
# DPDK Based iperf for Layer 4 DPDK Applications / Load Balancers

**M JAY**

**PLATFORM APPLICATION ENGINEER**

# Agenda

DPDK
DATA PLANE DEVELOPMENT KIT

- The Benefit of DPDK-iperf
- Why do you need it?
- ANS – Accelerated Network Stack
- DPDK-iperf – without a switch in between the server and client
- DPDK-iperf – with a switch in between the server and client
- Call To Action

https://github.com/ansyun/dpdk-iperf

# L2fwd, L3fwd - Stateless

DPDK
DATA PLANE DEVELOPMENT KIT

- We have all measured DPDK l2fwd, l3fwd traffic
  - Hardware Traffic Generator
  - Software Traffic Generator
    - PktGen
    - Trex
    - Ostinato
- When we go up the stack – stateful generator needed
  - Question- Which stateful Software Traffic Generators you have used?

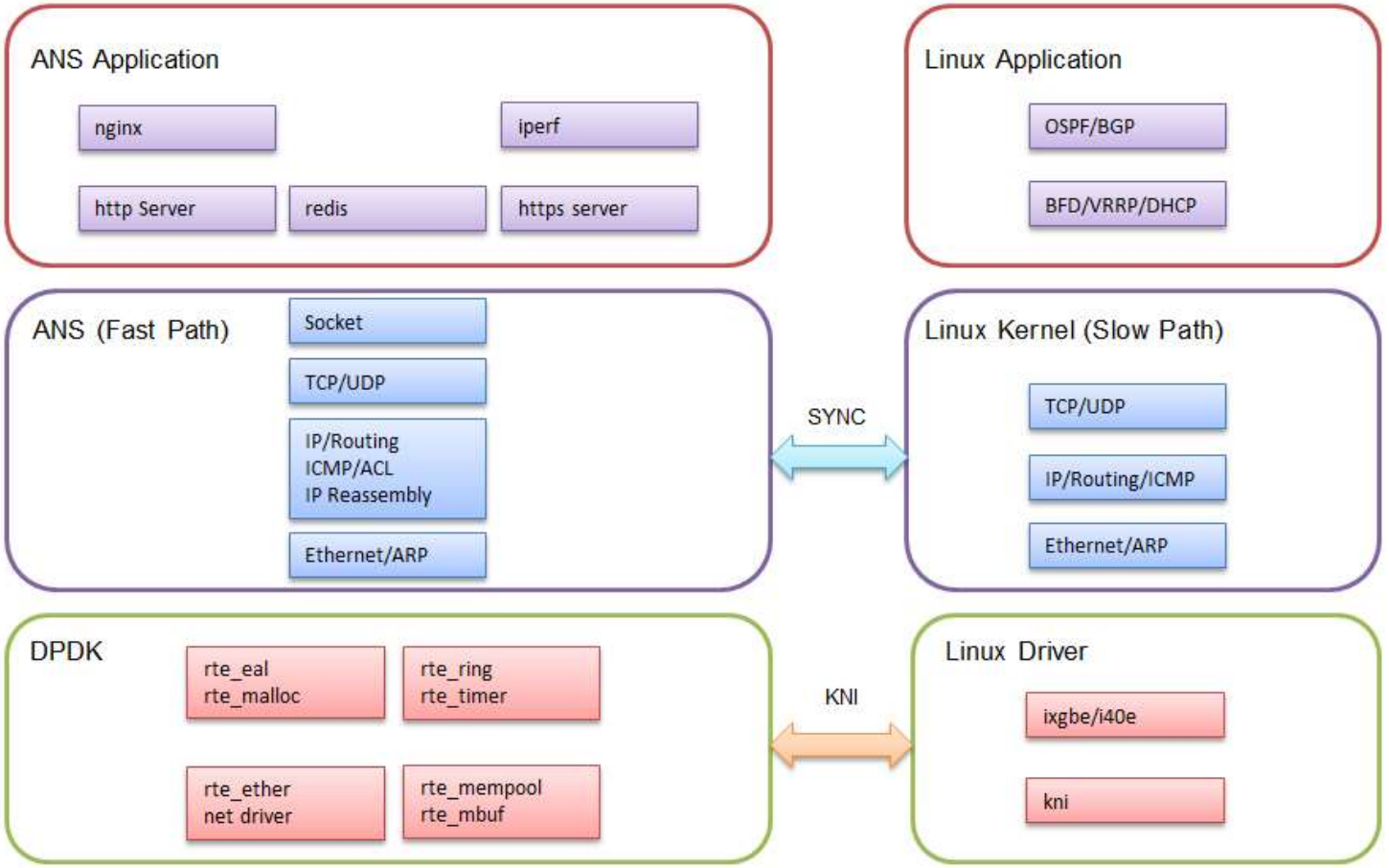
So, which stateless tool have you used?

# Kernel space tools – Bottleneck in the tool

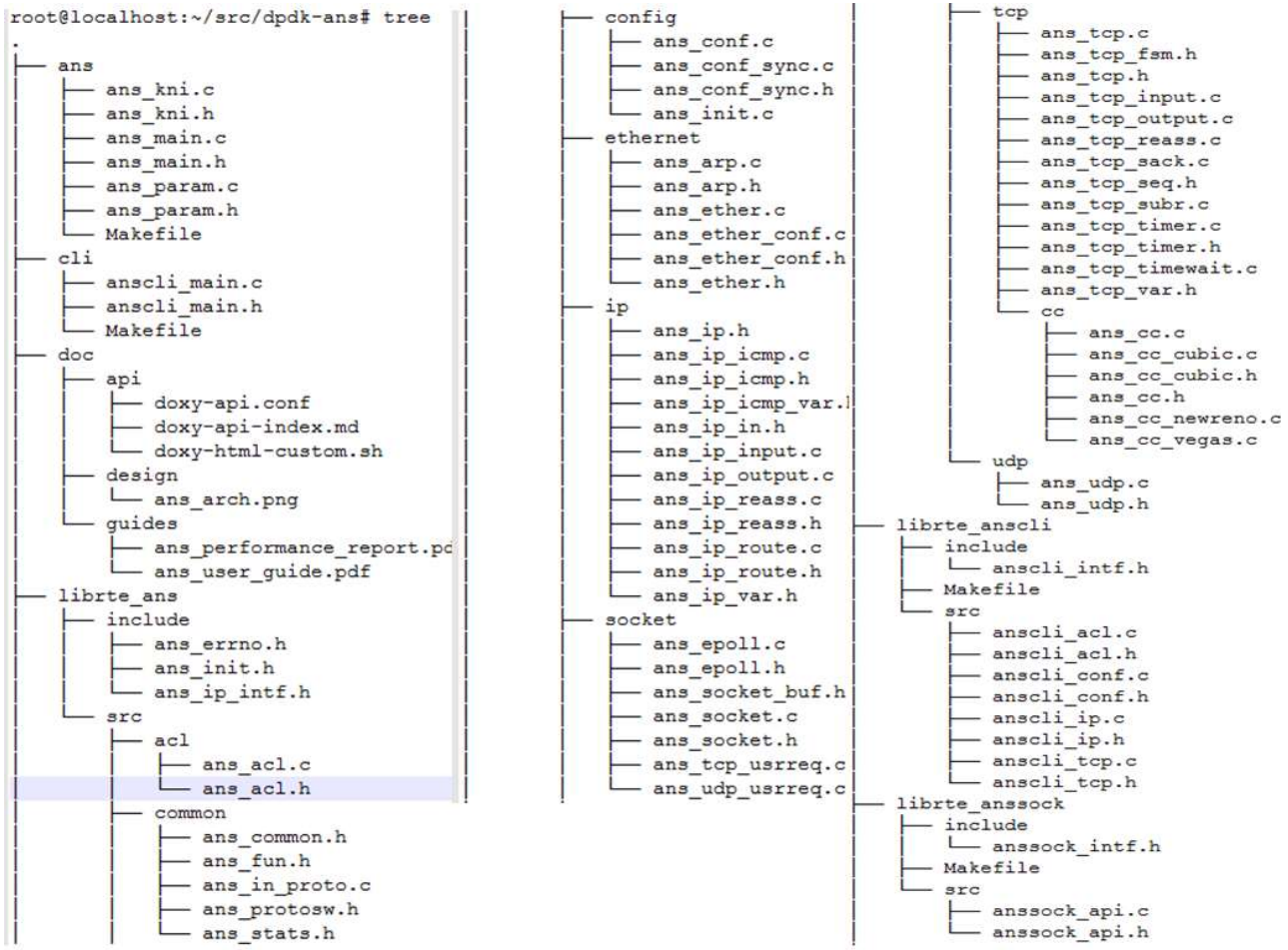
- Your DPDK based Layer 4 appliances
  - Layer 4 Load Balancer - DPVS
  - What other Layer 4 appliances you have built?
- What s/w generators to use?
- Kernel space generators
  - Most popular tools
    - Iperf
    - netperf
  - But, bottleneck in the tool for measuring user space appliances

So, which stateless tool do you recommend?

# DPDK-ANS Architecture



# ANS Tree


root@localhost:~/src/dpdk-ans# tree
.
├── ans
│   ├── ans_kni.c
│   ├── ans_kni.h
│   ├── ans_main.c
│   ├── ans_main.h
│   ├── ans_param.c
│   ├── ans_param.h
│   └── Makefile
├── cli
│   ├── anscli_main.c
│   ├── anscli_main.h
│   └── Makefile
├── doc
│   ├── api
│   │   ├── doxy-api.conf
│   │   ├── doxy-api-index.md
│   │   └── doxy-html-custom.sh
│   ├── design
│   │   └── ans_arch.png
│   └── guides
│       ├── ans_performance_report.pd
│       └── ans_user_guide.pdf
├── librte_ans
│   ├── include
│   │   ├── ans_errno.h
│   │   ├── ans_init.h
│   │   └── ans_ip_intf.h
│   └── src
│       ├── acl
│       │   ├── ans_acl.c
│       │   └── ans_acl.h
│       ├── common
│       │   ├── ans_common.h
│       │   ├── ans_fun.h
│       │   ├── ans_in_proto.c
│       │   ├── ans_protosw.h
│       │   └── ans_stats.h
│       ├── config
│       │   ├── ans_conf.c
│       │   ├── ans_conf_sync.c
│       │   ├── ans_conf_sync.h
│       │   └── ans_init.c
│       ├── ethernet
│       │   ├── ans_arp.c
│       │   ├── ans_arp.h
│       │   ├── ans_ether.c
│       │   ├── ans_ether_conf.c
│       │   ├── ans_ether_conf.h
│       │   └── ans_ether.h
│       ├── ip
│       │   ├── ans_ip.h
│       │   ├── ans_ip_icmp.c
│       │   ├── ans_ip_icmp.h
│       │   ├── ans_ip_icmp_var.
│       │   ├── ans_ip_in.h
│       │   ├── ans_ip_input.c
│       │   ├── ans_ip_output.c
│       │   ├── ans_ip_reass.c
│       │   ├── ans_ip_reass.h
│       │   ├── ans_ip_route.c
│       │   ├── ans_ip_route.h
│       │   └── ans_ip_var.h
│       ├── socket
│       │   ├── ans_epoll.c
│       │   ├── ans_epoll.h
│       │   ├── ans_socket_buf.h
│       │   ├── ans_socket.c
│       │   ├── ans_socket.h
│       │   ├── ans_tcp_usrreq.c
│       │   └── ans_udp_usrreq.c
│       ├── tcp
│       │   ├── ans_tcp.c
│       │   ├── ans_tcp_fsm.h
│       │   ├── ans_tcp.h
│       │   ├── ans_tcp_input.c
│       │   ├── ans_tcp_output.c
│       │   ├── ans_tcp_reass.c
│       │   ├── ans_tcp_sack.c
│       │   ├── ans_tcp_seq.h
│       │   ├── ans_tcp_subr.c
│       │   ├── ans_tcp_timer.c
│       │   ├── ans_tcp_timer.h
│       │   ├── ans_tcp_timewait.c
│       │   ├── ans_tcp_var.h
│       │   └── cc
│       │       ├── ans_cc.c
│       │       ├── ans_cc_cubic.c
│       │       ├── ans_cc_cubic.h
│       │       ├── ans_cc.h
│       │       ├── ans_cc_newreno.c
│       │       └── ans_cc_vegas.c
│       └── udp
│           ├── ans_udp.c
│           └── ans_udp.h
├── librte_anscli
│   ├── include
│   │   └── anscli_intf.h
│   ├── Makefile
│   └── src
│       ├── anscli_acl.c
│       ├── anscli_acl.h
│       ├── anscli_conf.c
│       ├── anscli_conf.h
│       ├── anscli_ip.c
│       ├── anscli_ip.h
│       ├── anscli_tcp.c
│       └── anscli_tcp.h
└── librte_anssock
    ├── include
    │   └── anssock_intf.h
    ├── Makefile
    └── src
        ├── anssock_api.c
        └── anssock_api.h

# Dpdk-iperf

- **Step 1:**
- Clone and Compile DPDK
- Git checkout stable Release
- Configure DPDK
- Install DPDK

```
[root@localhost dpdk-iperf]# #****************************************
[root@localhost dpdk-iperf]# # Clone and Compile DPDK
[root@localhost dpdk-iperf]# #****************************************
[root@localhost dpdk-iperf]# git clone http://dpdk.org/git/dpdk
Cloning into 'dpdk'...
remote: Counting objects: 152703, done.
remote: Compressing objects: 100% (25957/25957), done.
remote: Total 152703 (delta 126320), reused 152612 (delta 126249)
Receiving objects: 100% (152703/152703), 43.58 MiB | 7.48 MiB/s, done.
Resolving deltas: 100% (126320/126320), done.
[root@localhost dpdk-iperf]#
```

```
root@localhost dpdk-iperf]# #**************************
root@localhost dpdk-iperf]# # checkout LTS 18.11
root@localhost dpdk-iperf]# #**************************
root@localhost dpdk-iperf]# cd dpdk
root@localhost dpdk]# git checkout v18.11
lote: checking out 'v18.11'.

'ou are in 'detached HEAD' state. You can look around, make experimental
:hanges and commit them, and you can discard any commits you make in this
state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may
lo so (now or later) by using -b with the checkout command again. Example:

  git checkout -b new_branch_name

HEAD is now at 0da7f44... version: 18.11.0
```

```
[root@localhost dpdk]# #**************************************
[root@localhost dpdk]# # Configure DPDK
[root@localhost dpdk]# #**************************************
[root@localhost dpdk]#
```

```
[root@localhost dpdk]# make config T=x86_64-native-linuxapp-gcc
Configuration done using x86_64-native-linuxapp-gcc
```

```
dpdk]#
dpdk]# #****************************************
dpdk]# #Install DPDK
dpdk]# #****************************************
dpdk]# make install T=x86_64-native-linuxapp-gcc
```

# Accelerated Network Stack (ANS)

- Step 2:
- Set Env variables
- Git clone ANS

```
[root@localhost dpdk]# #*************************************
[root@localhost dpdk]# # Export env variables
[root@localhost dpdk]# #*************************************
[root@localhost dpdk]#
[root@localhost dpdk]# export RTE_SDK=$PWD
[root@localhost dpdk]# export RTE_TARGET=x86_64-native-linuxapp-gcc
```

```
[root@localhost dpdk]# #*******************************************
[root@localhost dpdk]# # Git clone ANS (Advanced Networking Stack
[root@localhost dpdk]# #*******************************************
[root@localhost dpdk]# cd ..
[root@localhost dpdk-iperf]# pwd
/home/user/dpdk-iperf
[root@localhost dpdk-iperf]# git clone https://github.com/ansyun/dpdk-ans.git
Cloning into 'dpdk-ans'...
remote: Enumerating objects: 92, done.
remote: Counting objects: 100% (92/92), done.
remote: Compressing objects: 100% (65/65), done.
remote: Total 2674 (delta 44), reused 56 (delta 27), pack-reused 2582
Receiving objects: 100% (2674/2674), 15.66 MiB | 17.20 MiB/s, done.
Resolving deltas: 100% (1513/1513), done.
```

```
[root@localhost dpdk-iperf]# ls -al
total 8
drwxr-xr-x.  4 root root   34 Feb 15 21:01 .
drwx------. 24 user user 4096 Feb 15 20:39 ..
drwxr-xr-x. 19 root root 4096 Feb 15 20:50 dpdk
drwxr-xr-x. 11 root root  180 Feb 15 21:01 dpdk-ans
```

# Install_deps.sh - accelerated Network Stack

- Step 3:
- Set env for ANS
- Install_deps.sh
- **Look for march related message**

```
[root@localhost dpdk-iperf]# #************************************
[root@localhost dpdk-iperf]# # Export env for ANS (Advanced Network Stack
[root@localhost dpdk-iperf]# #************************************
[root@localhost dpdk-iperf]# cd dpdk-ans/
[root@localhost dpdk-ans]# pwd
/home/user/dpdk-iperf/dpdk-ans
[root@localhost dpdk-ans]# export RTE_ANS=$PWD
[root@localhost dpdk-ans]#
[root@localhost dpdk-ans]# echo $RTE_ANS
/home/user/dpdk-iperf/dpdk-ans
```

```
[root@localhost dpdk-ans]# ls
ans  doc       install_deps.sh  librte_anscli   README.md
cli  examples  librte_ans       librte_anssock  test
[root@localhost dpdk-ans]#
```

```
[root@localhost dpdk-ans]# #**********************************
[root@localhost dpdk-ans]# # ./install_deps.sh
[root@localhost dpdk-ans]# #**********************************
[root@localhost dpdk-ans]# ./install_deps.sh
Start to generate librte_ans.a/librte_anssock.a/librte_anscli.a for -mnative
Your native march is core-avx2.
Your native march is none of westmere/sandybridge/ivybridge/haswell/broadwell/knl.
You may generate librte_ans.a/librte_anssock.a based on your -mnative by manual,
or ask help from ANS develop team.
core2:Intel Core 2 CPU with 64-bit extensions, MMX, SSE, SSE2, SSE3 and SSSE3 instruct
 support.
westmere:Intel Westmere CPU with 64-bit extensions, MMX, SSE, SSE2, SSE3, SSSE3, SSE4.
.2, POPCNT, AES and PCLMUL instruction set support.
sandybridge:Intel Sandy Bridge CPU with 64-bit extensions, MMX, SSE, SSE2, SSE3, SSSE3
1, SSE4.2, POPCNT, AVX, AES and PCLMUL instruction set support.
ivybridge:Intel Ivy Bridge CPU with 64-bit extensions, MMX, SSE, SSE2, SSE3, SSSE3, SS
SE4.2, POPCNT, AVX, AES, PCLMUL, FSGSBASE, RDRND and F16C instruction set support.
haswell:Intel Haswell CPU with 64-bit extensions, MOVBE, MMX, SSE, SSE2, SSE3, SSSE3,
```

```
broadwell:Intel Broadwell CPU with 64-bit extensions, MOVBE, MMX, SSE, SSE2, SSE3, SSSE3, SSE
4.1, SSE4.2, POPCNT, AVX, AVX2, AES, PCLMUL, FSGSBASE, RDRND, FMA, BMI, BMI2, F16C, RDSEED, A
DCX and PREFETCHW instruction set support.
knl:Intel Knight's Landing CPU with 64-bit extensions, MOVBE, MMX, SSE, SSE2, SSE3, SSSE3, SS
E4.1, SSE4.2, POPCNT, AVX, AVX2, AES, PCLMUL, FSGSBASE, RDRND, FMA, BMI, BMI2, F16C, RDSEED,
ADCX, PREFETCHW, AVX512F, AVX512PF, AVX512ER and AVX512CD instruction set support..
[root@localhost dpdk-ans]#
```

# DPDK-ANS – three libs, cli, ans executable

- Step 4: Build DPDK-ANS
- Lookout for 3 files
- Librte_ans.a
- Librte_anscli.a
- Librte_anssock.a

```
]# #*********************************
]# # list directories
]# #*********************************
]# ls -al

  200 Feb 15 21:11 .
   34 Feb 15 21:01 ..
  130 Feb 15 21:01 ans
   64 Feb 15 21:01 cli
   45 Feb 15 21:01 doc
   45 Feb 15 21:01 examples
  163 Feb 15 21:01 .git
  656 Feb 15 21:11 help-dummy.o
 5335 Feb 15 21:01 install_deps.sh
  194 Feb 15 21:01 librte_ans
  212 Feb 15 21:01 librte_anscli
  218 Feb 15 21:01 librte_anssock
12546 Feb 15 21:01 README.md
```

```
dpdk-ans]#
dpdk-ans]# cd ans
ans]# ls -al

root root   130 Feb 15 21:01 .
root root   200 Feb 15 21:11 ..
root root 15542 Feb 15 21:01 ans_kni.c
root root  2343 Feb 15 21:01 ans_kni.h
root root 31101 Feb 15 21:01 ans_main.c
root root  5048 Feb 15 21:01 ans_main.h
root root 12831 Feb 15 21:01 ans_param.c
root root  2017 Feb 15 21:01 ans_param.h
root root  2551 Feb 15 21:01 Makefile
```

```
[root@localhost ans]#
[root@localhost ans]# #*************************************************************
[root@localhost ans]# # build ANS (Advanced Networking Stack
[root@localhost ans]# #*************************************************************
[root@localhost ans]# make
  CC ans_param.o
  CC ans_main.o
  CC ans_kni.o
  LD ans
/bin/ld: cannot find /home/user/dpdk-iperf/dpdk-ans/librte_ans/librte_ans.a: No such file or
directory
collect2: error: ld returned 1 exit status
make[1]: *** [ans] Error 1
make: *** [all] Error 2
[root@localhost ans]#
```

# You should have installed binutils

- march – machine arch
- DPDK Prog. Manual
  - Second half sections
- Any suggestions?
- Brute force copy? – all 3 .a files
- Successful make !!

```
e_ans]# #*****************************************************
e_ans]# # By googling, one suggestion was to install binutils
e_ans]# #*****************************************************
e_ans]#
e_ans]# # apt-get update
e_ans]# # apt-get install build-essential
e_ans]# #*****************************************************
e ans]#
# #*****************************************************
# # since this is CentOS
# # yum update
# # yum install gcc gcc-c++ make
# # yum group install "Development Tools"
```

```
[root@localhost librte_ans]# cp librte_ans_sandybridge.a librte_ans.a
```

```
[root@localhost dpdk-ans]# cd ans
[root@localhost ans]# ls
ans_kni.c  ans_main.c  ans_param.c  build
ans_kni.h  ans_main.h  ans_param.h  Makefile
[root@localhost ans]# make
  LD ans
  INSTALL-APP ans
  INSTALL-MAP ans.map
[root@localhost ans]# make clean
[root@localhost ans]# make
  CC ans_param.o
  CC ans_main.o
  CC ans_kni.o
  LD ans
  INSTALL-APP ans
```

# Get Started To Run Accelerated Network Stack

- Step 5
- Check HugePage
- Bind Port of Interest
- Run Accelerated Network Stack

```
[root@localhost ans]# #****************************************
[root@localhost ans]# # 1) Check huge page
[root@localhost ans]# # 2) Bind the port of interest to igb_uio
[root@localhost ans]# #****************************************
[root@localhost ans]#
[root@localhost ans]# cat /proc/meminfo | grep Huge
AnonHugePages:    616448 kB
HugePages_Total:    1024
HugePages_Free:     1024
HugePages_Rsvd:        0
HugePages_Surp:        0
Hugepagesize:       2048 kB
[root@localhost ans]#
```

```
[root@localhost usertools]#
[root@localhost usertools]# modprobe uio
```

```
[root@localhost kmod]# insmod igb_uio.ko
[root@localhost kmod]# pwd
/home/user/dpdk-iperf/dpdk/x86_64-native-linuxapp-gcc/kmod
```

```
[root@localhost usertools]# ./dpdk-devbind.py --unbind 0000:17:00.1
```

```
#
# ./dpdk-devbind.py --bind=igb_uio 0000:17:00.1
#
```

```
#*************************************************************
# sudo ./build/ans -c 0x2 -n 1 -- -p 0x1 --config="(0,0,1)"
#*************************************************************
```

# Open Second Terminal & Run ANS CLI

- Step 6 – Leave first terminal as it is.
- Now open New terminal to start CLI. Why start CLI?
- Client and Server will have same IP address
- Need to change one of that. Hence CLI.

```
[root@localhost cli]#
[root@localhost cli]# #************************************
[root@localhost cli]# # Get anscli prompt
[root@localhost cli]# #************************************
[root@localhost cli]#
[root@localhost cli]# ./build/anscli
ans> help
ip addr add IFADDR dev STRING
ip addr del IFADDR dev STRING
ip addr show
ip route add DESTIP via NEXTHOP
ip route del DESTIP
ip route show
ip link show
ip neigh show
ip stats show
tcp stats show
udp stats show
```

```
show all IPs:

 veth0: mtu 1500
        link/ether 3c:fd:fe:c1:45:a8
        inet addr: 10.0.0.2/24

add static route

ANS IP routing table
 10.0.0.0/24 via dev veth0 src 10.0.0.2
 10.10.0.0/24 via 10.0.0.5 dev veth0


Checking link status done
Port 0 Link Up - speed 40000 Mbps - full-duplex
USER8: main loop on lcore 0
USER8:  -- lcoreid=0 portid=0 rxqueueid=0
hz: 2300000000
```

**Note:**

*** Use "ip addr show" to see IP address of server / client**

- **To change the IP Address – to keep addresses unique**
- **do "ip add del 10.0.0.2/24 dev veth0"**
- **Followed by "ip addr add 10.0.0.2/24 dev veth0"**
- **Verify with "ip addr show"**

# Open Third Terminal for dpdk-iperf

- Step 7:
- Git clone dpdk-iperf
- Make dpdk-iperf
- Start Server in one machine
- Start client in another machine

```
[root@localhost dpdk-iperf]# #*************************************
[root@localhost dpdk-iperf]# # make dpdk-iperf
[root@localhost dpdk-iperf]# #*************************************
[root@localhost dpdk-iperf]#  make dpdk-iperf
gcc -D_HAVE_DPDK_ANS_ -c -O3 -I/home/user/dpdk-iperf/dpdk-ans
 src/cjson.dpdk.o
gcc -D_HAVE_DPDK_ANS_ -c -O3 -I/home/user/dpdk-iperf/dpdk-ans
ent_api.c -o src/iperf_client_api.dpdk.o
gcc -D_HAVE_DPDK_ANS_ -c -O3 -I/home/user/dpdk-iperf/dpdk-ans
ale.c -o src/iperf_locale.dpdk.o
```

```
[user@localhost ~]$
[user@localhost ~]$ #************************************************
[user@localhost ~]$ # Open a new terminal
[user@localhost ~]$ # and run dpdk-iperf
[user@localhost ~]$ #************************************************
[user@localhost ~]$ sudo su -
[sudo] password for user:
Last login: Fri Feb 15 20:35:42 PST 2019 on pts/2
[root@localhost ~]# cd /home/user/dpdk-iperf/
[root@localhost dpdk-iperf]# ls
dpdk  dpdk-ans
[root@localhost dpdk-iperf]# git clone https://github.com/ansyun/dpdk-iperf.git
Cloning into 'dpdk-iperf'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 247 (delta 0), reused 1 (delta 0), pack-reused 243
Receiving objects: 100% (247/247), 628.16 KiB | 0 bytes/s, done.
Resolving deltas: 100% (118/118), done.
[root@localhost dpdk-iperf]#
```

```
[root@localhost dpdk-iperf]# ./dpdk_iperf3 -s --bind 10.0.0.2
EAL: Detected 16 lcore(s)
EAL: Detected 1 NUMA nodes
EAL: Multi-process socket /var/run/dpdk/rte/mp_socket_14033_29278183519b
EAL: Probing VFIO support...
USER8: LCORE[-1] anssock any lcore id 0xffffffff
USER8: LCORE[1] anssock app id: 14033
USER8: LCORE[1] anssock app name: dpdk_iperf3
USER8: LCORE[1] anssock app lcoreId: 1
USER8: LCORE[1] mp ops number 4, mp ops index: 0
USER8: LCORE[1] setsockopt: not support optname 2
-----------------------------------------------------------
Server listening on 5201
-----------------------------------------------------------
```

```
dpdk-iperf]# #************************************************
dpdk-iperf]# # Start dpdk-iperf client
dpdk-iperf]# #************************************************
[root@localhost dpdk-iperf]# ./dpdk_iperf3 -c 10.0.0.2
EAL: Detected 16 lcore(s)
EAL: Detected 1 NUMA nodes
EAL: Multi-process socket /var/run/dpdk/rte/mp_socket_13703_277df5bf4bab
EAL: Probing VFIO support...
USER8: LCORE[-1] anssock any lcore id 0xffffffff
USER8: LCORE[1] anssock app id: 13703
USER8: LCORE[1] anssock app name: dpdk_iperf3
USER8: LCORE[1] anssock app lcoreId: 1
USER8: LCORE[1] mp ops number 4, mp ops index: 0
fcntl(F_GETFL): Bad file descriptor
iperf3: error - unable to receive control message: Resource temporarily unavailable
[root@localhost dpdk-iperf]#
```

# Tips, Tricks

# Directly connection – No Switch – in between Server & Client

- Works Out Of The Box
- Default Server and Client will have same address
- Change IP address for one of them
- So they can be in same subnet each with its own unique IP address

- Delete the default IP address
- Add new IP address ***in same subnet and mask***

# Bad File Descriptor – How to take care of it?

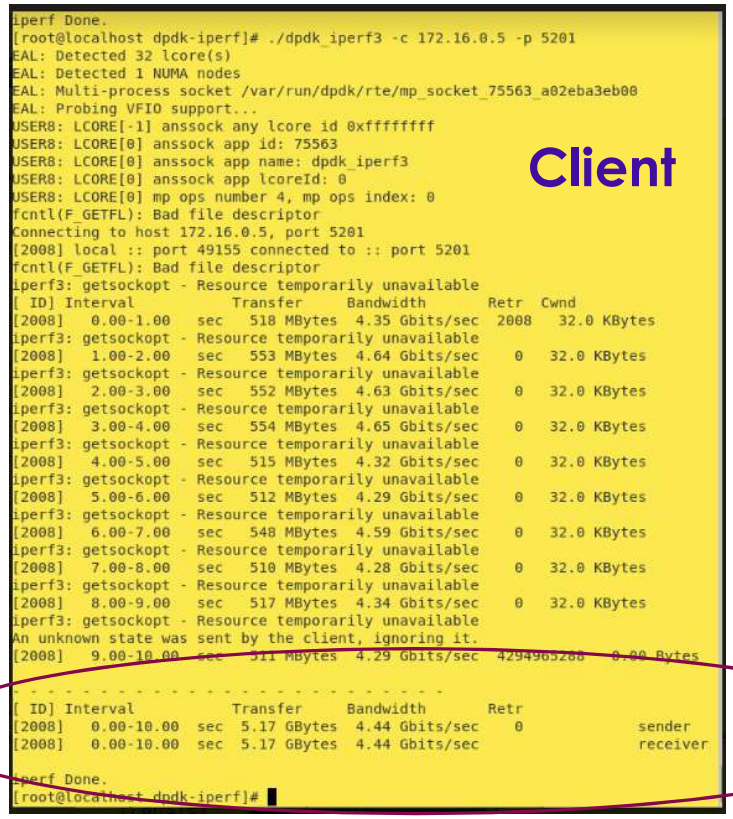
- 1) Make sure client & server are at different IP addr -
  - Through ANS CLI

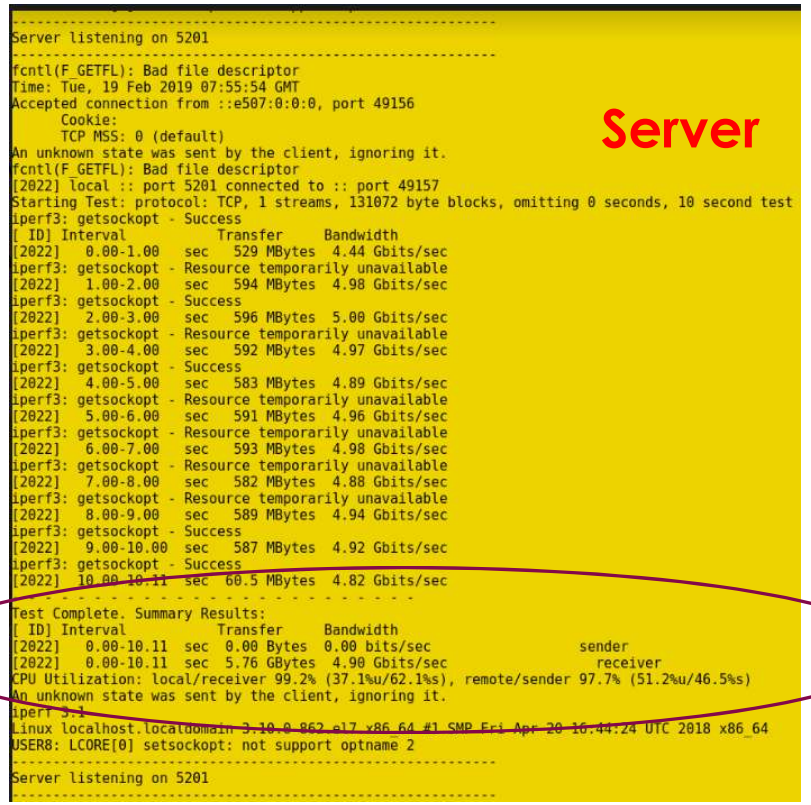
- 2) Works without switch in between.
- Suggestions To Make it work with Switch
  - Enable VLAN?ans_main.c
    - How?
    - In ans_main.c

```
.rxmode =
  {
   .mq_mode = ETH_MQ_RX_RSS,
   .max_rx_pkt_len = ETHER_MAX_LEN,
   .split_hdr_size = 0,
   .offloads = DEV_RX_OFFLOAD_CHECKSUM |
DEV_RX_OFFLOAD_VLAN_STRIP,
  },
```

# Dpdk-iperf output

# dpdk-iperf output




- Unidirectional - 5.76 Gbytes transfer
- 4.90 Gbits/sec bandwidth

# Acknowledgements


- https://github.com/ansyun/dpdk-iperf
- Excellent Support. Thanks to anssupport anssupport@163.com
  - Even on 3 days weekend – everyday got reply from support

# ANS library, ANS CLI and DPDK-IPerf

# Backup

# When You Have Switch in between Server & Client


- Enable VLAN
- Change code in ans_main.c

```
.rxmode =
 {
  .mq_mode = ETH_MQ_RX_RSS,
  .max_rx_pkt_len = ETHER_MAX_LEN,
  .split_hdr_size = 0,
  .offloads = DEV_RX_OFFLOAD_CHECKSUM | DEV_RX_OFFLOAD_VLAN_STRIP,
},
```


# Contact

M Jay
Muthurajan.Jayakumar@intel.com