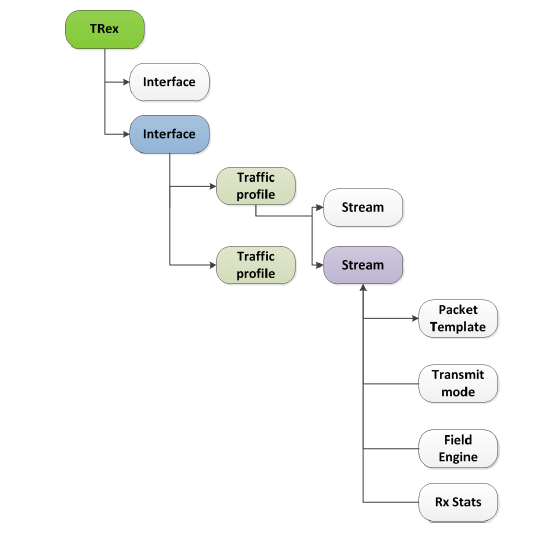
# 软件介绍

## 1.1 TRex基本介绍

对于网络设备而言，一款好的发包工具至关重要，而目前像IXIA， Sprient等仪表厂商的发包仪表虽然好用，但是价格昂贵，那么是否有一款在功能和性能上能与仪表相当，却又是开源的工具呢？思科的TRex的给出了答案。

目前，路由器厂商测试用的都是商业的仪表，一般的性能衡量都是以PPS作为标准，而路由器很复杂，牵扯到7层的应用，而且希望能够提供一种更加真实的流场景测试，TRex应运而生，TRex是一款开源的发包工具，它跑在标准的inter处理芯片上，同时支持stateful和stateless模式，stateful可以描述L4~L7层的应用场景，而stateless主要用来进行定制包的发包。

## 1.2 TRex结构



TRex: 每个装有TRex的主机可以提供多个接口进行发包

Interface: 每个接口支持1个或多个 traffic profile

Traffic Profile: 每个traffic profile可以包含一条或多条流

Stream: 每条流包含以下几方面

Packet： 由scapy进行构建

Field Engine: 可以提供包里面字段的改变，如源IP，如包大小，提供一定的灵活性

Mode: 包发送模式 cont/burst/mult-burst

Rx Stats: 根据每条流进行统计

Rate: 速率（pps/bandwidth）

Action: 可以指定流与流之间的关系

## 1.3 stateless模式和stateful模式

Stateless 与 Stateful不同，它更注重单个包的发送状态，它不在乎server有没有回应，它只在乎这个包如何构造，以什么样的形式发送，它更像IXExplorer。

Stateless是基于构包和发包两个阶段来完成， 构包完全基于python的scapy库，而发包则是其自开发的trex\_stl\_lib来完成。

stateful 是一种有状态的发包，虽然基于包模板进行顺序发送，但对于DUT来说是一个有状态的连接，因此它可以用在我们很多的测试场景中。

## 1.4 TRex的功能和性能

Stateful

支持高达200Gb/s的流量测试，需要网卡以及CPU支持，一般跟core个数有关

支持时延和抖动

支持NAT/PAT自动学习

支持L4~L7

Stateless

轻易产生stateless traffic stream

支持大流量20mpps

支持stream并发

支持更改报文字段范围

支持每条流的统计

支持pythonAPI

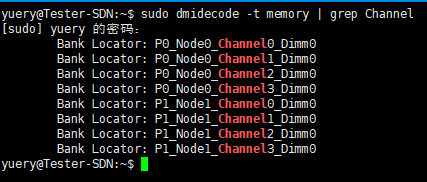
# 下载安装TRex

## 2.1 检查服务器的硬件

TRex软件必须安装在linux操作系统上，因为其依赖于linux内核，TRex当前依赖于X86系统架构，并且能够很好的在思科ucs硬件上运行。

### 2.1.1 服务器通道检查

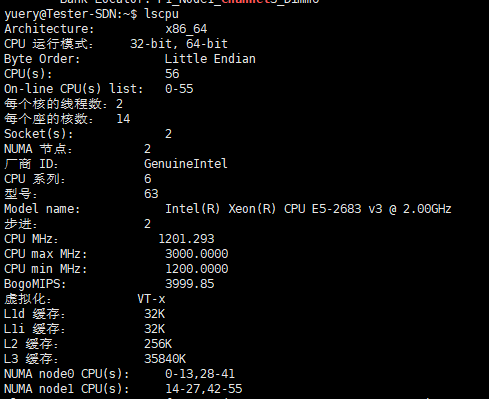
注意：比较重要的一点是服务器必须有4个DRAM通道，通道太少可能会影响性能问题，在服务器上使用sudo dmidecode -t memory | grep Channel来检查有几个通道。在我们使用的服务器上，如下图：



由此可以看出是4个channel，是符合TRex安装要求的。

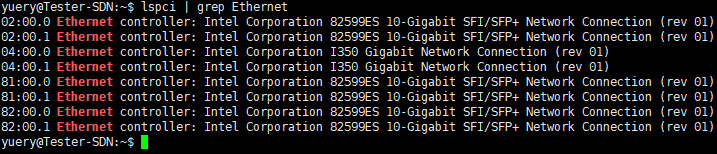
### 2.1.2 服务器CPU检查

服务器的CPU信息如下图：



### 2.1.3 服务器网卡检查

网卡信息如下图：



由如上信息可以看出，服务器有两个1G的I350网卡，主要用作管理口，有六个10G的Intel Corporation 82599ES网卡，用来进行发包。

## 2.2 安装操作系统

### 2.2.1 支持的版本

支持如下linux版本：

•Fedora 20-23, 64-bit kernel (not 32-bit)

• Ubuntu 14.04.1 LTS, 64-bit kernel (not 32-bit)

• Ubuntu 16.xx LTS, 64-bit kernel (not 32-bit)—Not fully supported.

• CentOs/RedHat 7.2 LTS, 64-bit kernel (not 32-bit)—This is the only working option for ConnectX-4.

这里我们选择64位的Ubuntu16.04.1LTS（desktop）版本来安装服务器。

### 2.2.2 下载linux

建议去ubuntu官网下载安装文件，下载时注意选择desktop版本，不要选择server版本，因为server版本没有桌面显示，可能会影响到TRex软件的使用。

### 2.2.3 安装linux

安装完成后，设置服务器的IP,GATEWAY ,DNS,ROUTE,DNS等相关信息。

## 2.3 获取TRex安装包并进行安装

在服务器上创建trex这个目录，进入该目录，使用wget命令来进行TRex安装包的下载，将安装包进行解压。

cd /opt

mkdir trex

cd trex

wget --no-cache http://trex-tgn.cisco.com/trex/release/latest

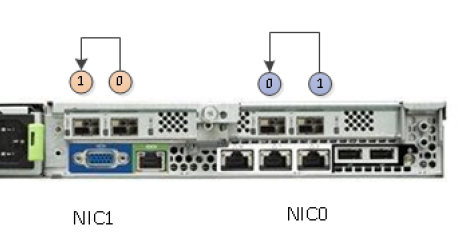
tar -xzvf latest

从网站上下载的是tar包，解压缩到指定目录即可。

由于服务器上的ubuntu版本已经安装好了，所以可以不必关心操作系统的安装以及TRex软件的安装，若感兴趣，可自行进行深入学习。

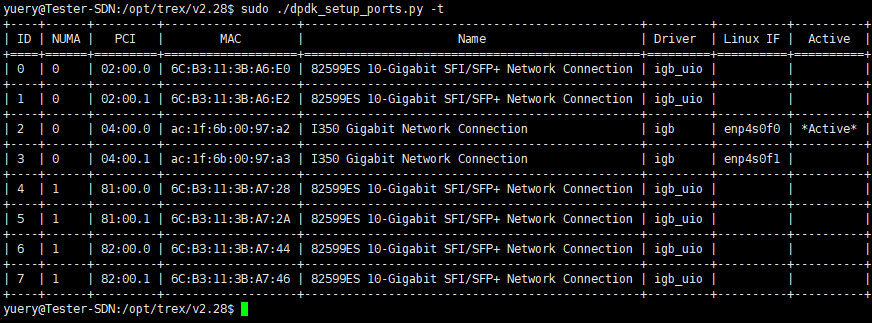
# 3、运行TRex

## 3.1 loopback配置

使用TRex服务器连接被测设备前，需要先检查网卡的环回功能正确。检查的方法如下图：

### 3.1.1 识别端口

使用TREx之前，需要先识别仪表上的端口，如下图：

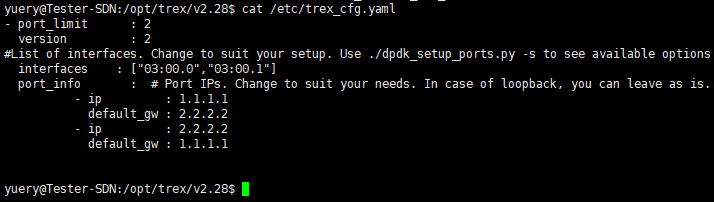


由上述结果，可以看出来，PCI号为04:00.0的网口为active的端口，该端口是管理端口，所以不能将该端口配置到config文件中，否则无法对服务器进行管理，其余端口可以进行配置。

### 3.1.2 创建配置文件

默认的配置文件为/etc/trex\_cfg.yaml（绝对路径），TRex已经提供了示例文件/opt/trex/v2.28/cfg/simple\_cfg.yaml（绝对路径），我们可以直接复制使用，命令为cp /opt/trex/v2.28/cfg/simple\_cfg.yaml /etc/trex\_cfg.yaml。

接下来要做的就是修改/etc/trex\_cfg.yaml这个文件的内容，修改方法为直接在命令行界面vim编辑（vim的使用这里不做过多描述，可自行进行学习，只需会简单使用即可满足本文档需求）。



如上图：

Port\_limit指的是要使用的网口数目的限制，这里的值与interfaces中端口的个数要一致，否则执行时会报错。

Version：指的是trex版本，这个字段无需修改。

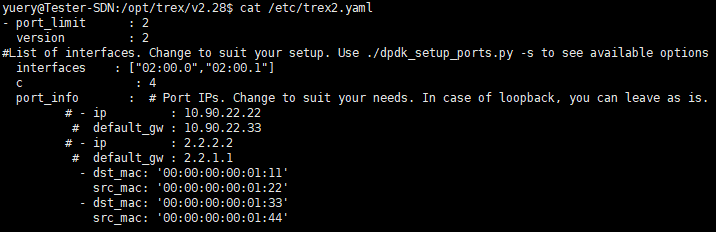
#后跟的内容是指注释内容。

Interfaces：03:00.0是指相对应网口的PCI号，查看PCI号的命令为sudo ./dpdk\_setup\_ports.py -t，其中“-t”参数指的是以table形式显示。

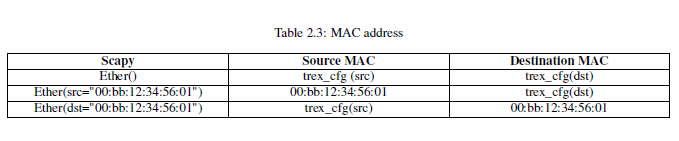
Port\_info：ip指的是本端虚接口地址，default\_gw指的是DUT接口地址

## 3.2 修改配置文件

在使用stateless模式发流之前，我们需要先对默认的/etc/trex\_cfg.yaml进行修改，使其对应于我们服务器上的信息，在这里给出一个示例，如下图：



由于我们的被测设备默认不开启三层功能，所以将IP信息注释掉，底下的src\_mac以及dst\_mac的用法是，如果后续发流包的报文中没有填充mac字段，则默认使用配置文件中的mac信息，若填充了mac字段，则使用填充信息，如下图：



## 3.3 stateless模式说明

Trex Stateless在运用时有三种方式

1、简单基于端口的一些测试模式，如ARP, PING

2、简单命令行交互，需要特定的py文件

3、纯Python接口编写的py文件，可直接执行

由于我们设备默认不开启三层功能，所以这里不予介绍第一种使用方式，仅介绍第二种使用方式。

## 3.4 scapy构包

Stateless的包是基于Scapy的，而在第一种使用模式中并没有用到scapy，是因为封装在service里函数里，我们用到的ARP，PING实际是封装好的，但是这不够灵活，函数也是有限的，如果希望能够灵活的发包，则Scapy是基础。

Scapy 是python用于发包的一个库，它通过layer的叠加从而达到构包的目的，如果你只需要构二层包，那么Ether()则够了， 如果你需要构三层IP包，则需要Ether()/IP()，同理L4~L7层的包都可以通过这种方式进行迭代，每个层里面都会有不同的协议，Scapy支持了目前的大多数协议，因此从理论上说，Scapy可以构造任意类型的包。



## 3.5 TRex已经给出的包模板

在stateless模式下，TRex已经给出了一些包模板，在/opt/trex/v2.28/stl这个目录下。

# stateless模式下发包

由于TRex给出的包模板很多，我们这里以/opt/trex/v2.28/stl/tests下的single\_cont.py为例进行说明。

## 4.1 搭建测试环境

**ssh连接**

**ssh连接**

**TREX**

**T9600**

**Port0**

**Port1**

**1/43**

**1/44**

**自动化脚本**

图4-1

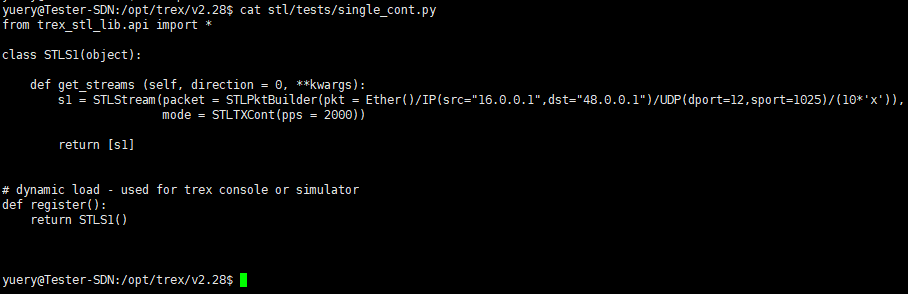
## 4.2 测试前的检查

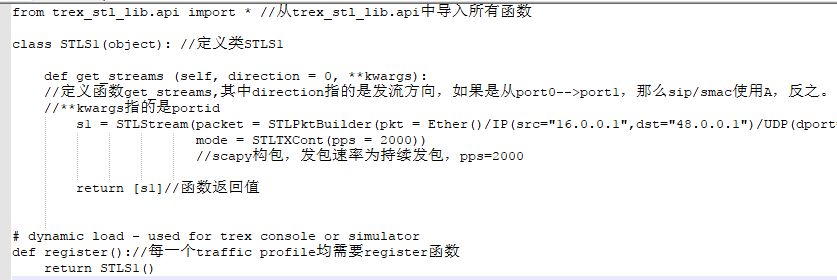
在进入stateless模式之前，先确认服务器上要使用的端口都是up状态，查看方法：（1）查看指示灯，（2）在CLI界面执行命令：ethtool 网卡名字，确认link status状态为yes。

此时，测试设备上的端口应该也是up状态。

如果测试仪上端口有down的情况，那么后续就无法进行测试，所以此步骤务必进行确认。

## 4.3查看single\_cont.py文件





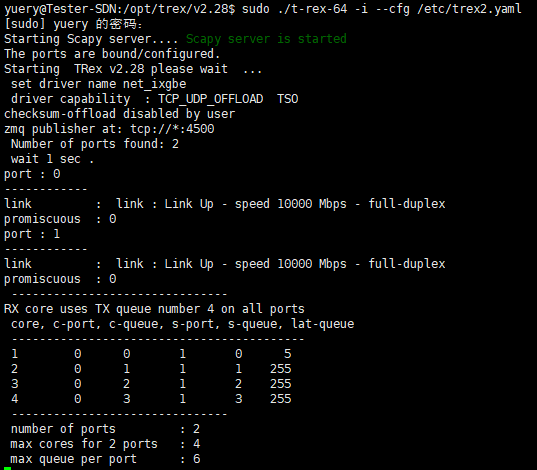
## 4.4执行命令进入stateless模式

### 4.4.1 执行命令sudo ./t-rex-64 -i --cfg /etc/trex2.yaml

-i:意思是进入stateless模式

--cfg:如果不加该参数，默认的配置文件为/etc/trex\_cfg.yaml，加了该参数，后面跟要使用的配置文件，我们这里使用/etc/trex2.yaml文件。

执行命令成功后，如下图：



### 4.4.2 启动终端

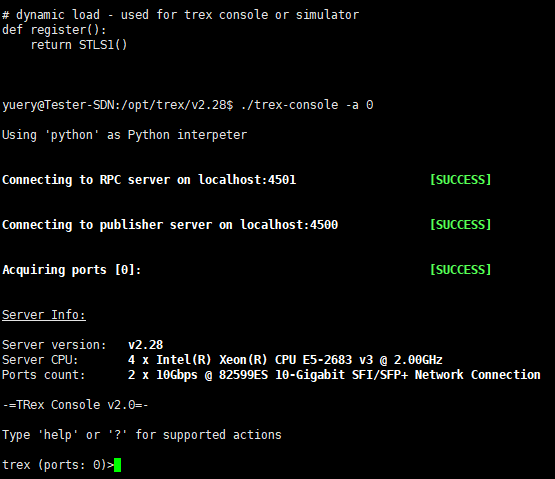
重新打开一个ssh连接，使用普通用户登录，进入/opt/trex/v2.28这个目录，

执行命令./trex-console –a 0

-a：accquire端口，占用端口，端口0：指的是服务器上一个网口，对应上图中的port 0。

默认会占用所有可以使用的端口。

执行命令成功后，进入如下界面：

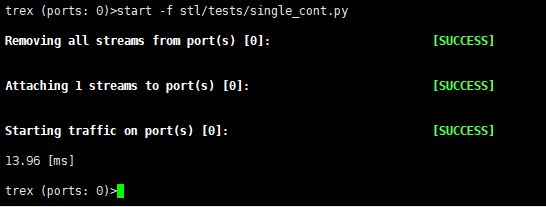


由上图可以看出，占用端口0成功。

### 4.4.3 发流

进入终端后，可以使用命令进行打流，停止打流等操作。

开始打流：



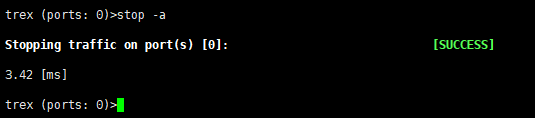
打流成功，其中-f参数指的是选择哪个py文件来进行使用。

也可以跟-m参数，指的是以多大的速率来进行打流。

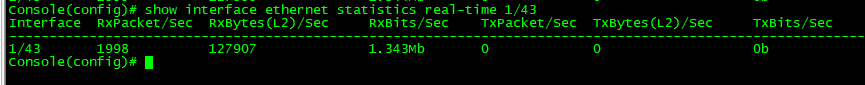
-m: [number][<empty> | bps | kbps | mbps | gbps | pps |kpps | mpps | % ]

加l1是指以layer1的速率发流，不加是指以layer2的速率发流。

停止打流



在DUT上查看流量如下：

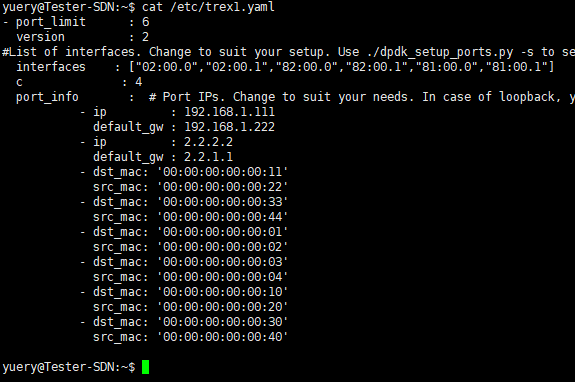


## 4.5 service模式

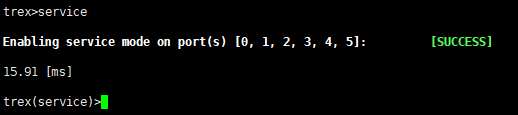
进入TRex终端后，有一种模式为service模式，该模式下，可以进行ARP/ICMP的回应，也可以进行报文的capture，下面我们分别来进行说明。

### 4.5.1 ARP/ICMP response

环境搭建如图4-1，配置43口的ip地址为192.168.1.222/24，在/etc/trex1.yaml配置文件中添加仪表口port0的IP地址为192.168.1.111，default\_gw为192.168.1.222，如下图所示：

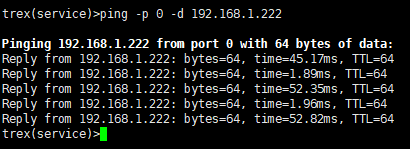


先进入stateless模式，命令为sudo ./t-rex-64 -i --cfg /etc/trex1.yaml，成功后开启另一个ssh连接，进入console终端，继而进入终端的service模式，如下图所示：



可以看出，service不加参数，默认会将所有可用端口开启service模式。

在该模式下，我们使用ping命令来对DUT进行操作，如下图：



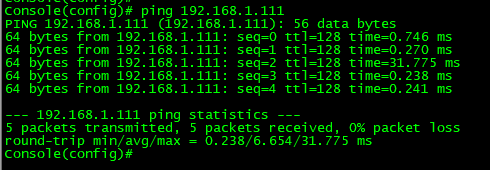
其中：

-p:指的是端口，如0指的就是端口0；

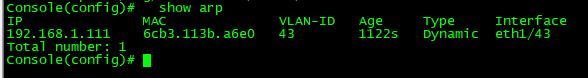
-d:指的是地址；

可以看出，ping DUT（192.168.1.222）成功。

到被测设备上，ping 测试仪，如下图：



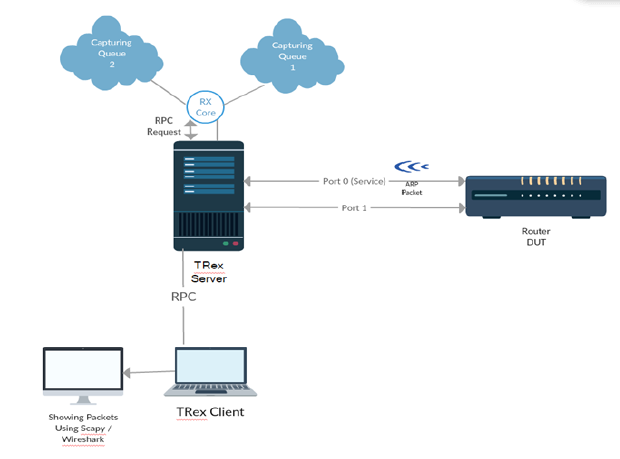
查看被测设备上的arp信息，如下图：



可以看出，ping 测试仪的port0（192.168.1.111）成功，且学习测试仪port0的MAC正确；

### 4.5.2 Packet Capturing

对于报文的捕获，TRex提供了两种方式（capture monitoring/capture recording）来检查和操作RX与TX的报文，下图是报文捕获的架构：



方式一：捕获监控

在这种方式下，TREX又提供了3种模式：

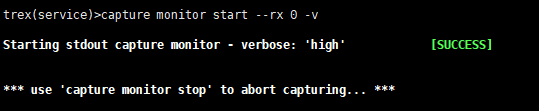
• Low Verbose - short line per packet will be displayed

• High Verbose - a full scapy show will be displayed per packet

• Wireshark Pipe - launches Wireshark with a pipe connected to the traffic being captured

在这里我们使用Verbose这种模式来举例说明，在该模式下，捕获到的报文会直接显示在console中，便于查看。

在service模式下开启监控，如下图：

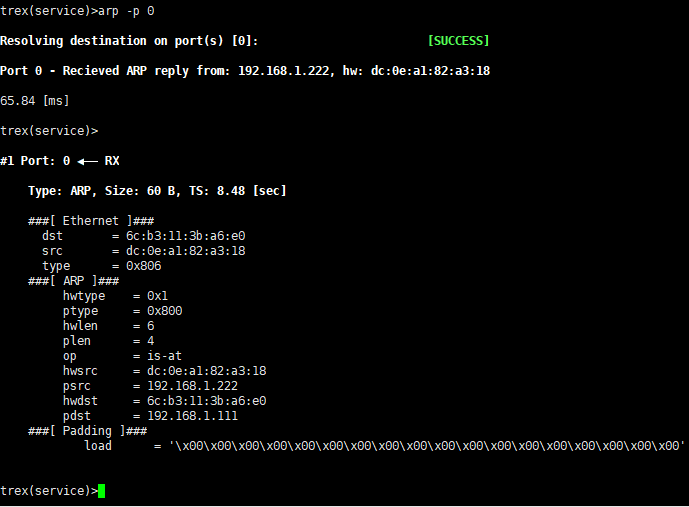


其中：

--rx:指的是捕获port0 RX方向的报文

-v:指的是verbose

开启监控后，我们在port0发送arp-request，观察报文捕获情况，如下图：

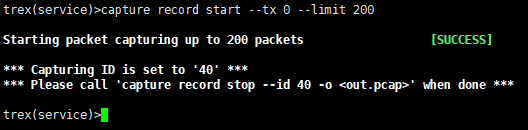


可以在console中直接对报文进行查看。

方式二：捕获记录

在这种方式下，我们使用TRex软件提供的py文件来进行发流，从而完成port0的报文捕获。

开启报文捕获记录，如下图所示：



其中：

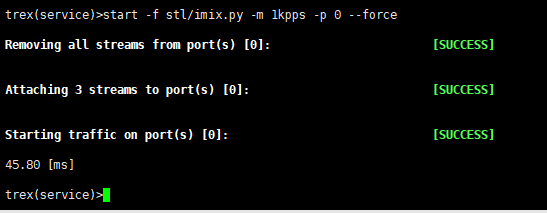
--tx: 对port0的TX方向进行报文捕获

--limit: 限制报文的个数

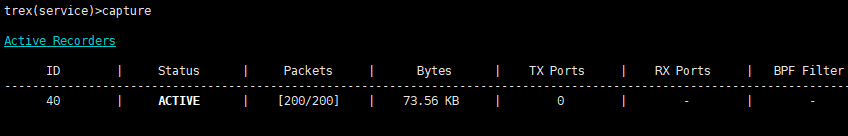
0: port 0

图中的capture ID为40

接下来发送报文，如下图：

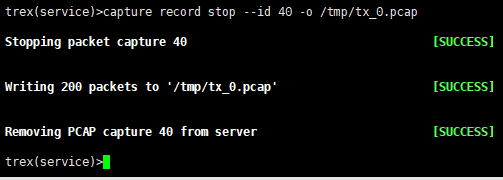


查看捕获，如下图所示：



可以看到已经捕获了200个报文，

停止抓包，并将捕获到的报文导出，如下图所示：



其中：

--id：40是我们在开启捕获记录的功能时，系统给出的ID；

-o:output的意思，将报文导出到你指定的目录下并且给予命名。

此时，可以去/tmp目录下查看已经有这个报文了，可以将该报文导出到本地使用wireshark查看，也可以到服务器桌面上打开wireshark应用进行查看。

# 补充

在使用过程中，如果对哪个命令有不明白的地方，可以跟“-h”进行查看，多试几次，对这个命令的使用就会熟悉了。

另外，对于stateful模式的使用，暂时还没有看到那里，所以后续进行补充。