Red Hat

Venkata Kommaddi <vkommadi@redhat.com>

Forcing multicast packet iptable rule to use conntrack

6 messages

Anil Venkata <anilvenkata@redhat.com> Tue, Sep 8, 2020 at 1:46 PM
To: perf-dept <perf-dept@redhat.com>, rhos-dev <rh-openstack-dev@redhat.com>

Hi All

Need suggestions regarding iptables and connection tracking for multicast packets i.e force conntrack for multicast packet.

On the OSP13 (non-NFV) environment, a customer (nasdoc) reported a bug [1][2] about performance drop for multicast traffic. GSS team suggested enabling OVS IGMP snooping and disabling iptables. Customer confirmed he got better results but yet to share the performance numbers. Customer was using below iperf commands for multicast testing

server: iperf -s -u -B 239.73.175.100%eth0 -p 10000 -i 1 --window $((64*1024*1024))
client: iperf -c 239.73.175.100 -p 10000 -u -T 32 -t 100 -i 1 -b 126000000 --window $((64*1024*1024)) -l 105 -B 10.11.0.231

note: iperf client is running in a baremetal outside osp environment. Server is a VM (created on a VLAN using OSP provider networks) on compute node and each compute has at least 10 VMs. VM is connected to OVS switch (br-int) through a linux bridge and iptables firewall rules are applied on this linux bridge.

We tried to locally reproduce this issue with the same iperf commands. We see a packet drop of 5%-10% **with** iptables + OVS IGMP snooping i.e
[ID] Interval Transfer Bandwidth Jitter Lost/Total Datagrams
[3] 0.0-100.0 sec 1.38 GBytes 119 Mbits/sec 0.003 ms 874440/15000001 (5.8%)
[3] 0.0-100.0 sec 1.30 GBytes 111 Mbits/sec 0.003 ms 1741265/15000001 (12%)
This drop is inconsistent.

With OVS IGMP snooping and **disabling** iptables, the drop is in between 0.4% to 3%
[ID] Interval Transfer Bandwidth Jitter Lost/Total Datagrams
[3] 0.0-100.0 sec 1.46 GBytes 125 Mbits/sec 0.006 ms 80714/15000001 (0.54%)
[3] 0.0-100.0 sec 1.46 GBytes 125 Mbits/sec 0.007 ms 66786/15000001 (0.45%)
For disabling iptables, used below commands
1) iptables -F
1) echo 0 > /proc/sys/net/bridge/bridge-nf-call-iptables

As the customer can't disable iptables firewall, we need an optimization solution for handling multicast through iptables.

When we use iptables firewall, conntrack entry for this multicast traffic is staying in "UNREPLIED" state, forcing each packet to travel through all iptables rules.
conntrack v1.4.4 (conntrack-tools): 48 flow entries have been shown.
udp 17 29 src=10.11.0.231 dst=239.73.175.100 sport=42668 dport=10000 [UNREPLIED] src=239.73.175.100 dst=10.11.0.231 sport=10000 dport=42668 mark=0 secctx=system_u:object_r:unlabeled_t:s0 zone=4098 use=1

iptables -L neutron-openvswi-ibae3be42-9 -n -v --line-numbers
5 3799K 505M ACCEPT all -- * * 10.11.0.231 0.0.0.0/0 PKTTYPE = multicast
6 34M 4505M RETURN udp -- * * 10.11.0.231 0.0.0.0/0 udp multiport dports 10000:10001

Note: OSP used udp protocol for the iptables rule. I later manually created iptables rule with pkttype multicast.

Is there any way we can add iptables rule to force multicast packet to use conntrack and avoid iptables rules travel for each packet?

Also which tool does the perf team use for benchmarking multicast traffic? Do we have any benchmarking results or docs regarding multicast traffic?

[1] https://bugzilla.redhat.com/show_bug.cgi?id=1871650
[2] https://access.redhat.com/support/cases/#/case/02662779/

Thanks
Anil

Anil Venkata <anilvenkata@redhat.com> Tue, Sep 8, 2020 at 4:05 PM
To: perf-dept <perf-dept@redhat.com>, rhos-dev <rh-openstack-dev@redhat.com>, ovs-devel@redhat.com

+ovs-devel@redhat.com
[Quoted text hidden]

Aaron Conole <aconole@redhat.com> Tue, Sep 8, 2020 at 8:35 PM
To: Anil Venkata <anilvenkata@redhat.com>
Cc: perf-dept <perf-dept@redhat.com>, rhos-dev <rh-openstack-dev@redhat.com>, ovs-devel@redhat.com

[Quoted text hidden]
I think that's right. Unless the socket is local, I believe the
nf_conntrack code will not consider it 'replied' - but someone can
correct me if I'm mistaken.

> iptables -L neutron-openvswi-ibae3be42-9 -n -v --line-numbers
> 5 3799K 505M ACCEPT all -- * * 10.11.0.231 0.0.0.0/0 PKTTYPE = multicast
> 6 34M 4505M RETURN udp -- * * 10.11.0.231 0.0.0.0/0 udp multiport dports
> 10000:10001
>
> Note: OSP used udp protocol for the iptables rule. I later manually created iptables rule with pkttype multicast.
>
> Is there any way we can add iptables rule to force multicast packet to use conntrack and avoid iptables rules travel for
> each packet?

There are many ways to do that - you can mark multicast packets, and
jump to the multicast rules directly - would that work? There's also
xt_cluster to do hashing. I'm not sure what the desired flow of
traffic should look like so I can only throw out vague suggestions.

Is there a reason to continue to use iptables rather than converting to
OvS? Maybe that would be a better long-term solution?
[Quoted text hidden]

Jianzhu Zhang <jianzzha@redhat.com> Wed, Sep 9, 2020 at 12:05 AM
To: Aaron Conole <aconole@redhat.com>
Cc: Anil Venkata <anilvenkata@redhat.com>, perf-dept <perf-dept@redhat.com>, ovs-devel@redhat.com, rhos-dev <rh-openstack-dev@redhat.com>

Anil,

This is a very interesting test. I assume you don't really have a multicast capable router upstream in your test, so enabling the IGMP snooping probably won't really have an effect in your test. The performance gain you observed might be just due to the iptable removal. You can remove the IGMP snooping and see if the result changes.
[Quoted text hidden]

Anil Venkata <anilvenkata@redhat.com> Wed, Sep 9, 2020 at 4:19 PM
To: Jianzhu Zhang <jianzzha@redhat.com>, Sai Sindhur Malleni <smalleni@redhat.com>
Cc: Aaron Conole <aconole@redhat.com>, perf-dept <perf-dept@redhat.com>, ovs-devel@redhat.com, rhos-dev <rh-openstack-dev@redhat.com>

On Wed, Sep 9, 2020 at 12:05 AM Jianzhu Zhang <jianzzha@redhat.com> wrote:
> Anil,

> This is a very interesting test. I assume you don't really have a multicast capable router upstream in your test, so enabling the IGMP snooping probably won't really have an effect in your test. The performance gain you observed might be just due to the iptable removal. You can remove the IGMP snooping and see if the result changes.

Thanks Jainzhu. I don't have multicast capable router. In the below 2 tests "iptables" were enabled.
iptables + OVS IGMP snooping
[ID] Interval Transfer Bandwidth Jitter Lost/Total Datagrams
[3] 0.0-100.0 sec 1.34 GBytes 115 Mbits/sec 0.007 ms 1265774/15000000 (8.4%)

iptables + NO OVS IGMP snooping
[ID] Interval Transfer Bandwidth Jitter Lost/Total Datagrams
[3] 0.0-100.1 sec 110 MBytes 9.18 Mbits/sec 0.047 ms 13906041/15000001 (93%)

As we see we are losing 93% packets when OVS IGMP snooping disabled.
[Quoted text hidden]
Thanks Aaron. I have attached iptables-save output.
Neutron is creating a linux bridge for each vm and firewall (iptables) rules are programmed on this linux bridge. In 'raw' table's PREROUTING chain, it assigns CT zone for each linux bridge.
In 'filter' table's 'FORWARD' chain, neutron creates chains for each bridge interface and then adds firewall rules.

As you suggested I marked multicast packets and then added first rule in FORWARD chain to accept them.
iptables -R neutron-openvswi-ibae3be42-9 6 -s 10.11.0.231/32 -p udp -m udp -m multiport --dports 10000:10001 -j CONNMARK --set-mark 123
(note: we can aslo pkttype multicast in above rule)
iptables -I FORWARD 1 -t filter --match connmark --mark 123 -j ACCEPT

With this, the drop count has significantly reduced (i.e similar to deleting iptables)
[3] 0.0-100.0 sec 1.46 GBytes 125 Mbits/sec 0.007 ms 81632/15000001 (0.54%)

output of conntrack and iptables -
[root@compute-1 ~]# conntrack -L conntrack | grep 239.73
conntrack v1.4.4 (conntrack-tools): 96 flow entries have been shown.
udp 17 29 src=10.11.0.231 dst=239.73.175.100 sport=36205 dport=10000 packets=14733167 bytes=1959511211 [UNREPLIED] src=239.73.175.100 dst=10.11.0.231 sport=10000 dport=36205 packets=0 bytes=0 mark=123 secctx=system_u:object_r:unlabeled_t:s0 zone=4120 use=1

[root@compute-1 ~]# iptables -L neutron-openvswi-ibae3be42-9 -n -v --line-numbers
num pkts bytes target prot opt in out source destination
6 48M 6380M CONNMARK udp -- * * 10.11.0.231 0.0.0.0/0 udp multiport dports 10000:10001 CONNMARK set 0x7b
[root@compute-1 ~]# iptables -L FORWARD -n -v --line-numbers
num pkts bytes target prot opt in out source destination
1 861K 115M ACCEPT all -- * * 0.0.0.0/0 0.0.0.0/0 connmark match 0x7b

Hi Neutron team,
As neutron's security groups API doesn't provide the option to mark the packets, do you see any alternative approach for adding security group rules for multicast traffic for improving the performance?
I could reproduce the packet drop issue with few security group rules in my setup (i.e only 10 security group rules https://gist.github.com/venkataanil/c0a7bb9f64ede49ff62dfeb4bb7c8432). As customers will have large number of security group rules they would definitely see high drop count.

> Is there a reason to continue to use iptables rather than converting to OvS? Maybe that would be a better long-term solution?

We need to evaluate that as well.
[Quoted text hidden]

iptables_save
35K

Anil Venkata <anilvenkata@redhat.com> Wed, Sep 9, 2020 at 4:21 PM

To: Jianzhu Zhang <jianzzha@redhat.com>, Sai Sindhur Malleni <smalleni@redhat.com>, rh-osp-network <rh-osp-network@redhat.com>
Cc: Aaron Conole <aconole@redhat.com>, perf-dept <perf-dept@redhat.com>, ovs-devel@redhat.com, rhos-dev <rh-openstack-dev@redhat.com>

+rh-osp-network
[Quoted text hidden]