New issue

Jump to bottom

Large tailored UDP packets gives segmentation fault #235

Open **Assar-Westman** opened this issue on 17 Feb 2019 · 2 comments

Assar-Westman commented on 17 Feb 2019

Hi
I am really new to MoonGen and also Lua so maybe my problem is due to lack of knowledge. My system is a Ubuntu 18.04 server with Mellanox ConnectX-4 LX cards running at 25GB. I am trying to make a UDP packet generator for a device for which I only have the protocol at the moment. The protocol is quite simple but the UDP packets are large, the UDP packet length is 9562 bytes. I have crafted the UDP packet and I can also send packets with MoonGen as long as I do not try to put data in the pkt.payload above a 2kbyte, which than gives a segmentation fault. I have used the l3-load-latency.lua script as a boiler plate for my script. I should also mention when i do a buf:dump() I see that som junk values appears at adress 0x0820 and above (they are not zero)

What can I have done wrong?

Best regards Assar Westman

Assar-Westman changed the title ~~Large tailored UDP gives segmentation fault~~ **Large tailored UDP packets gives segmentation fault** on 17 Feb 2019

hopoaat commented on 20 Feb 2019

Hi,

The problem is that. MoonGen out of the box does not supports Jumbo frame.
I also met this problem when tested Jumbo frame(9000+). Payload filled with trash content.
The file that creates the mempool for you is the:
libmoon/lua/memory.lua

The questioned function, and arguments is the:
function mod.createMemPool(...)
args.bufSize = args.bufSize or 2048

By default the bufSize is 2048 only and that is too small for your needs.

You can pass arguments to the creator, or override the default value as example:
args.bufSize = args.bufSize or 9192

Note that, I only sent out max. pkt size as 9070 bytes with this modification, and did not touched the payload, it filled with zeros only(default values). You can of course finetune this size, for me it was a tryanderror thing.

Hopefully, this solves your problem too.
Make sure the Jumbo frame support is also enabled. /Earlier discussion mentioned how you can enable it for your NIC(s).
#165
/

Br,

Peter

emmericp commented on 21 Feb 2019

Jumbo frame support is still somewhat poor in MoonGen, sorry.
The main reason for this is that we are aiming at high packet rates (the thing that software packet generators were really bad at in the past).
Jumbo frames at 25G is just 330 kpps aka well in the range of iperf3.

With that excuse in mind: you can add it as a parameter to the mempool creation in the script you are using.

```
local mempool = memory.createMemPool{
        bufSize = 10000,
        func = function(buf) ... end
}
```

For receiving: the device configuration also takes a bufSize parameter that is passed on to the mempool created for receiving packets from the device.

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Linked pull requests

Successfully merging a pull request may close this issue.

None yet

3 participants