iSEC
information security inc.

TRex

Information Security Inc.

Contents

- About Trex
- Current Challenges
- Implications
- Trex Stateful Features
- Trex Stateless Features
- Topology and the Testing Environment
- Installing TRex
- Using TRex
- References

About TRex

- TRex is a traffic generator for Stateful and Stateless use cases
- TRex -> realistic traffic generator

Current Challenges

- **Cost** : Commercial State-full traffic generators are expensive
- **Scale** : Bandwidth does not scale up well with features complexity
- **Standardization** : Lack of standardization of traffic patterns and methodologies
- **Flexibility** : Commercial tools do not allow agility when flexibility and changes are needed

Implications

- High capital expenditure (capex) spent by different teams
- Testing in low scale and extrapolation became a common practice, it is not accurate, and hides real life bottlenecks and quality issues
- TRex addresses these problems through an innovative and extendable software implementation and by leveraging standard and open SW and x86/UCS HW.

TRex Stateful Features

- Fueled by DPDK
- Generates and analyzes L4-7 traffic and able to provide in one tool capabilities provided by commercial L7 tools.
- Stateful traffic generator based on pre-processing and smart replay of real traffic templates.
- Generates and **amplifies** both client and server side traffic.
- Customized functionality can be added.
- Scale to 200Gb/sec for one UCS
- Low cost

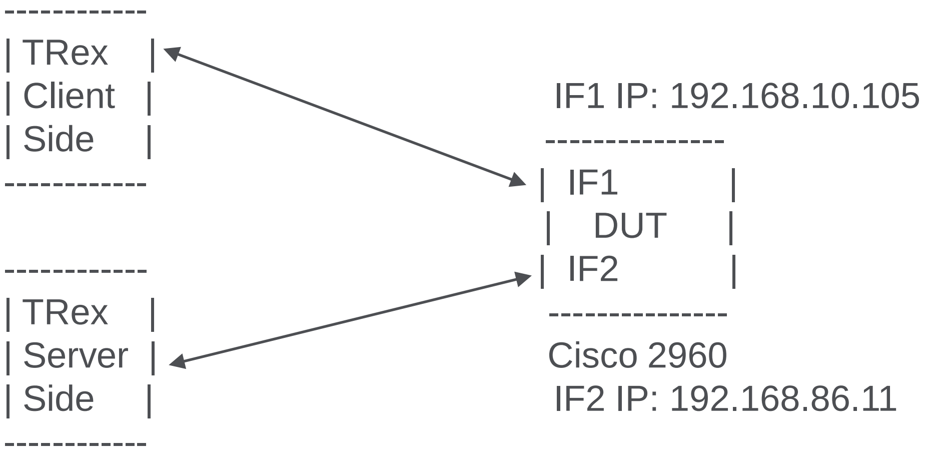
TRex Stateless Features

- Large scale - Supports about 10-30 million packets per second (mpps) per core, scalable with the number of cores
- Profile can support multiple streams, scalable to 10K parallel streams
- Interactive support - Fast Console, GUI
- Statistics per interface
- PCAP file import/export

Topology

- TRex testing topology -> TRex running in stateful mode
  -> TRex emulates/creates a internal router

Router IP: 192.168.10.188



TRex Client Side

IF1 IP: 192.168.10.105

IF1

DUT

IF2

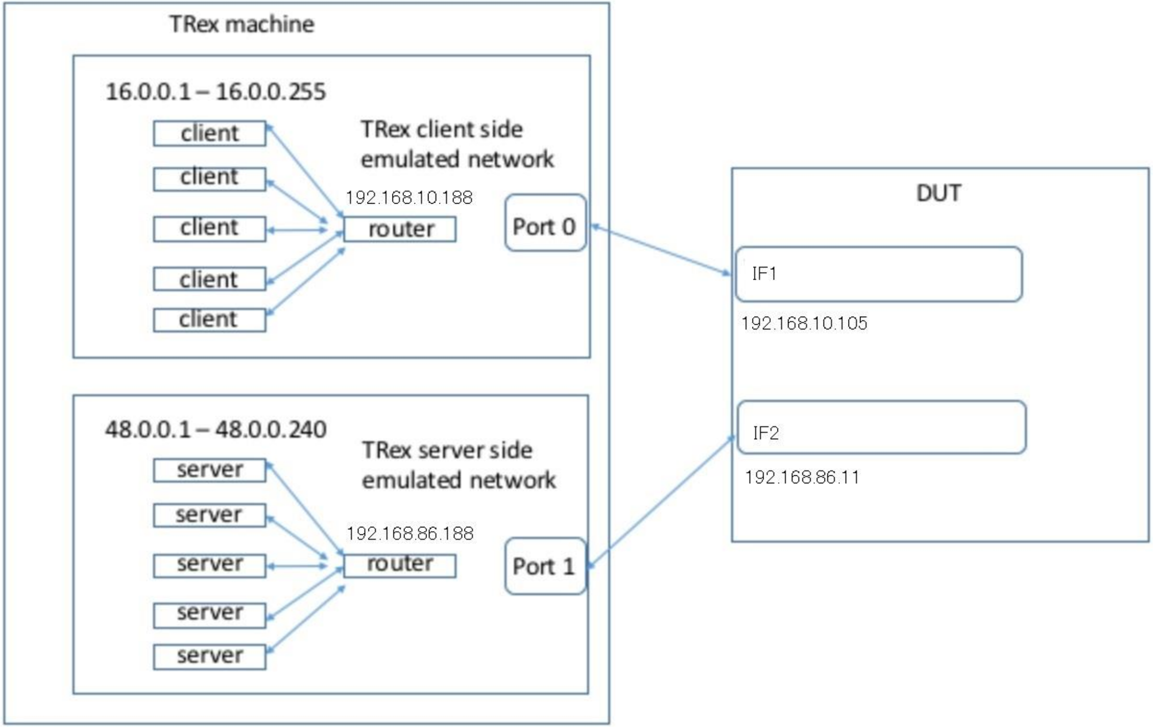
Cisco 2960

IF2 IP: 192.168.86.11

TRex Server Side

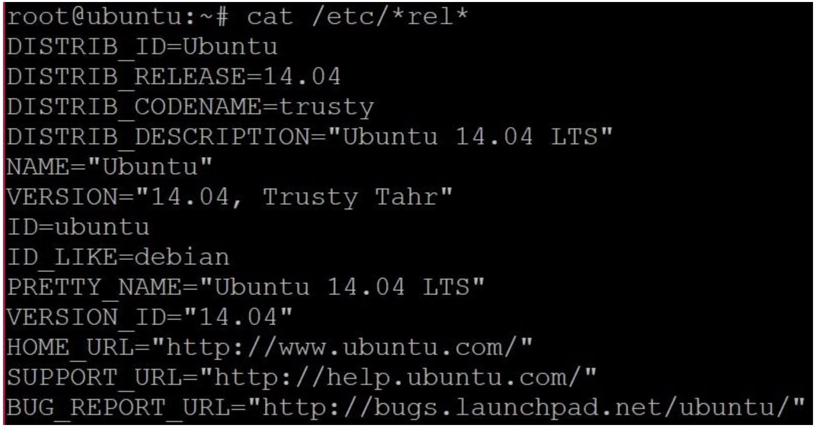
Router IP: 192.168.86.188

Topology


TRex machine
16.0.0.1 – 16.0.0.255
client
client
client
client
client
TRex client side emulated network
192.168.10.188
router
Port 0
48.0.0.1 – 48.0.0.240
server
server
server
server
server
TRex server side emulated network
192.168.86.188
router
Port 1
DUT
IF1
192.168.10.105
IF2
192.168.86.11

Testing Environment

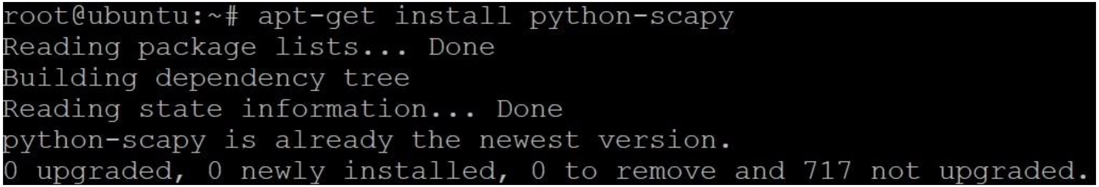
- Ubuntu 14.04 LTS



```
root@ubuntu:~# cat /etc/*rel*
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=14.04
DISTRIB_CODENAME=trusty
DISTRIB_DESCRIPTION="Ubuntu 14.04 LTS"
NAME="Ubuntu"
VERSION="14.04, Trusty Tahr"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 14.04 LTS"
VERSION_ID="14.04"
HOME_URL="http://www.ubuntu.com/"
SUPPORT_URL="http://help.ubuntu.com/"
BUG_REPORT_URL="http://bugs.launchpad.net/ubuntu/"
```

Installing TRex

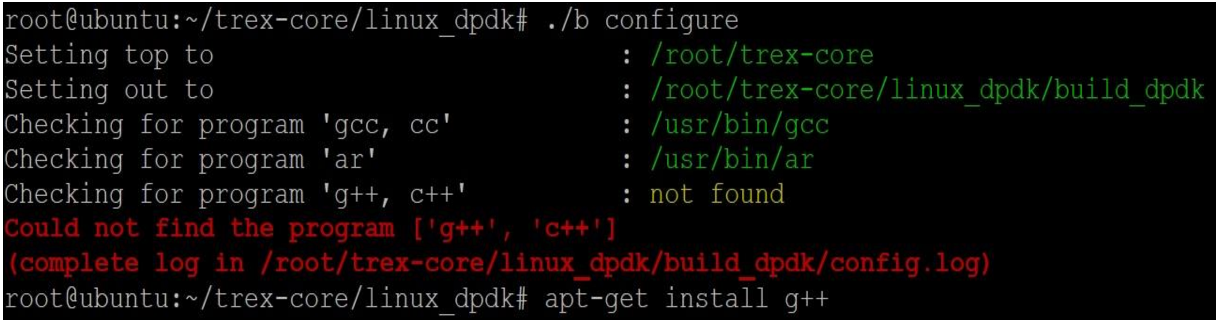
- Installing dependencies -> scapy



```
root@ubuntu:~# apt-get install python-scapy
Reading package lists... Done
Building dependency tree
Reading state information... Done
python-scapy is already the newest version.
0 upgraded, 0 newly installed, 0 to remove and 717 not upgraded.
```

Installing TRex

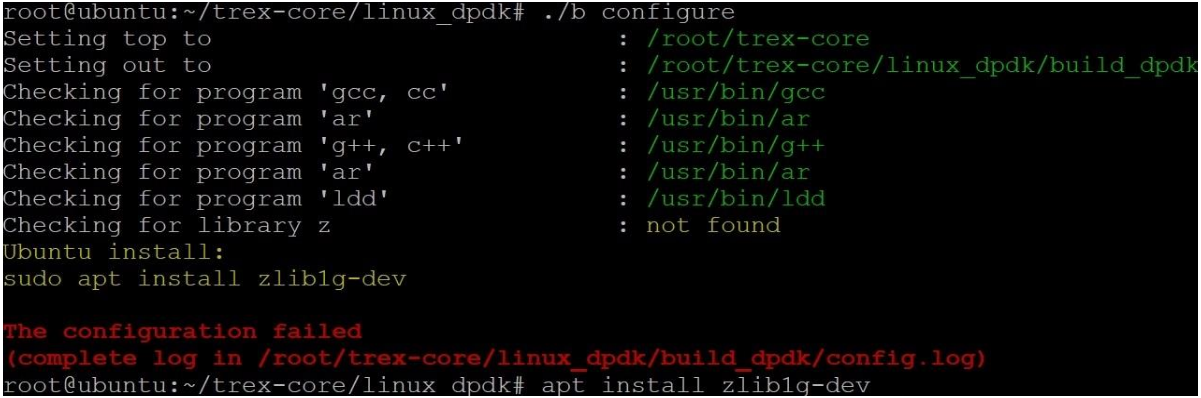
- Installing dependencies -> g++



```
root@ubuntu:~/trex-core/linux_dpdk# ./b configure
Setting top to                            : /root/trex-core
Setting out to                            : /root/trex-core/linux_dpdk/build_dpdk
Checking for program 'gcc, cc'            : /usr/bin/gcc
Checking for program 'ar'                 : /usr/bin/ar
Checking for program 'g++, c++'           : not found
Could not find the program ['g++', 'c++']
(complete log in /root/trex-core/linux_dpdk/build_dpdk/config.log)
root@ubuntu:~/trex-core/linux_dpdk# apt-get install g++
```

Installing TRex

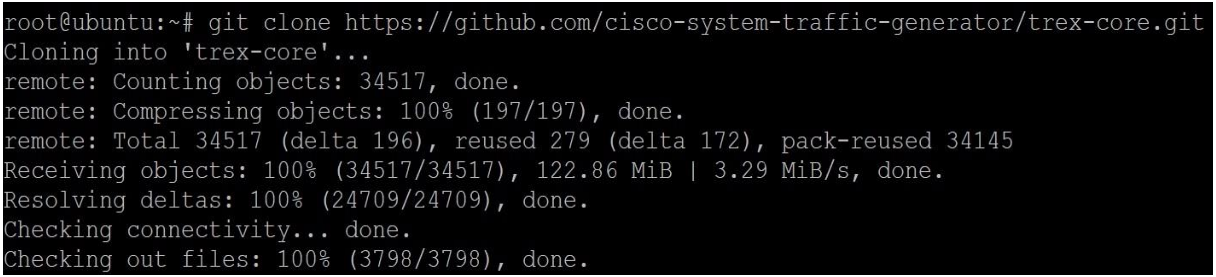
- Installing dependencies -> zlib1g-dev



```
root@ubuntu:~/trex-core/linux_dpdk# ./b configure
Setting top to                            : /root/trex-core
Setting out to                            : /root/trex-core/linux_dpdk/build_dpdk
Checking for program 'gcc, cc'            : /usr/bin/gcc
Checking for program 'ar'                 : /usr/bin/ar
Checking for program 'g++, c++'           : /usr/bin/g++
Checking for program 'ar'                 : /usr/bin/ar
Checking for program 'ldd'                : /usr/bin/ldd
Checking for library z                    : not found
Ubuntu install:
sudo apt install zlib1g-dev

The configuration failed
(complete log in /root/trex-core/linux_dpdk/build_dpdk/config.log)
root@ubuntu:~/trex-core/linux_dpdk# apt install zlib1g-dev
```

Installing TRex

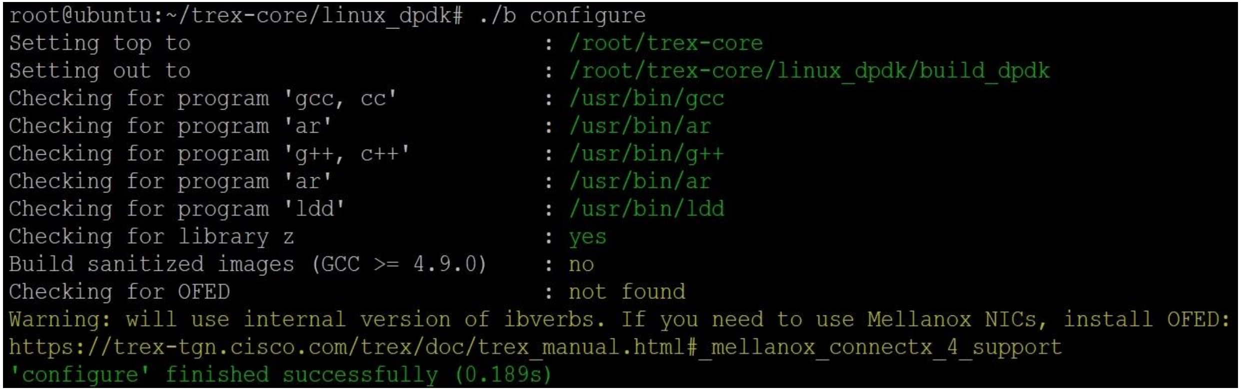
- Cloning GitHub repository



```
root@ubuntu:~# git clone https://github.com/cisco-system-traffic-generator/trex-core.git
Cloning into 'trex-core'...
remote: Counting objects: 34517, done.
remote: Compressing objects: 100% (197/197), done.
remote: Total 34517 (delta 196), reused 279 (delta 172), pack-reused 34145
Receiving objects: 100% (34517/34517), 122.86 MiB | 3.29 MiB/s, done.
Resolving deltas: 100% (24709/24709), done.
Checking connectivity... done.
Checking out files: 100% (3798/3798), done.
```

Installing TRex

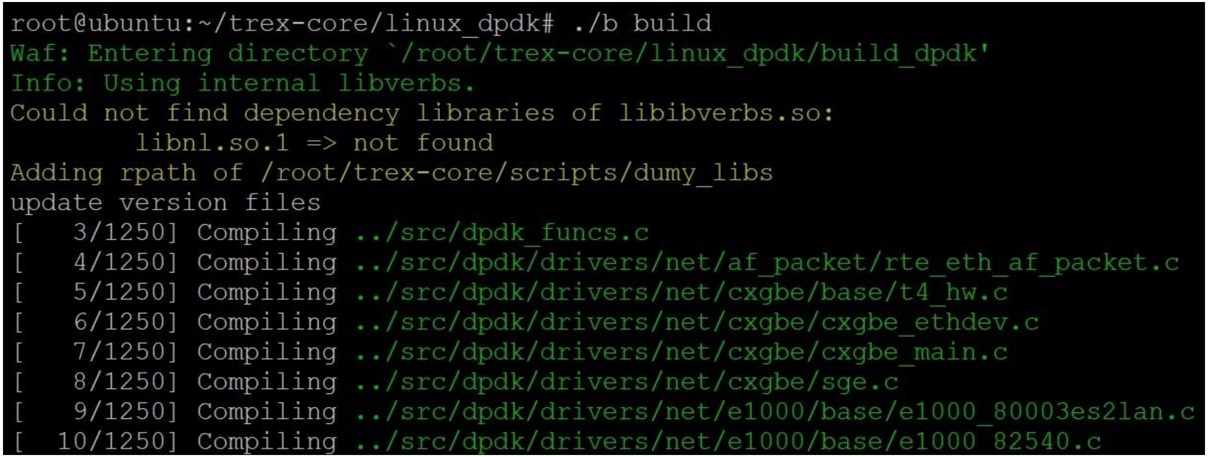
- Configuring TRex -> ./b configure



```
root@ubuntu:~/trex-core/linux_dpdk# ./b configure
Setting top to                           : /root/trex-core
Setting out to                           : /root/trex-core/linux_dpdk/build_dpdk
Checking for program 'gcc, cc'           : /usr/bin/gcc
Checking for program 'ar'                : /usr/bin/ar
Checking for program 'g++, c++'          : /usr/bin/g++
Checking for program 'ar'                : /usr/bin/ar
Checking for program 'ldd'               : /usr/bin/ldd
Checking for library z                   : yes
Build sanitized images (GCC >= 4.9.0)    : no
Checking for OFED                        : not found
Warning: will use internal version of ibverbs. If you need to use Mellanox NICs, install OFED:
https://trex-tgn.cisco.com/trex/doc/trex_manual.html#_mellanox_connectx_4_support
'configure' finished successfully (0.189s)
```

Installing TRex

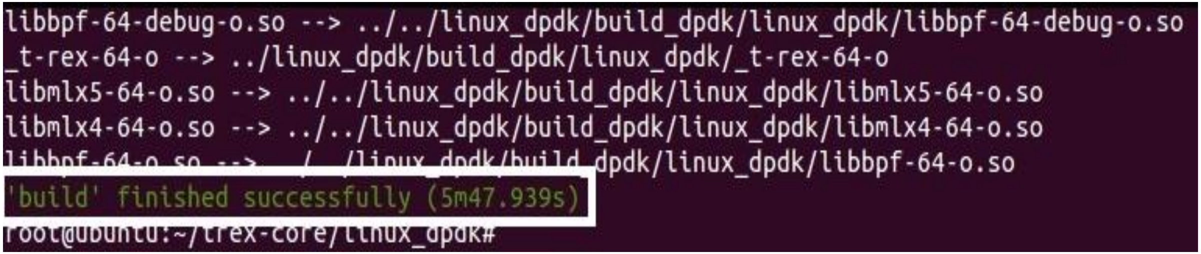
- Building Trex -> ./b build



```
root@ubuntu:~/trex-core/linux_dpdk# ./b build
Waf: Entering directory `/root/trex-core/linux_dpdk/build_dpdk'
Info: Using internal libverbs.
Could not find dependency libraries of libibverbs.so:
        libnl.so.1 => not found
Adding rpath of /root/trex-core/scripts/dumy_libs
update version files
[   3/1250] Compiling ../src/dpdk_funcs.c
[   4/1250] Compiling ../src/dpdk/drivers/net/af_packet/rte_eth_af_packet.c
[   5/1250] Compiling ../src/dpdk/drivers/net/cxgbe/base/t4_hw.c
[   6/1250] Compiling ../src/dpdk/drivers/net/cxgbe/cxgbe_ethdev.c
[   7/1250] Compiling ../src/dpdk/drivers/net/cxgbe/cxgbe_main.c
[   8/1250] Compiling ../src/dpdk/drivers/net/cxgbe/sge.c
[   9/1250] Compiling ../src/dpdk/drivers/net/e1000/base/e1000_80003es2lan.c
[  10/1250] Compiling ../src/dpdk/drivers/net/e1000/base/e1000_82540.c
```

Installing TRex

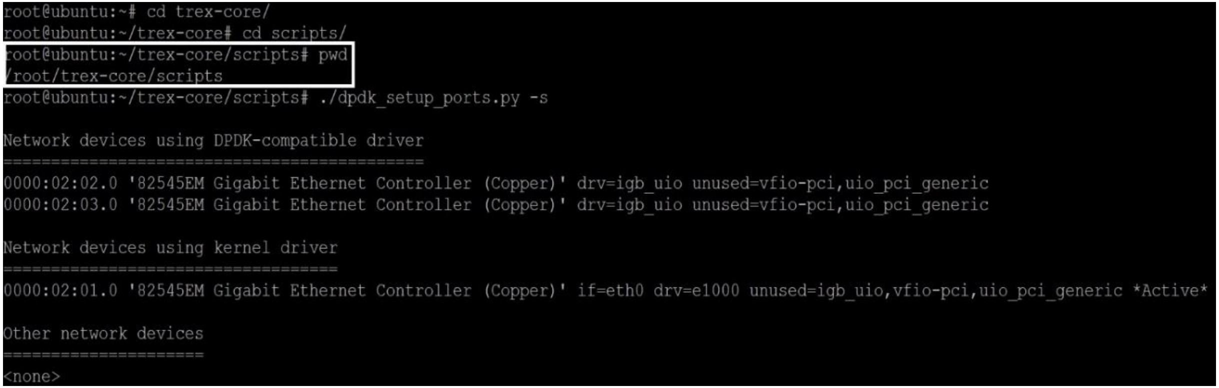
- Building Trex -> ./b build



```
libbpf-64-debug-o.so --> ../../linux_dpdk/build_dpdk/linux_dpdk/libbpf-64-debug-o.so
_t-rex-64-o --> ../linux_dpdk/build_dpdk/linux_dpdk/_t-rex-64-o
libmlx5-64-o.so --> ../../linux_dpdk/build_dpdk/linux_dpdk/libmlx5-64-o.so
libmlx4-64-o.so --> ../../linux_dpdk/build_dpdk/linux_dpdk/libmlx4-64-o.so
libbpf-64-o.so --> ../../linux_dpdk/build_dpdk/linux_dpdk/libbpf-64-o.so
'build' finished successfully (5m47.939s)
root@ubuntu:~/trex-core/linux_dpdk#
```

Using TRex

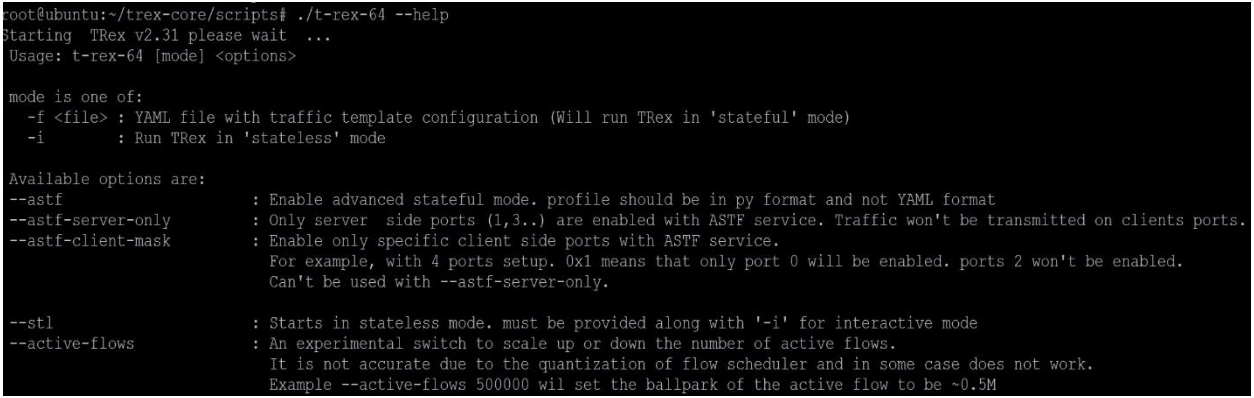
- Identify the ports



```
root@ubuntu:~# cd trex-core/
root@ubuntu:~/trex-core# cd scripts/
root@ubuntu:~/trex-core/scripts# pwd
/root/trex-core/scripts
root@ubuntu:~/trex-core/scripts# ./dpdk_setup_ports.py -s

Network devices using DPDK-compatible driver
============================================
0000:02:02.0 '82545EM Gigabit Ethernet Controller (Copper)' drv=igb_uio unused=vfio-pci,uio_pci_generic
0000:02:03.0 '82545EM Gigabit Ethernet Controller (Copper)' drv=igb_uio unused=vfio-pci,uio_pci_generic

Network devices using kernel driver
===================================
0000:02:01.0 '82545EM Gigabit Ethernet Controller (Copper)' if=eth0 drv=e1000 unused=igb_uio,vfio-pci,uio_pci_generic *Active*

Other network devices
=====================
<none>
```

Using TRex

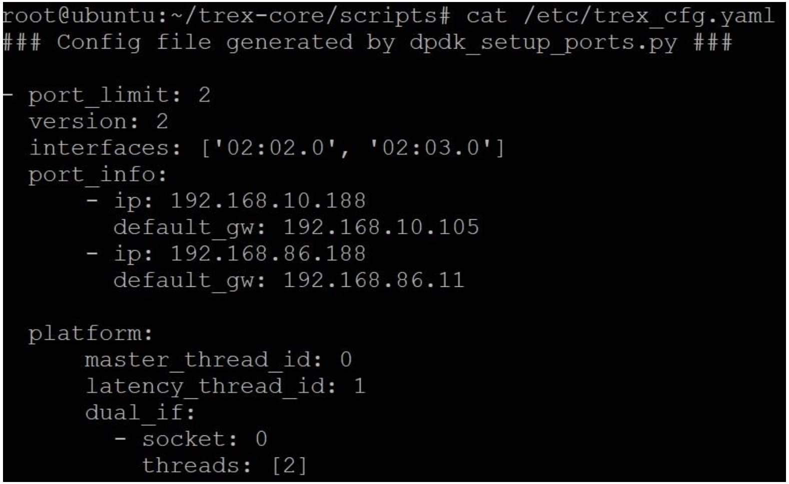
- TRex help menu



```
root@ubuntu:~/trex-core/scripts# ./t-rex-64 --help
Starting  TRex v2.31 please wait  ...
 Usage: t-rex-64 [mode] <options>

 mode is one of:
   -f <file> : YAML file with traffic template configuration (Will run TRex in 'stateful' mode)
   -i        : Run TRex in 'stateless' mode

 Available options are:
 --astf                     : Enable advanced stateful mode. profile should be in py format and not YAML format
 --astf-server-only         : Only server  side ports (1,3..) are enabled with ASTF service. Traffic won't be transmitted on clients ports.
 --astf-client-mask         : Enable only specific client side ports with ASTF service.
                              For example, with 4 ports setup. 0x1 means that only port 0 will be enabled. ports 2 won't be enabled.
                              Can't be used with --astf-server-only.

 --stl                      : Starts in stateless mode. must be provided along with '-i' for interactive mode
 --active-flows             : An experimental switch to scale up or down the number of active flows.
                              It is not accurate due to the quantization of flow scheduler and in some case does not work.
                              Example --active-flows 500000 wil set the ballpark of the active flow to be ~0.5M
```

Using TRex

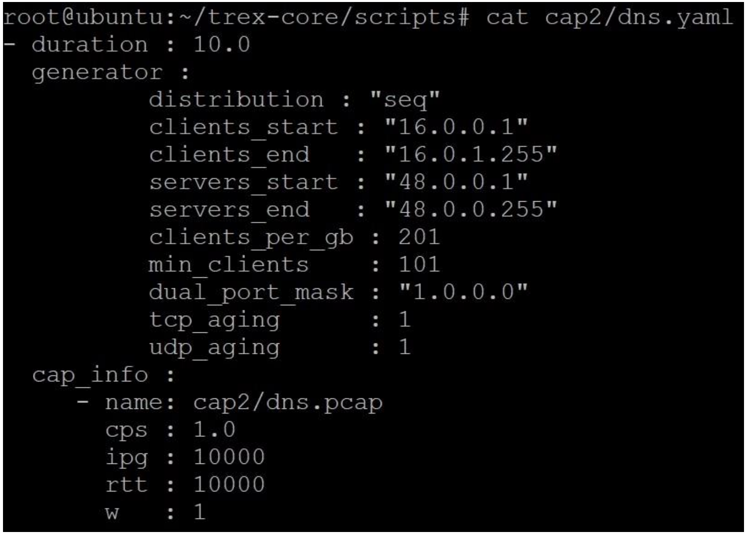
- Default config file -> /etc/trex_cfg.yaml



```
root@ubuntu:~/trex-core/scripts# cat /etc/trex_cfg.yaml
### Config file generated by dpdk_setup_ports.py ###

- port_limit: 2
  version: 2
  interfaces: ['02:02.0', '02:03.0']
  port_info:
      - ip: 192.168.10.188
        default_gw: 192.168.10.105
      - ip: 192.168.86.188
        default_gw: 192.168.86.11

  platform:
      master_thread_id: 0
      latency_thread_id: 1
      dual_if:
        - socket: 0
          threads: [2]
```

Using TRex

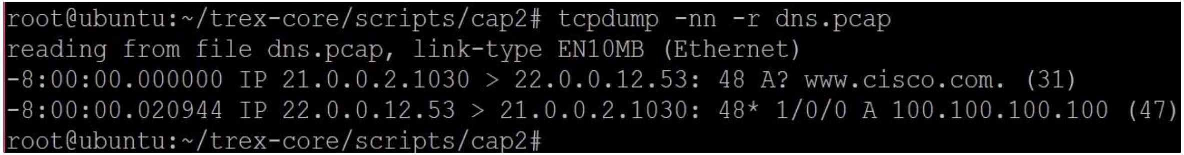
- Traffic config file -> in this example 255 clients talking to 240 servers



```
root@ubuntu:~/trex-core/scripts# cat cap2/dns.yaml
- duration : 10.0
  generator :
          distribution : "seq"
          clients_start : "16.0.0.1"
          clients_end   : "16.0.1.255"
          servers_start : "48.0.0.1"
          servers_end   : "48.0.0.255"
          clients_per_gb : 201
          min_clients    : 101
          dual_port_mask : "1.0.0.0"
          tcp_aging      : 1
          udp_aging      : 1
  cap_info :
     - name: cap2/dns.pcap
       cps : 1.0
       ipg : 10000
       rtt : 10000
       w   : 1
```

Using TRex

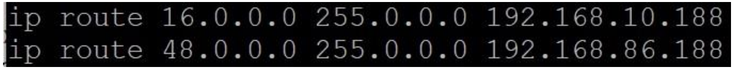
- Pcap file contents -> dns.pca, query for www.cisco.ip



```
root@ubuntu:~/trex-core/scripts/cap2# tcpdump -nn -r dns.pcap
reading from file dns.pcap, link-type EN10MB (Ethernet)
-8:00:00.000000 IP 21.0.0.2.1030 > 22.0.0.12.53: 48 A? www.cisco.com. (31)
-8:00:00.020944 IP 22.0.0.12.53 > 21.0.0.2.1030: 48* 1/0/0 A 100.100.100.100 (47)
root@ubuntu:~/trex-core/scripts/cap2#
```

Using TRex

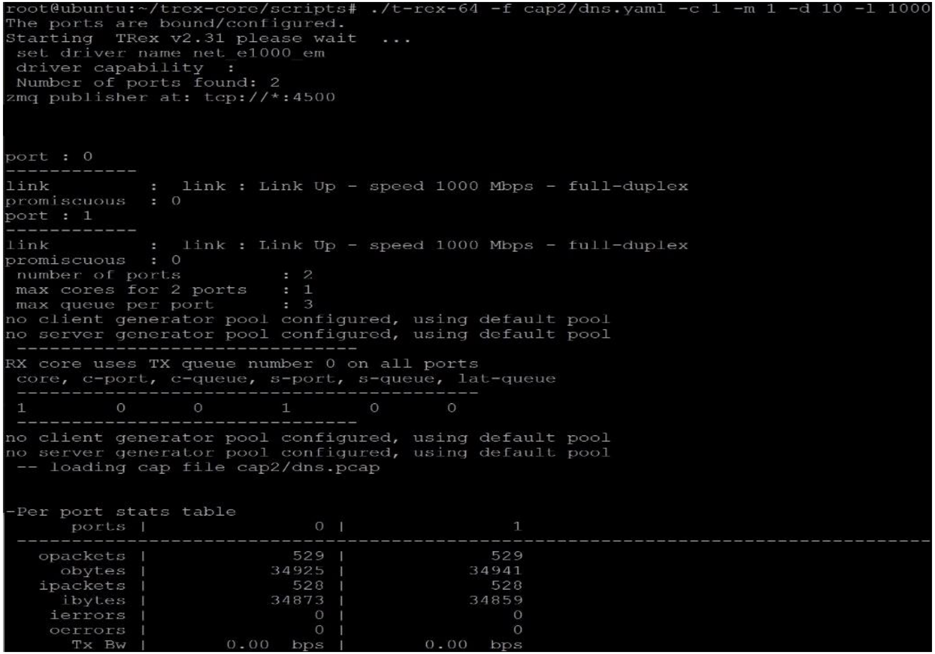
- DUT config -> static routes



```
ip route 16.0.0.0 255.0.0.0 192.168.10.188
ip route 48.0.0.0 255.0.0.0 192.168.86.188
```

Using TRex

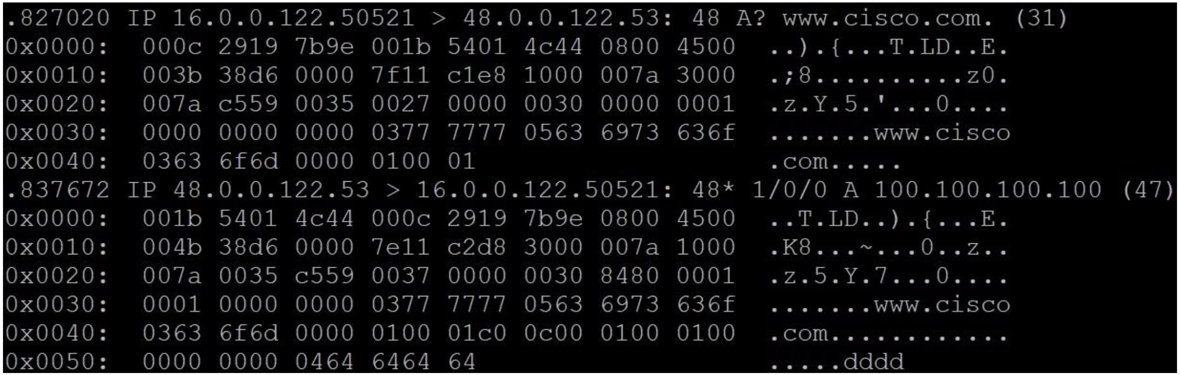
- Running TRex



```
root@ubuntu:~/trex-core/scripts# ./t-rex-64 -f cap2/dns.yaml -c 1 -m 1 -d 10 -l 1000
The ports are bound/configured.
Starting  TRex v2.31 please wait  ...
 set driver name net_e1000_em
 driver capability  :
 Number of ports found: 2
zmq publisher at: tcp://*:4500


port : 0
------------
link          :  link : Link Up - speed 1000 Mbps - full-duplex
promiscuous   : 0
port : 1
------------
link          :  link : Link Up - speed 1000 Mbps - full-duplex
promiscuous   : 0
 number of ports         : 2
 max cores for 2 ports   : 1
 max queue per port      : 3
no client generator pool configured, using default pool
no server generator pool configured, using default pool
 -------------------------------
RX core uses TX queue number 0 on all ports
 core, c-port, c-queue, s-port, s-queue, lat-queue
 ------------------------------------------
 1        0      0       1       0      0
 -------------------------------
no client generator pool configured, using default pool
no server generator pool configured, using default pool
 -- loading cap file cap2/dns.pcap


-Per port stats table
      ports |               0 |               1
 -----------------------------------------------------------------------------------
   opackets |             529 |             529
     obytes |           34925 |           34941
   ipackets |             528 |             528
     ibytes |           34873 |           34859
    ierrors |               0 |               0
    oerrors |               0 |               0
      Tx Bw |       0.00  bps |       0.00  bps
```

Using TRex

- Tcpdump showing TRex DNS query (taken on another machine)



```
.827020 IP 16.0.0.122.50521 > 48.0.0.122.53: 48 A? www.cisco.com. (31)
0x0000:  000c 2919 7b9e 001b 5401 4c44 0800 4500  ..).{...T.LD..E.
0x0010:  003b 38d6 0000 7f11 c1e8 1000 007a 3000  .;8.........z0.
0x0020:  007a c559 0035 0027 0000 0030 0000 0001  .z.Y.5.'...0....
0x0030:  0000 0000 0000 0377 7777 0563 6973 636f  .......www.cisco
0x0040:  0363 6f6d 0000 0100 01                   .com.....
.837672 IP 48.0.0.122.53 > 16.0.0.122.50521: 48* 1/0/0 A 100.100.100.100 (47)
0x0000:  001b 5401 4c44 000c 2919 7b9e 0800 4500  ..T.LD..).{...E.
0x0010:  004b 38d6 0000 7e11 c2d8 3000 007a 1000  .K8...~...0..z..
0x0020:  007a 0035 c559 0037 0000 0030 8480 0001  .z.5.Y.7...0....
0x0030:  0001 0000 0000 0377 7777 0563 6973 636f  .......www.cisco
0x0040:  0363 6f6d 0000 0100 01c0 0c00 0100 0100  .com............
0x0050:  0000 0000 0464 6464 64                   .....dddd
```

References

- TRex
https://trex-tgn.cisco.com/trex/

- Ubuntu 14.04 LTS
http://old-releases.ubuntu.com/releases/14.04.1/ubuntu-14.04-desktop-amd64.iso

- TRex Manual
https://trex-tgn.cisco.com/trex/doc/trex_manual.html