SANGFOR
深信服科技

计算机网络基础

深信服科技研发专业能力系列课程

讲师介绍

SANGFOR 深信服科技

讲师照片

姓名：方统浩

部门：创新研究院

联系电话：15114603741

电子邮箱：jungle845943968@outlook.com

职位职称：
创新研究院网络研发工程师

SANGFOR
深信服科技

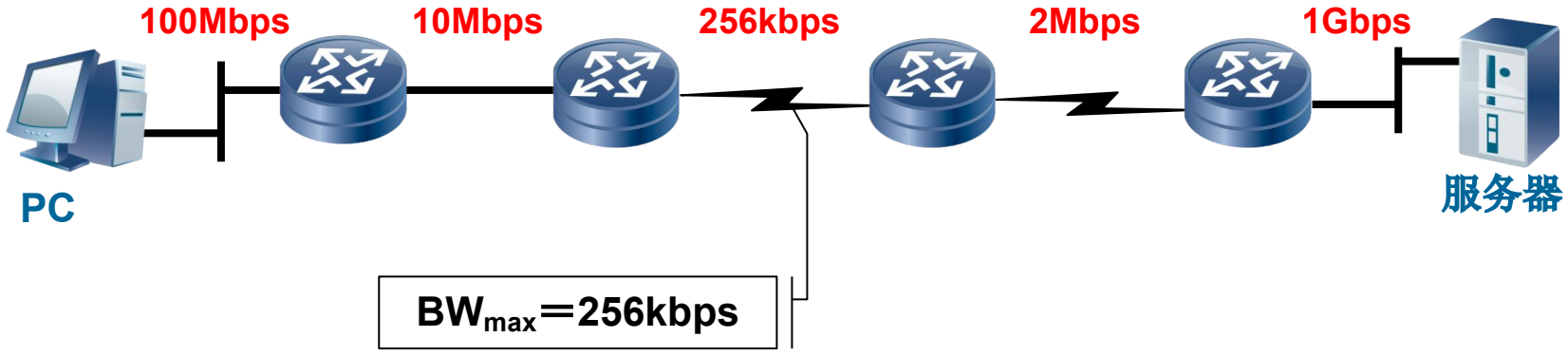
目录

SANGFOR
深信服科技

性能指标

速率与带宽

- 比特（bit）是计算机中数据量的单位，也是信息论中使用的信息量的单位
- 速率即比特率(bit rate)是计算机网络中最重要的一个性能指标。速率的单位是 b/s，或Kb/s, Mb/s, Gb/s 等
- 带宽



速率单位

SANGFOR 深信服科技

- 常用的速率单位是
  - 千比每秒，即 kb/s （10^3 b/s）
  - 兆比每秒，即 Mb/s（10^6 b/s）
  - 吉比每秒，即 Gb/s（10^9 b/s）
  - 太比每秒，即 Tb/s（10^{12} b/s）
- 请注意：在存储界，K = 2^{10} = 1024

  M = 2^{20}, G = 2^{30}, T = 2^{40}

吞吐量

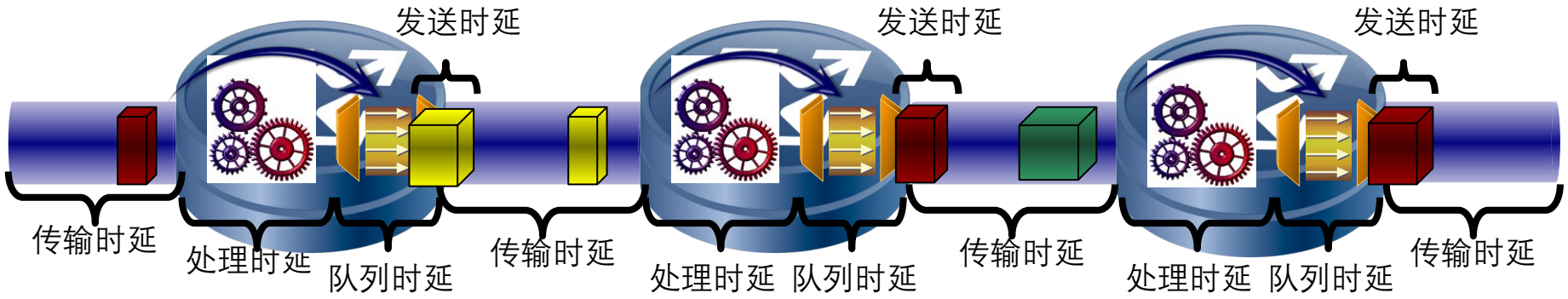
SANGFOR 深信服科技

- 吞吐量(throughput)表示在单位时间内通过某个网络（或信道、接口）的数据量
- 吞吐量更经常地用于对现实世界中的网络的一种测量，以便知道实际上到底有多少数据量能够通过网络或节点
- 吞吐量受网络的带宽或网络的额定速率的限制

总时延

- 数据经历的总时延就是发送时延、传输时延、处理时延和队列时延之和：

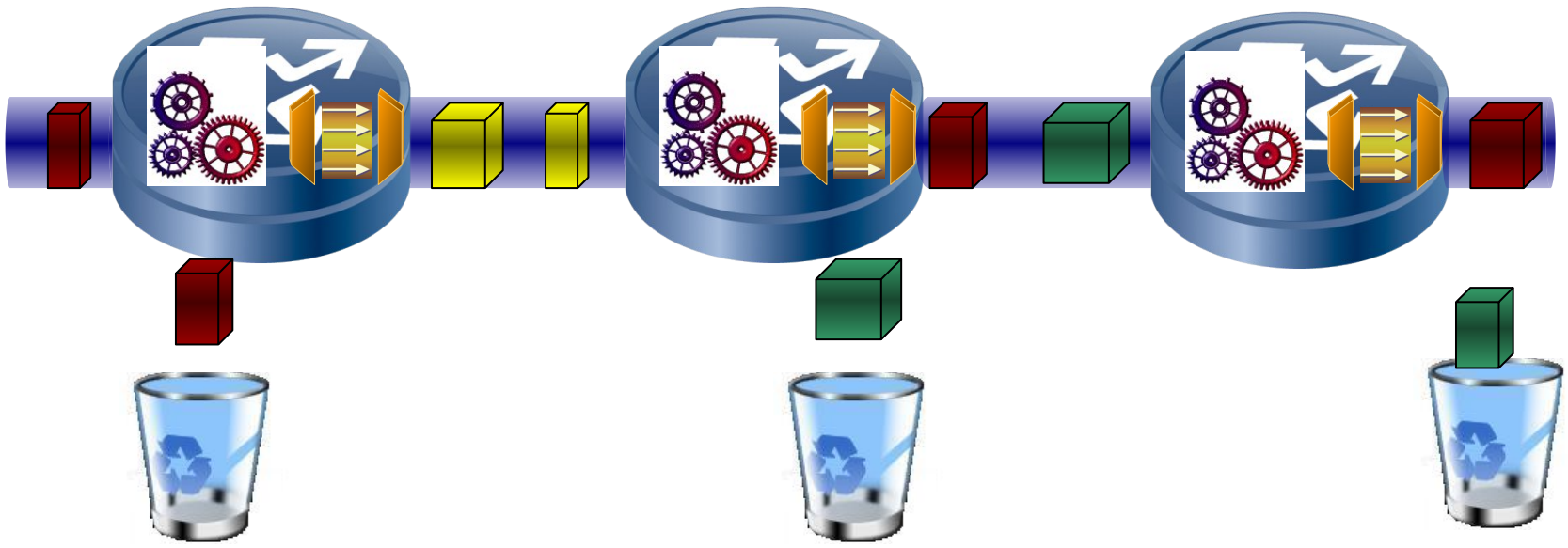
总时延 = 发送时延+传输时延+处理时延+队列时延



丢包率及抖动

SANGFOR 深信服科技

- 丢包率　　报文在传递过程中因为各种原因导致的丢包比率
- 抖动　　报文传递延迟发生的变化



新建

新建通常指新建连接速率，即Maximum TCP Connection Establishment Rate。从字面意思理解即每秒能处理的TCP链接个数。

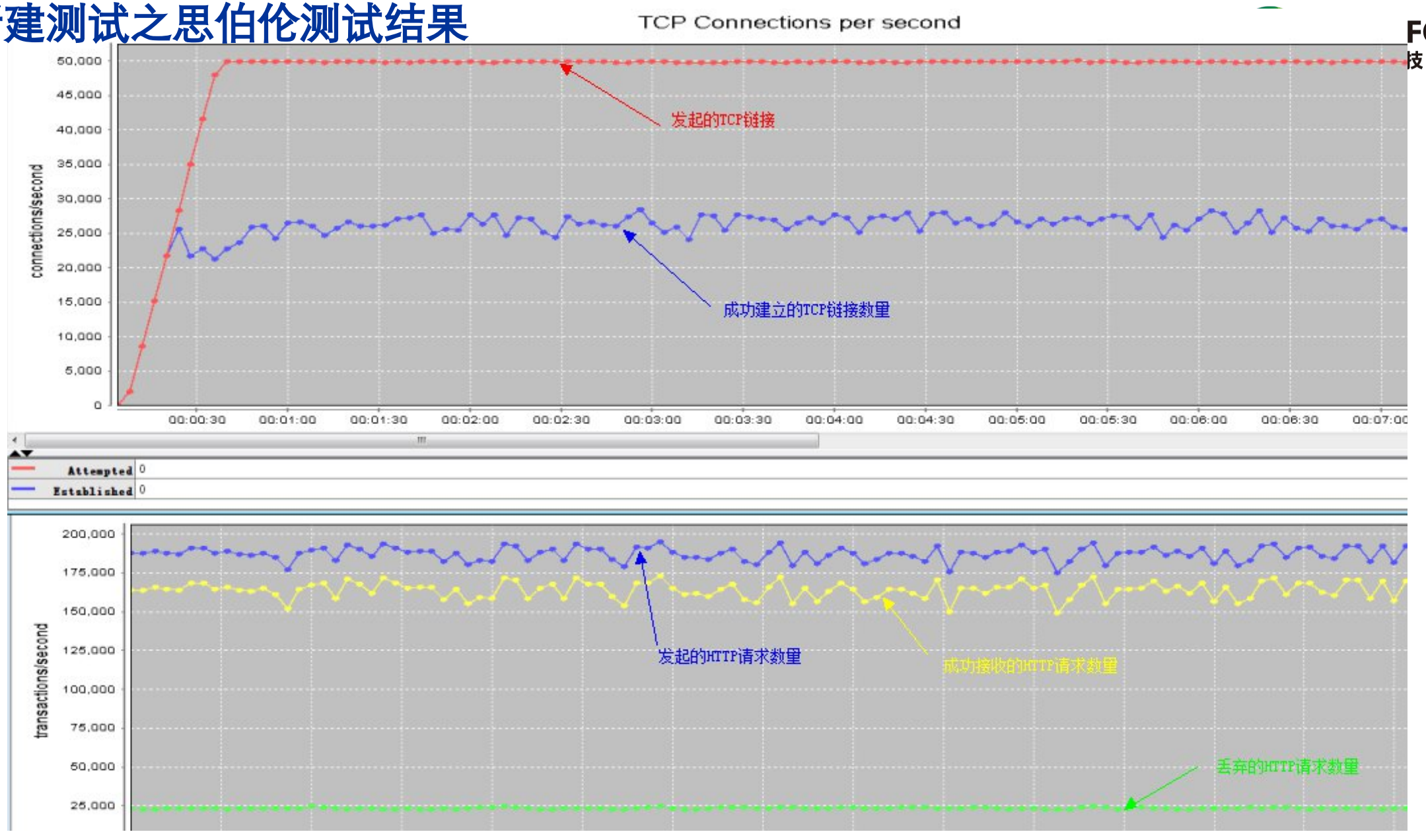
但是在业务场景下，新建通常分为TCP，即4层新建指标以及HTTP，即7层新建指标。由于一次TCP连接中可能携带多个HTTP请求，因此4层新建和7层新建往往性能是不同的。

并发

并发通常指并发连接数，即Concurrent TCP Connection Capacity。从字面意思理解即能同时处理的连接会话个数。

但是在业务场景下，并发与新建相同，通常也分为4层并发和7层并发。并发与新建不同的是，新建连接测试会立刻结束建立的连接，而并发测试不会结束建立的连接，所有已经建立的连接会已知保持直到达到设备的处理极限。

新建测试之思伯伦测试结果



SANGFOR
深信服科技

测试工具

netperf

Netperf是一种网络性能的测量工具，主要针对基于TCP或UDP的传输。Netperf以C/S的方式工作。通常，Netperf工具的client端用于发起网络测试，client端会通过控制连接向server端告知测试配置信息，其次client端还会向server端建立测试连接用于测试性能。利用netperf可以测量以下几种性能指标：

- 判断网络应用层是否可用，即TCP、UDP是否可达。
- TCP、UDP吞吐、时延等指标

netperf

netperf可以测量以下几种模式的TCP和UDP网络性能：

- TCP_STREAM : Client端向Server端发送批量的TCP数据。
- UDP_STREAM : Client端向Server端发送批量的UDP数据
- TCP_RR和TCP_CRR : 前者是在同一个连接中进行多次request和response请求，后者是每次请求新建一个连接（HTTP）
- UDP_RR : 使用UDP进行request和response请求

netperf使用

SANGFOR 深信服科技

```
root@ubuntu:~# netserver -p 1168
Starting netserver with host 'IN(6)ADDR_ANY' port '1168' and family AF_UNSPEC
root@ubuntu:~# netstat -ap | grep 1168
tcp6       0      0 [::]:1168               [::]:*                  LISTEN      6583/netserver
root@ubuntu:~#
```

netperf的Server端

```
[root@localhost bin]# ./netperf -t TCP_STREAM -H 10.230.4.5 -p 1168
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to 10.230.4.5 () port 0 AF_INET
Recv   Send    Send
Socket Socket  Message  Elapsed
Size   Size    Size     Time     Throughput
bytes  bytes   bytes    secs.    10^6bits/sec

 87380  16384  16384    10.03      940.01
```

netperf的Client端

Netperf局限性

作为一款“经典”的网络性能测试工具，netperf凭借轻量级、简单易用成为了测试性能的处不选择，但是Netperf同样有着一些局限性和问题

- Netperf的配置有限，无法配置发送数据包的源端口和目的端口。
- Netperf必须先建立连接才能进行发包测试，但是有一些测试场景不需要也不能建立连接的情况下Netperf就无法使用。
- 测试会受限于操作系统本身的系统调度影响，导致数据不准确。
- 无法测试太过于复杂的场景和功能。

RFC2544

RFC2544提供了对网络设备测试的一个基准，在此RFC中规定了一系列的测试过程和测试方法，

通过一个基准，让用户和服务商之间对测试的实施和结果达成共识。

- 吞吐量（Throughput）
- 时延（Latency）
- 丢包率（Frame Loss Rate）
- 背靠背（Back to Back）
- 系统恢复（System recovery）
- 复位测试（Rest）

RFC2544

RFC2544不仅仅提供了一系列基准指标用于测试，还指定了测试场景以及测试的配置。

在测试场景中，RFC2544提供了二层交换和三层转发的测试两种测试场景。

其次，在测试的配置中还指定了帧长度大小的规定，通常有

64byte、128byte、256byte、512byte、768byte、1024byte、1280byte、1518byte等几个档位。

9.1 Frame sizes to be used on Ethernet

64, 128, 256, 512, 1024, 1280, 1518

These sizes include the maximum and minimum frame sizes permitted by the Ethernet standard and a selection of sizes between these extremes with a finer granularity for the smaller frame sizes and higher frame rates.

SANGFOR
深信服科技

网络排障

网络排障

SANGFOR 深信服科技

- 台式机网络配置
  - IP地址
  - 子网掩码
  - 网关地址
  - 网线
- 网络设备配置
  - 办公区交换机ACL
  - AC访问控制
  - 网线
- 服务器、网安设备网络配置
  - IP地址
  - 子网掩码
  - 网关地址
  - 网线
- 实验室交换机
  - 网线
  - 环路
  - ACL

ifconfig

ifconfig（interface configuration）是在类Unix作业系统中于命令行界面（CLI）下或系统配置脚本中用于配置、控制及查询TCP/IP网络接口的系统管理工具。

通常，ifconfig可以用来查看以下信息：

- Mac地址、网口IP、Netmask；
- MTU大小。
- 收发包情况。

因此在排查网络设备无法正常连接的情况通常首先会利用ifconfig等命令查看网口是否有IP以及IP、Netmask是否正确。

```
Sangfor:aSV/host-a0369f9baf54 /sf # ifconfig
eth0      Link encap:Ethernet  HWaddr a0:36:9f:9b:af:54
          inet addr:10.250.0.7  Bcast:10.250.0.255  Mask:255.255.255.0
          UP BROADCAST ALLMULTI MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

eth1      Link encap:Ethernet  HWaddr a0:36:9f:9b:af:55
          inet addr:10.230.4.2  Bcast:10.230.255.255  Mask:255.255.0.0
          inet6 addr: fe80::a236:9fff:fe9b:af55/64 Scope:Link
          UP BROADCAST RUNNING ALLMULTI MULTICAST  MTU:1500  Metric:1
          RX packets:79108 errors:0 dropped:0 overruns:0 frame:0
          TX packets:15151 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:22693417 (21.6 MiB)  TX bytes:5635341 (5.3 MiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:2181239075 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2181239075 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:437316671393 (407.2 GiB)  TX bytes:437316671393 (407.2 GiB)
```

ip

ip命令和ifconfig命令类似，但是前者的功能更加强大，后者在net-tools中已经过时，在一些Linux发行版中甚至默认不携带ifconfig。

```
Sangfor:aSV/host-a0369f9baf54 /sf # ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
8: eth0: <NO-CARRIER,BROADCAST,MULTICAST,ALLMULTI,UP> mtu 1500 qdisc mq state DOWN qlen 1000
    link/ether a0:36:9f:9b:af:54 brd ff:ff:ff:ff:ff:ff
    inet 10.250.0.7/24 brd 10.250.0.255 scope global eth0
       valid_lft forever preferred_lft forever
34: eth1: <BROADCAST,MULTICAST,ALLMULTI,UP,LOWER_UP> mtu 1500 qdisc mq state UP qlen 1000
    link/ether a0:36:9f:9b:af:55 brd ff:ff:ff:ff:ff:ff
    inet 10.230.4.2/16 brd 10.230.255.255 scope global eth1
       valid_lft forever preferred_lft forever
    inet6 fe80::a236:9fff:fe9b:af55/64 scope link
       valid_lft forever preferred_lft forever
35: eth4: <BROADCAST,MULTICAST,ALLMULTI> mtu 1500 qdisc noop state DOWN qlen 1000
    link/ether 74:a4:b5:00:ee:68 brd ff:ff:ff:ff:ff:ff
36: eth5: <BROADCAST,MULTICAST,ALLMULTI> mtu 1500 qdisc noop state DOWN qlen 1000
    link/ether 74:a4:b5:00:ee:69 brd ff:ff:ff:ff:ff:ff
37: eth2: <BROADCAST,MULTICAST,ALLMULTI> mtu 1500 qdisc noop state DOWN qlen 1000
    link/ether 0c:c4:7a:ab:b5:5a brd ff:ff:ff:ff:ff:ff
38: eth3: <BROADCAST,MULTICAST,ALLMULTI> mtu 1500 qdisc noop state DOWN qlen 1000
    link/ether 0c:c4:7a:ab:b5:5b brd ff:ff:ff:ff:ff:ff
Sangfor:aSV/host-a0369f9baf54 /sf #
```

ifconfig以及ip排障场景的应用

在网络故障的场景下，网络故障的原因往往错综复杂，无法单一的靠某条信息或某条命令就可以完成排查的工作，但是ifconfig和ip命令仍然提供了一些宝贵的线索。

例如在某网络无法连接的场景下（无法ping或无法ssh），可以用如下步骤进行初步的故障诊断。

- 首先ping目标ip是否可以ping通，观察ping返回的结果是否为Destination Host Unreachable.
- 在机器后台可以通过ifconfig以及ip link命令查看目标网口的状态是否为UP状态。
- 如果网口的状态已经为UP状态，可以利用ifconfig以及ip addr命令查看网口的ip、掩码是否正常。
- 如果网口的ip以及掩码状态正常，但是无法ping通往关，可以通过ip route命令查看路由表是否配置正常。

ethtool

ethtool是linux系统下用于显示以及修改一些关于网卡和设备驱动配置的工具。（此工具功能强大，使用略复杂）。此工具通常用于排查网卡即驱动故障，例如以下几种场景。

- 查看网卡驱动、驱动版本、固件版本、总线地址。
- 查看网卡的收发包状态、是否有丢包、丢包原因。
- 查看网卡寄存器。
- 查看网卡特性（feature）以及收发队列大小。
- 更改网卡特性，例如可以让网卡开启TSO、LRO。

ethtool排障场景的应用其一

在测试网络性能的场景下，经常会遇到网卡收包出现了丢包的场景。ethtool工具可以告知使用者是否发生的丢包，是由于什么因素导致了丢包。

ethtool -S [Nic_name]

可以看到ethtool可以统计出出错的数据包的个数以及原因。以rx_missed_errors错误为例，此错误的直接原因是数据包还没有被收到网卡的rx ring即被丢弃。通常产生此错误可能是cpu处理不及时导致数据包无法被收到rx ring中，数据包因此被Nic丢弃。

```
root@ubuntu:~# ethtool -S enp4s0f0
NIC statistics:
     rx_packets: 52839
     tx_packets: 604
     rx_bytes: 17512586
     tx_bytes: 100612
     rx_crc_errors: 0
     rx_no_buffer_count: 0
     rx_missed_errors: 0
     rx_long_length_errors: 0
     rx_short_length_errors: 0
     rx_align_errors: 0
     rx_long_byte_count: 17512586
```

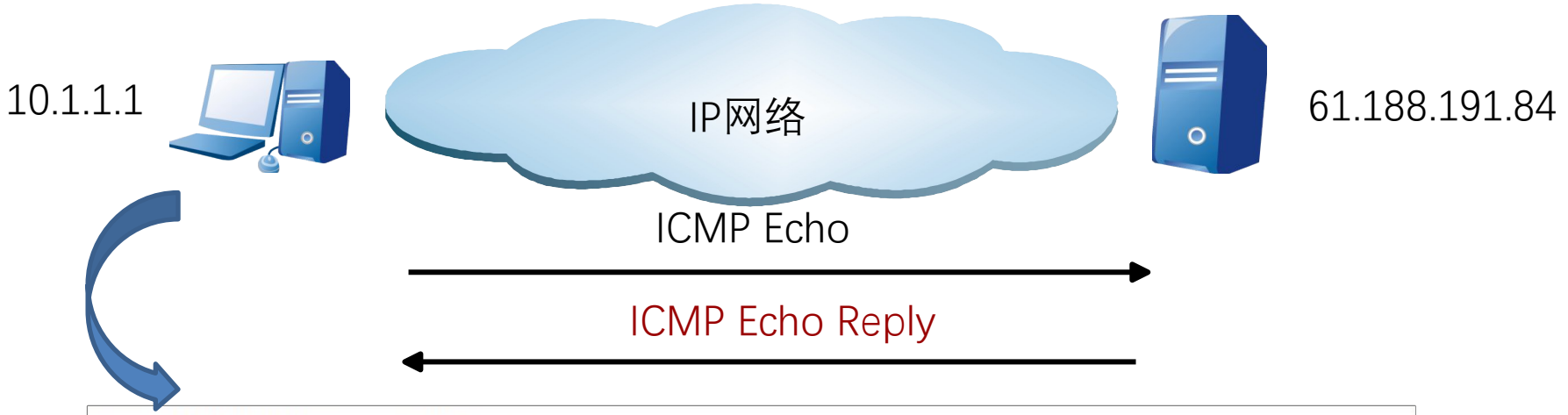
ethtool排障场景的应用其二

在测试网络性能的场景下，经常会遇到网卡无法UP的情况，即插上网线后发现网卡无法被上电等问题，此问题有时会涉及到查看网卡驱动版本以及网卡固件版本。

ethtool -i [Nic_name]

```
root@ubuntu:~# ethtool -i ens5f1
driver: ixgbe
version: 4.2.1-k
firmware-version: 0x2b2c0001
expansion-rom-version:
bus-info: 0000:81:00.1
supports-statistics: yes
supports-test: yes
supports-eeprom-access: yes
supports-register-dump: yes
supports-priv-flags: no
```

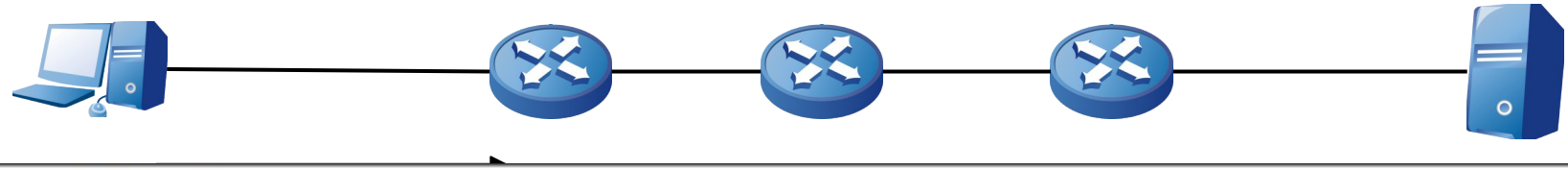
ICMP应用——Ping




```
C:\WINDOWS\system32\cmd.exe

C:\>ping 61.188.191.84

正在 Ping 61.188.191.84 具有 32 字节的数据:
来自 61.188.191.84 的回复: 字节=32 时间=11ms TTL=56
来自 61.188.191.84 的回复: 字节=32 时间=10ms TTL=56
来自 61.188.191.84 的回复: 字节=32 时间=29ms TTL=56
来自 61.188.191.84 的回复: 字节=32 时间=29ms TTL=56

61.188.191.84 的 Ping 统计信息:
    数据包: 已发送 = 4，已接收 = 4，丢失 = 0 (0% 丢失)，
往返行程的估计时间(以毫秒为单位):
    最短 = 10ms，最长 = 29ms，平均 = 19ms
```

ICMP应用——tracert

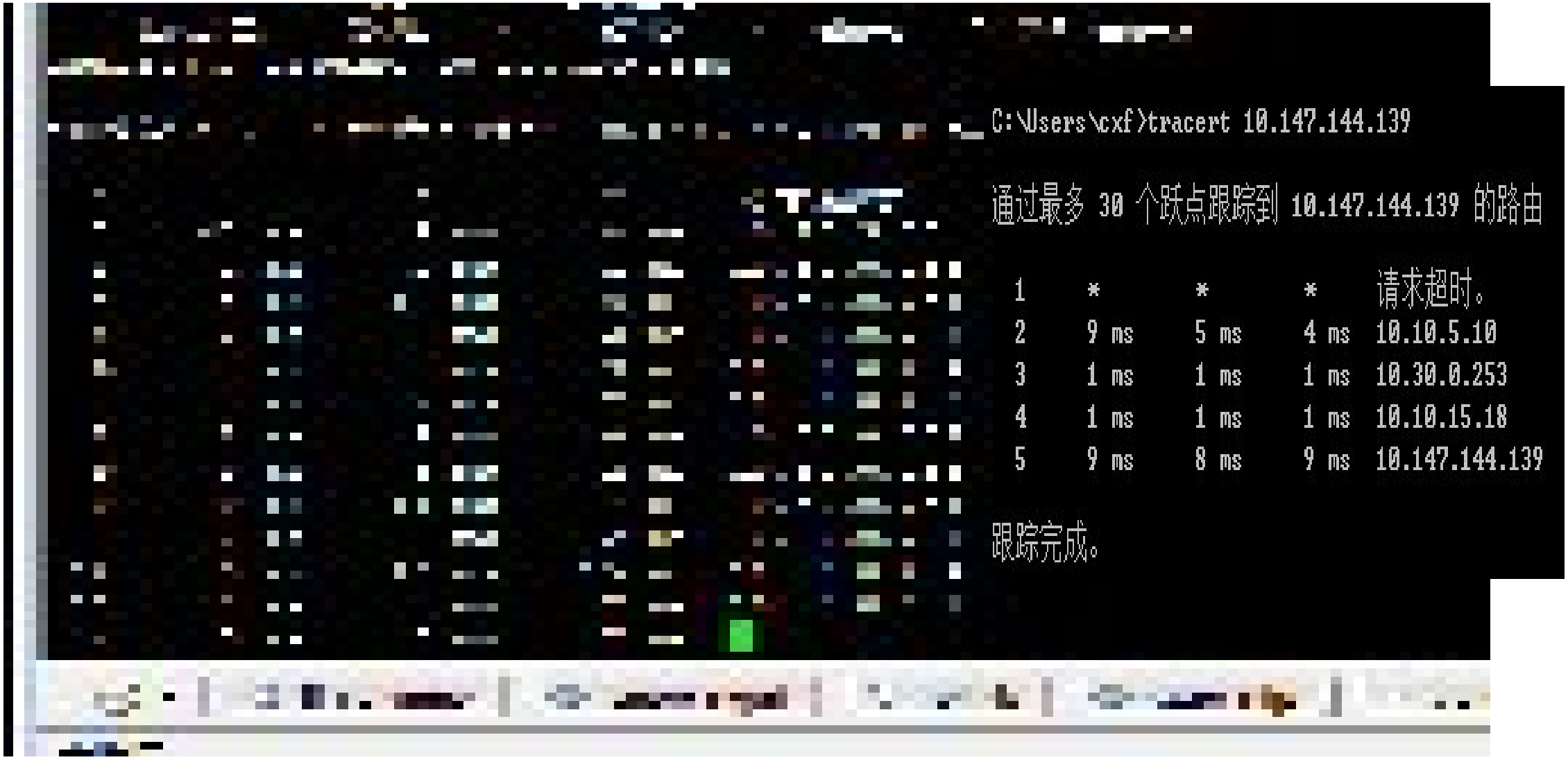
SANGFOR 深信服科技



命令提示符

```
C:\> tracert 182.151.192.101

通过最多 30 个跃点跟踪到 182.151.192.101 的路由

  1     2 ms     1 ms     1 ms  192.168.247.254
  2     2 ms     1 ms     1 ms  192.168.7.246
  3     6 ms     3 ms     4 ms  125.71.215.1
  4     9 ms     4 ms     4 ms  182.151.192.101

跟踪完成。
```

网络排障


C:\Users\cxf>tracert 10.147.144.139
通过最多 30 个跃点跟踪到 10.147.144.139 的路由
1 * * * 请求超时。
2 9 ms 5 ms 4 ms 10.10.5.10
3 1 ms 1 ms 1 ms 10.30.0.253
4 1 ms 1 ms 1 ms 10.10.15.18
5 9 ms 8 ms 9 ms 10.147.144.139
跟踪完成。

tcpdump

tcpdump是一个运行于类unix环境下的嗅探工具，通常用于用户拦截和显示发送或收到过网络连接到该计算机的TCP/IP和其他数据包。

```
[root@server2 ~]# tcpdump -c 5 -nn -i eth0 icmp

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
12:11:23.273638 IP 192.168.100.70 > 192.168.100.62: ICMP echo request, id 16422, seq 10, length 64
12:11:23.273666 IP 192.168.100.62 > 192.168.100.70: ICMP echo reply, id 16422, seq 10, length 64
12:11:24.356915 IP 192.168.100.70 > 192.168.100.62: ICMP echo request, id 16422, seq 11, length 64
12:11:24.356936 IP 192.168.100.62 > 192.168.100.70: ICMP echo reply, id 16422, seq 11, length 64
12:11:25.440887 IP 192.168.100.70 > 192.168.100.62: ICMP echo request, id 16422, seq 12, length 64
5 packets captured
6 packets received by filter
0 packets dropped by kernel
```

tcpdump使用其一

SANGFOR 深信服科技

tcpdump通常可以用于更细粒度的排查网络故障。例如可以通过tcpdump进行对包采样来判断“期望”的数据包在目标网卡上是否发送/接收。

以一个简单的场景为例：client与server的链接无法建立，并且发现：

- ip命令显示网卡Link状态正常、路由表正常、ip以及netmask正常。
- ping工具显示对对端的三层网络可达。

但是以上两步排查实际上仍然无法判断client端发起的SYN TCP数据包是否到server端，此时可以利用tcpdump来进行抓包观察SYN TCP数据包是否已经在client端成功发出，且在目的server端成功抵达来排除协议栈以及中间网络出现故障的可能性。

tcpdump使用其二

有时，经过网卡的数据包过多，需要对数据包进行过滤后再抓取，tcpdump工具支持抓取数据包时开启过滤功能，同时也支持将抓取的数据包保存为.pcap文件，供wireshark等软件来进一步分析。

```
[root@server2 ~]# tcpdump -c 2 -q -XX -vvv -nn -i eth0 tcp dst port 22
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
12:15:54.788812 IP (tos 0x0, ttl 64, id 19303, offset 0, flags [DF], proto TCP (6), length 40)
    192.168.100.1.5788 > 192.168.100.62.22: tcp 0
        0x0000:  000c 2908 9234 0050 56c0 0008 0800 4500  ..)..4.PV.....E.
        0x0010:  0028 4b67 4000 4006 a5d8 c0a8 6401 c0a8  .(Kg@.@.....d...
        0x0020:  643e 169c 0016 2426 5fd6 1fec 2b62 5010  d>....$&_...+bP.
        0x0030:  0803 7844 0000 0000 0000 0000            ..xD........
12:15:54.842641 IP (tos 0x0, ttl 64, id 19304, offset 0, flags [DF], proto TCP (6), length 40)
    192.168.100.1.5788 > 192.168.100.62.22: tcp 0
        0x0000:  000c 2908 9234 0050 56c0 0008 0800 4500  ..)..4.PV.....E.
        0x0010:  0028 4b68 4000 4006 a5d7 c0a8 6401 c0a8  .(Kh@.@.....d...
        0x0020:  643e 169c 0016 2426 5fd6 1fec 2d62 5010  d>....$&_...-bP.
        0x0030:  0801 7646 0000 0000 0000 0000            ..vF........
2 packets captured
2 packets received by filter
0 packets dropped by kernel
```

wireshark

SANGFOR 深信服科技

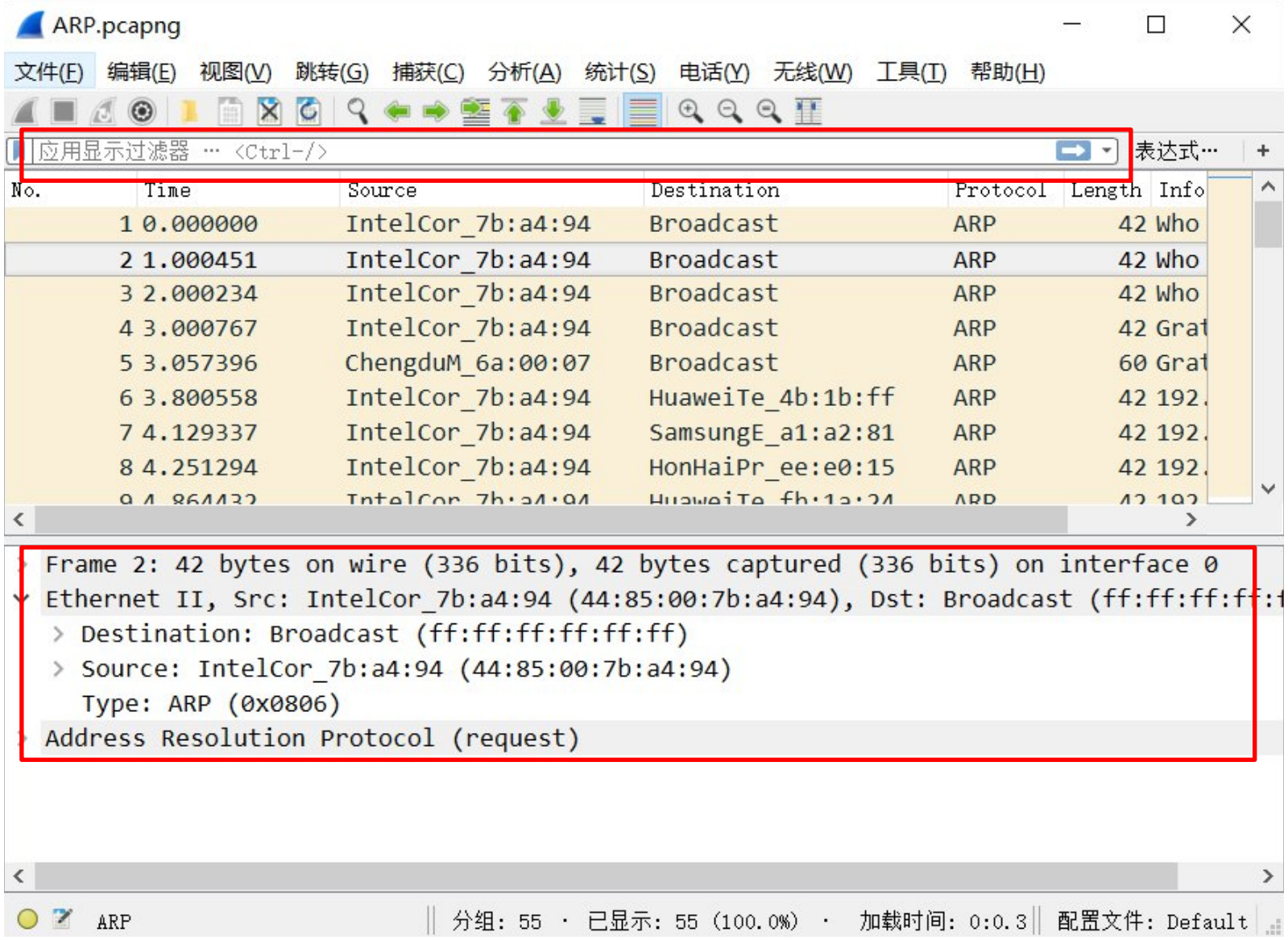
wireshark是一款免费开源的数据包分析软件，并且wireshark也可以实现类似tcpdump一样的抓包功能。

通常，wireshark用于相关工作人员用于检测网络故障等问题，在实际生产环境中，由于生产环境通常为Linux Server版，更倾向于利用tcpdump抓包后保存为.pcap格式文件，再在本地windows端利用wireshark来对pcap文件中的内容进行进一步的分析。

wireshark的特点：

- 可以捕获多种接口的报文
- 可以根据丰富的协议库对捕获报文进行解析

wireshark



wireshark可以将数据包中的内容以非常形象的方式展示出来，并且wireshark同样支持过滤功能来过滤出目标数据包

wireshark

SANGFOR 深信服科技

应用显示过滤器 … <Ctrl-/>

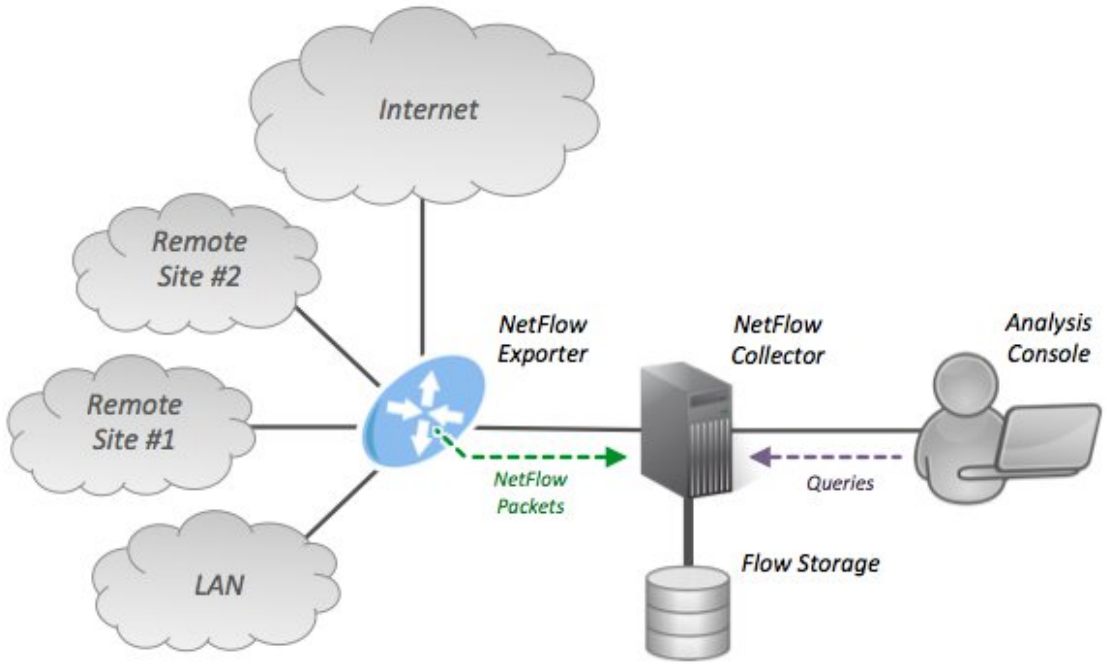
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.13	192.168.1.12	ICMP	74	Echo (ping) request id=0x0001, seq=9/2304, ttl=128 (no response found!)
2	4.686098	fe:fc:fe:8c:c7:34	fe:fc:fe:d6:6d:4b	ARP	42	Who has 192.168.1.12? Tell 192.168.1.13
3	4.686197	192.168.1.13	192.168.1.12	ICMP	74	Echo (ping) request id=0x0001, seq=10/2560, ttl=128 (no response found!)
4	4.686374	fe:fc:fe:d6:6d:4b	fe:fc:fe:8c:c7:34	ARP	42	192.168.1.12 is at fe:fc:fe:d6:6d:4b
5	9.186146	192.168.1.13	192.168.1.12	ICMP	74	Echo (ping) request id=0x0001, seq=11/2816, ttl=128 (no response found!)

```
Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
Ethernet II, Src: fe:fc:fe:8c:c7:34 (fe:fc:fe:8c:c7:34), Dst: fe:fc:fe:d6:6d:4b (fe:fc:fe:d6:6d:4b)
Internet Protocol Version 4, Src: 192.168.1.13, Dst: 192.168.1.12
Internet Control Message Protocol
```

```
0000  fe fc fe d6 6d 4b fe fc  fe 8c c7 34 08 00 45 00   ....mK.. ...4..E.
0010  00 3c 00 52 00 00 80 01  b7 05 c0 a8 01 0d c0 a8   .<.R.... ........
0020  01 0c 08 00 4d 52 00 01  00 09 61 62 63 64 65 66   ....MR.. ..abcdef
0030  67 68 69 6a 6b 6c 6d 6e  6f 70 71 72 73 74 75 76   ghijklmn opqrstuv
0040  77 61 62 63 64 65 66 67  68 69                    wabcdefg hi
```

NetFlow

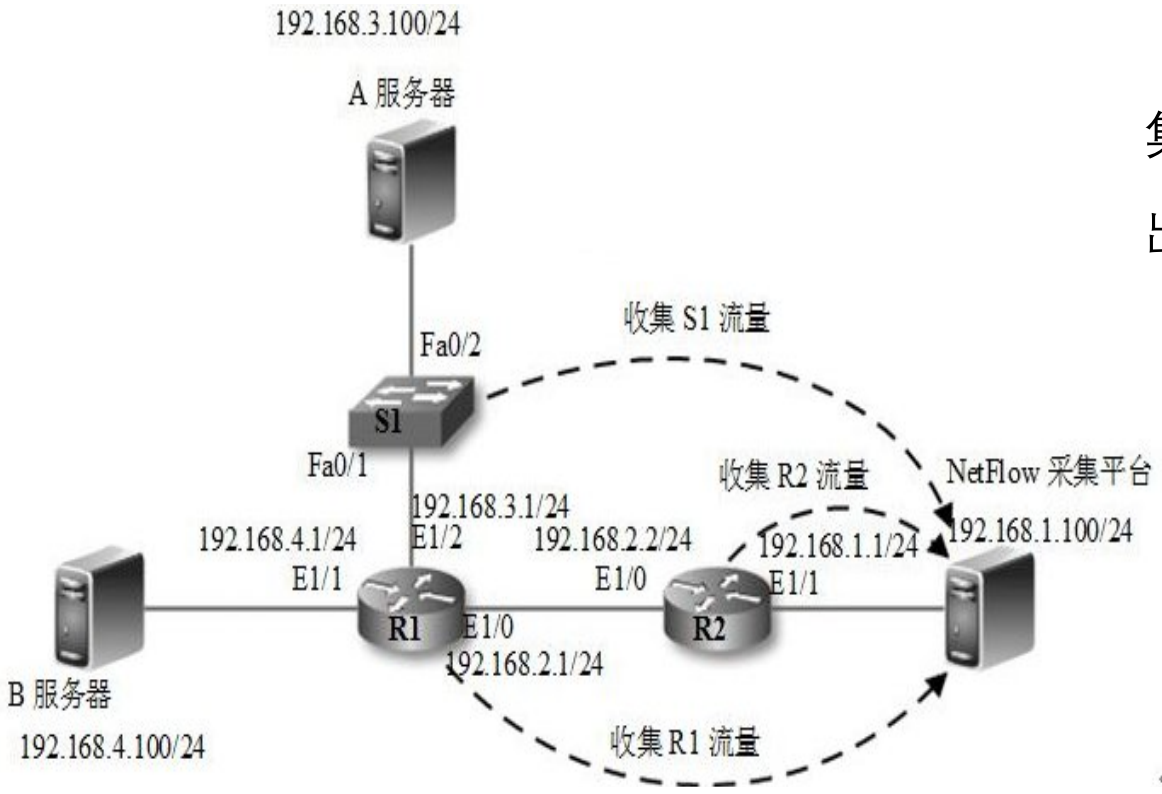
Netflow是一种思科开发的基于流的流量分析技术，通常应用在路由器以及交换机等产品上。经由分析Netflow收集到的信息，网络相关人员可以知道流量的来源以及目的、种类以及造成网络拥塞等故障的原因。



左图为常见的Netflow的架构。Netflow架构通常由以下几部分组成：

- Netflow Exporter : Netflow导出器用于导出Netflow流量。
- Netflow Collector : Netflow收集器，用于收集导出的Netflow流量。
- Flow Storage : 流量存储节点，由于网络中流量信息规模过大，需要单独的流量信息存储节点
- Analyzer : 分析器，对采集的流量进行分析。

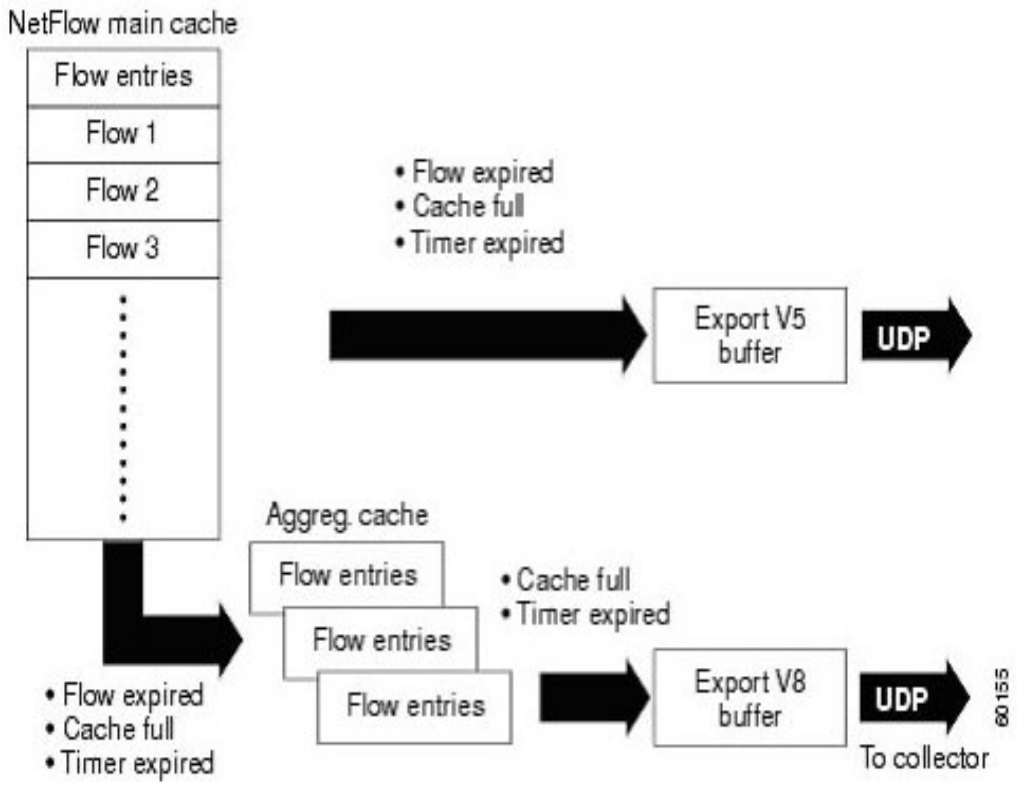
NetFlow的工作原理



以左图为例：

Netflow设备（通常为交换机和路由器）会收集网络中途径设备的流量信息，并将流量信息导出给Netflow采集平台。

NetFlow的工作原理



以左图为例：

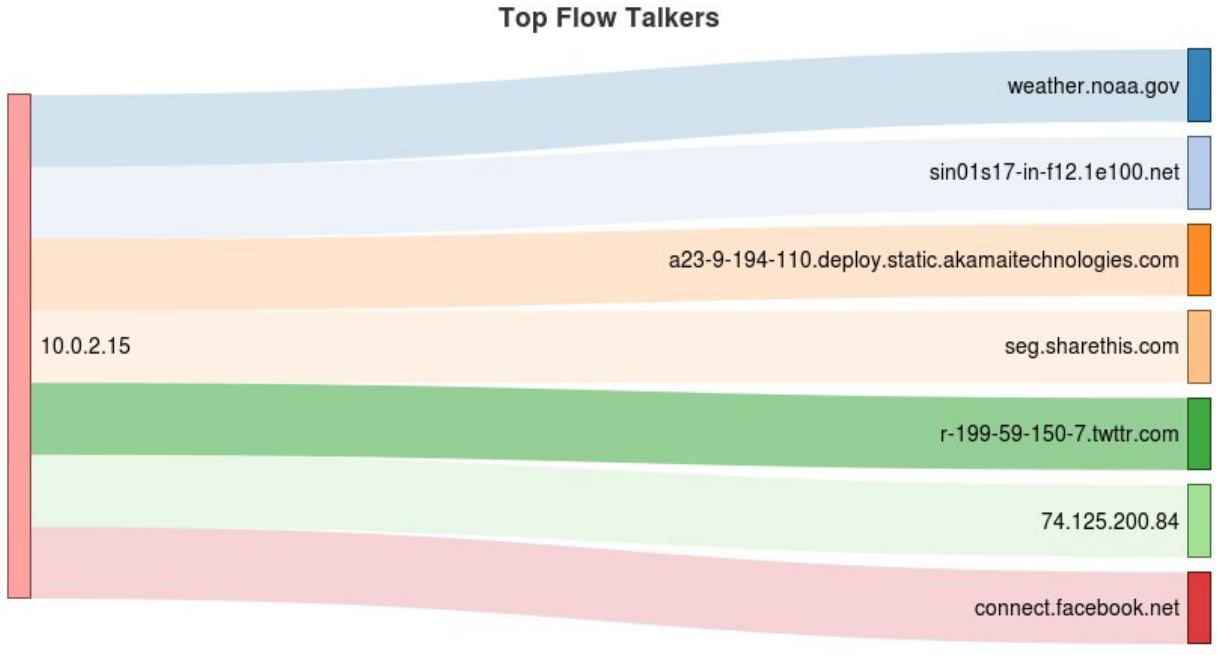
在Netflow设备上，途径的流量会按照一定的规则存储在本地Flow cache表中。在这张表中，每一个表项即代表这一条对应的网络流量，例如规则可以如下：

{src_ip, src_port, dst_ip, dst_port, l4_proto}

此外，Flow cache还有一定的超时机制，当某个表项面临超时，即代表表项已经转变为inactive状态，此时设备就会通过导出协议将Flow cache中的表项导出至Netflow Collector中。

需要注意的是，导出的协议叫做Netflow协议，但是Netflow技术本身并不止局限于Netflow导出协议，更重要的是Flow cache

NetFlow的使用



可以统计当前一段时间内的网络流量的带宽占比。

```
1 • SELECT * FROM ntopng.flowsv6_2;
```

100% 1:1

Result Grid | Filter Rows: Search | Edit: | Export/Import: | Fetch rows:

idx	VLAN_ID	L7_PROTO	IP_SRC_ADDR	L4_SRC_PORT	IP_DST_ADDR	L4_DST_PORT	PROTOCOL	BYTES	PACKETS	FIRST_SWITCHED	LAST_SWITCHED	INFO	JSON	PROFILE
158	0	7	::1	59089	::1	3000	6	3967	35	1447154752	1447154752	localhost	BLOB	tcp
159	0	7	::1	59090	::1	3000	6	3806	33	1447154752	1447154752	localhost	BLOB	tcp
160	0	7	::1	59091	::1	3000	6	3517	33	1447154752	1447154752	localhost	BLOB	tcp
466	0	7	::1	59420	::1	3000	6	43876	263	1447154784	1447154784	localhost	BLOB	tcp
467	0	7	::1	59421	::1	3000	6	1835	15	1447154784	1447154784	localhost	BLOB	tcp
468	0	7	::1	59422	::1	3000	6	1840	15	1447154784	1447154784	localhost	BLOB	tcp
469	0	7	::1	59423	::1	3000	6	1840	15	1447154784	1447154784	localhost	BLOB	tcp

Flow cache

IPFIX

IPFIX全称为IP Flow Information Export，意为IP流信息导出协议。IPFIX为IETF工作组根据思科Netflow v9的基础上发展而来，目前为IETF主推的流信息导出协议标准。

IPFIX协议也可以称为Netflow v10版本，从使用原理和架构上和Netflow并无区别，IPFIX相较于传统的Netflow支持了更多特性：

- 支持UDP、TCP、SCTP三种协议作为IPFIX的传输层协议载体。
- 支持模板类，可以相较于传统的Netflow导出更多的流信息。
- 支持变长信息元素和企业字段，可以导出Netflow中不具有的一些私有信息。

小结

SANGFOR 深信服科技

- 性能指标、测试工具及网络排障：

带宽，吞吐，时延，丢包，抖动，新建，并发

netperf，RFC2544，思伯伦

排障场景，排障工具

SANGFOR
深信服科技

THANK YOU

2018深信服科技