254,
authype simple, intvl 1s, length 20

16:51:02.353673 IP 192.168.122.102 > 224.0.0.18: VRRPv2, Advertisement, vrid 51, prio 254,
authype simple, intvl 1s, length 20

16:51:03.353753 IP 192.168.122.102 > 224.0.0.18: VRRPv2, Advertisement, vrid 51, prio 254,
authype simple, intvl 1s, length 20

^C

3 packets captured
3 packets received by filter
0 packets dropped by kernel
```

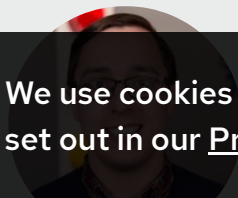
Try it out for yourself by running **tcpdump** while flipping the VIP back and forth between the two servers.

Wrapping up

This article took you through the fundamentals of **Keepalived** installation and configuration. You learned how to install **Keepalived** through the package manager and by compiling it from source, and you built a basic **Keepalived** configuration to enable VIP failover between two hosts. Finally, you tested out this configuration and used **tcpdump** to observe **VRRP** traffic. In the next article of this series, I will take you through some advanced **Keepalived** configurations.


[Need to learn more about Linux system administration? [Consider taking a Red Hat system administration course](#).]

Topics: [Systems architecture](#) [Linux](#)



Anthony Critelli

We use cookies on our websites to deliver our online services. Details about how we use cookies and how you may disable them are set out in our [Privacy Statement](#). By using this website you agree to our use of cookies.



Anthony Critelli is a Linux systems engineer with interests in automation, containerization, tracing, and performance. He started his professional career as a network engineer and eventually made the switch to the Linux