上篇文章说到，网络层的作用范围是主机到主机，而传输层的作用范围细分到了主机中的应用进程到应用进程。

一台主机上可能会同时运行多个应用进程，而传输层就是用端口号来区分同一个主机上不同的应用进程的，那么本篇文章来好好聊聊端口那些事儿~

一、端口的概述

在OSI第2层和第3层中，我们看到需要一个地址来标识与通信方式有关的必要元素。

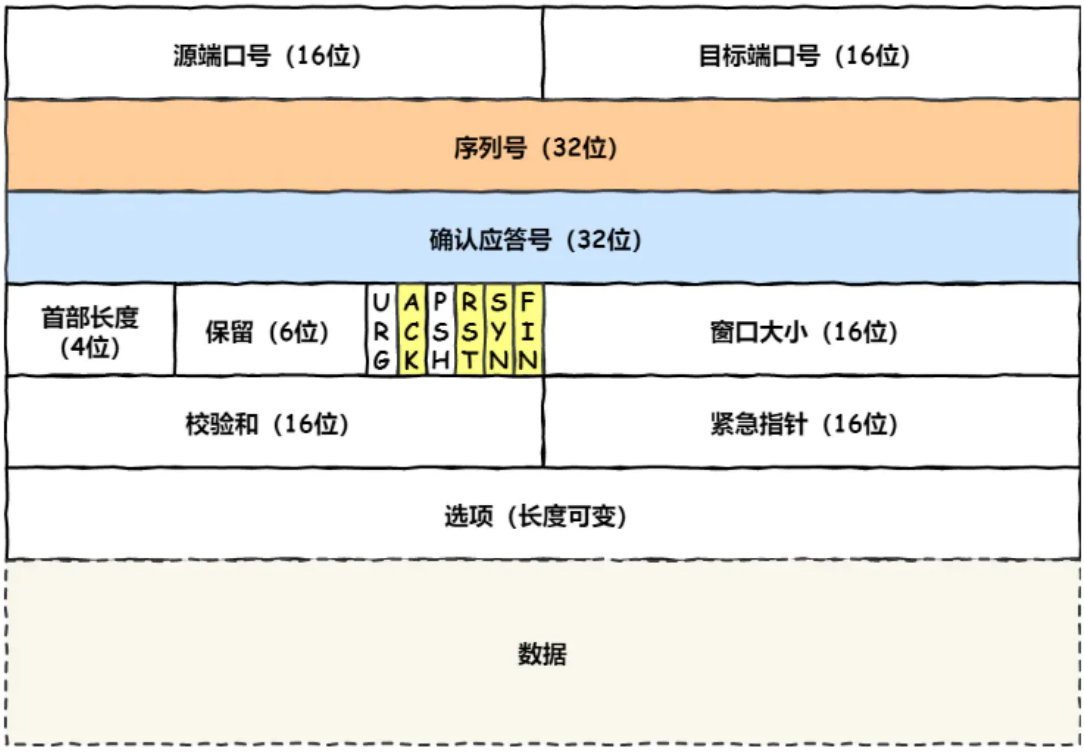
- MAC地址在第2层中标识网卡
- IP地址在第3层中标识机器在网络中的地址。

那么，在第4层中，使用的地址是port，即端口。

分层	标识
数据链路层地址	MAC地址
网络层地址	IP地址
传输层地址	端口号

在TCP的首部中用两个字节来表示端口，即一台主机最大允许 65536 个端口号的。下图为TCP首部格式，可以看到一开始就是2个字节的源端口号和2个字节的目标端口号，对应到之前学习的源MAC地址和目的MAC地址、源IP地址和目的IP地址。

TCP 头部格式



如果将ip地址比作一间房子，端口就是出入这间房子的门。房子一般只有几个门，但是一台主机端口最多可以有65536个。

65536个端口，已经很多了，不是吗？因此，理论上我们在一台电脑上最多可以运行65536个网络应用程序。

有了 IP 协议，数据包可以顺利的被传输到对应 IP 地址的主机，当主机收到一个数据包时，应该把这个数据包交给哪个应用程序进行处理呢？这台主机可能运行多个应用程序，比如处理HTTP请求的web服务器Nginx，Redis服务器， 读写MySQL服务器的客户端等。

传输层就是用端口号来区分同一个主机上不同的应用程序的。操作系统为有需要的进程分配端口号，当目标主机收到数据包以后，会根据数据报文首部的目标端口号将数据发送到对应端口的进程。

我们先来看看平时经常碰到的一些端口号：

应用程序	端口号
http	80
https	443
ssh	22
tomcat	8080
ftp	20/21
dns	53
mysql	3306

以上端口号五花八门，可用的范围也很大，达到了0~65535，实际上它被划分为了三种类型或三种范围，我们先说第一种：熟知端口号。

二、端口号分类-熟知端口号

熟知端口也被称为保留端口，由专门的机构由IANA分配和控制，范围为0~1023。

为什么要设置熟知端口号呢，顾名思义，为了能让客户端能随时找到自己，服务端程序的端口必须要是固定的。

我们访问HTTP的网站时，比如访问百度 `http://www.baidu.com` ，实际上就是向百度的服务器上的80端口发起请求，即在默认使用80端口的情况下可以不显式地写出来，我们可以执行 `curl -v http://www.baidu.com` 命令来看下访问百度地址的效果：

```
[C:\~]$ curl -v http://www.baidu.com
* Rebuilt URL to: http://www.baidu.com/
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0*
* TCP_NODELAY set
* Connected to www.baidu.com (180.101.49.12) port 80 (#0)
> GET / HTTP/1.1
> Host: www.baidu.com
> User-Agent: curl/7.55.1
> Accept: */*
>
< HTTP/1.1 200 OK
< Accept-Ranges: bytes
< Cache-Control: private, no-cache, no-store, proxy-revalidate, no-transform
< Connection: keep-alive
< Content-Length: 2381
< Content-Type: text/html
< Date: Sat, 11 Sep 2021 03:54:41 GMT
< Etag: "588604c8-94d"
< Last-Modified: Mon, 23 Jan 2017 13:27:36 GMT
< Pragma: no-cache
< Server: bfe/1.0.8.18
< Set-Cookie: BDORZ=27315; max-age=86400; domain=.baidu.com; path=/
<
{ [2381 bytes data]
100  2381  100  2381    0     0   2381      0  0:00:01 --:--:--  0:00:01 30525
* Connection #0 to host www.baidu.com left intact
<!DOCTYPE html>
<!--STATUS OK--><html> <head><meta http-equiv=content-type content=text/html;chars
ferrer><link rel=stylesheet type=text/css href=http://s1.bdstatic.com/r/www/cache/
cc> <div id=wrapper> <div id=head> <div class=head_wrapper> <div class=s_form> <di
o1.png width=270 height=129> </div> <form id=form name=f action=//www.baidu.com/s
=utf-8> <input type=hidden name=f value=8> <input type=hidden name=rsv_bp value=1>
 class="bg s_ipt_wr"><input id=kw name=wd class=s_ipt value maxlength=255 autocomp
惧害涓€涓? class="bg s_btn"></span> </form> </div> </div> <div id=u1> <a href=http
ame=tj_trhao123 class=mnav>hao123</a> <a href=http://map.baidu.com name=tj_trmap c
<a href=http://tieba.baidu.com name=tj_trtieba class=mnav>璐村惛</a> <noscript> <a
du.com%2f%3fbdorz_come%3d1 name=tj_login class=lb>鐧诲綍</a> </noscript> <script>d
RIComponent(window.location.href+ (window.location.search === "" ? "?" : "&")+ "bd
.com/more/ name=tj_briicon class=bri style="display: block;">鏇村  浜y 搁</a> </di
m>鍏充簬鐧惧害</a> <a href=http://ir.baidu.com>About Baidu</a> </p> <p id=cp>&copy
```

可以看到默认访问的是80端口号，再比如访问HTTPS网站，我们还以百度为例，执行 `curl -v https://www.baidu.com` 命令，可以验证默认是443端口号：

```
[C:\~]$ curl -v https://www.baidu.com
* Rebuilt URL to: https://www.baidu.com/
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0*
Trying 180.101.49.11...
* TCP_NODELAY set
* Connected to www.baidu.com (180.101.49.11) port 443 (#0)
* schannel: SSL/TLS connection with www.baidu.com port 443 (step 1/3)
* schannel: checking server certificate revocation
* schannel: sending initial handshake data: sending 184 bytes...
* schannel: sent initial handshake data: sent 184 bytes
* schannel: SSL/TLS connection with www.baidu.com port 443 (step 2/3)
* schannel: failed to receive handshake, need more data
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0* sc
hannel: SSL/TLS connection with www.baidu.com port 443 (step 2/3)
* schannel: encrypted data got 53
* schannel: encrypted data buffer: offset 53 length 4096
* schannel: SSL/TLS connection with www.baidu.com port 443 (step 2/3)
* schannel: encrypted data got 3779
* schannel: encrypted data buffer: offset 3779 length 4096
* schannel: SSL/TLS connection with www.baidu.com port 443 (step 2/3)
* schannel: encrypted data got 347
* schannel: encrypted data buffer: offset 347 length 4096
* schannel: sending next handshake data: sending 126 bytes...
* schannel: SSL/TLS connection with www.baidu.com port 443 (step 2/3)
* schannel: encrypted data got 226
* schannel: encrypted data buffer: offset 226 length 4096
* schannel: SSL/TLS handshake complete
* schannel: SSL/TLS connection with www.baidu.com port 443 (step 3/3)
* schannel: stored credential handle in session cache
> GET / HTTP/1.1
> Host: www.baidu.com
> User-Agent: curl/7.55.1
> Accept: */*
>
* schannel: client wants to read 102400 bytes
* schannel: encdata_buffer resized 103424
* schannel: encrypted data buffer: offset 0 length 103424
* schannel: encrypted data got 1412
* schannel: encrypted data buffer: offset 1412 length 103424
* schannel: failed to decrypt data, need more data
```

> 那么是否非80的HTTP网站就不行呢？不是的，你可以使用其他的端口来发布你的网站，比如你自己的个人网站是 **http://www.oursnail.cn:8888** ，此时就需要显式写出端口号了，而如果你使用的是80端口则不需要，只需要输入 **http://www.oursnail.cn** 即可，所以你会选择80还是8888作为你的门户网站访问的端口呢？

三、端口号分类-已登记端口号

已登记的端口不受 IANA 控制，不过由 IANA 登记并提供它们的使用情况清单。它的范围为 1024～49151。

为什么是 49151 这样一个魔数？其实是取的端口号最大值 65536 的 3/4 减 1（49151 = 65536 * 0.75 - 1）。可以看到已登记的端口占用了大约 75% 端口号的范围。

已登记的端口常见的端口号有：

1. MySQL：3306

2. Redis：6379
3. MongoDB：27017

四、端口号分类-临时端口号

我们看到，我们熟知的端口都是用于服务器应用程序的，但是客户端应用程序呢？需要为它们分配端口吗？显然需要，只是端口号是操作系统随机分配给客户端的，范围至少是要大于1024的。

对于服务器应用程序，由于它一直在监听，比如常见的80或443端口，是大家比较熟知的，因此需要给服务器应用分配比较固定的熟知端口号或已登记端口号。

而客户端应用程序将仅在运行时监听。因此，只要操作系统知道哪个客户端应用程序位于哪个端口即可，所以客户端应用程序只需要使用临时端口号即可。

如果应用程序没有调用bind()函数将socket绑定到特定的端口上，也就是说没有像比如tomcat固定监听在8080端口上时，那么TCP和UDP会为该socket分配一个唯一的临时端口。

IANA 将 49152～65535 范围的端口称为临时端口（ephemeral port）或动态端口（dynamic port），也称为私有端口（private port），这些端口可供本地应用程序临时分配端口使用。

> 实际临时端口的范围是由不同操作系统来选择的，一般Linux内核端口范围为32768~60999，可以通过调整 `/proc/sys/net/ipv4/ip_local_port_range` 来实现变更，比如在需要主动发起大量连接的服务器上（比如网络爬虫、正向代理）可以调整 `ip_local_port_range` 的值，允许更多的可用端口。

五、查看端口被什么进程监听占用

我们已经知道，当机器上运行一个应用程序时，会占据一个端口号，我们如何查看端口号被什么进程监听占用呢？我们以centos操作系统为例。

我们执行 `netstat -antp` 命令查看：

```
[root@VM-0-13-centos ~]# netstat -antp
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 172.17.0.13:6379        0.0.0.0:*               LISTEN      29974/./bin/redis-s
tcp        0      0 127.0.0.1:3307          0.0.0.0:*               LISTEN      15030/mariadbd
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      18582/nginx: worker
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      8132/sshd
tcp        0      0 0.0.0.0:443             0.0.0.0:*               LISTEN      18582/nginx: worker
tcp        0      0 0.0.0.0:389             0.0.0.0:*               LISTEN      4344/slapd
tcp        0      0 172.17.0.13:39994       169.254.0.55:8080       TIME_WAIT   -
tcp        0      0 172.17.0.13:39208       169.254.0.55:5574       ESTABLISHED 18656/YDService
tcp        0      0 172.17.0.13:22          222.94.73.49:3745       ESTABLISHED 311/sshd: root@pts/
tcp6       0      0 :::18888                :::*                    LISTEN      14775/httpd
tcp6       0      0 :::3306                 :::*                    LISTEN      5515/mariadbd
tcp6       0      0 127.0.0.1:8015          :::*                    LISTEN      28812/java
tcp6       0      0 :::9999                 :::*                    LISTEN      22257/java
tcp6       0      0 :::8019                 :::*                    LISTEN      28812/java
tcp6       0      0 :::21                   :::*                    LISTEN      12506/vsftpd
tcp6       0      0 :::23                   :::*                    LISTEN      12913/xinetd
tcp6       0      0 :::8088                 :::*                    LISTEN      28812/java
tcp6       0      0 :::389                  :::*                    LISTEN      4344/slapd
tcp6       0      0 172.17.0.13:36978       172.17.0.13:3306        ESTABLISHED 28812/java
tcp6       0      0 172.17.0.13:36662       172.17.0.13:3306        ESTABLISHED 28812/java
tcp6       0      0 172.17.0.13:36156       172.17.0.13:3306        ESTABLISHED 28812/java
tcp6       0      0 172.17.0.13:3306        172.17.0.13:36978       ESTABLISHED 5515/mariadbd
tcp6       0      0 172.17.0.13:3306        172.17.0.13:36880       ESTABLISHED 5515/mariadbd
tcp6       0      0 172.17.0.13:36930       172.17.0.13:3306        ESTABLISHED 28812/java
tcp6       0      0 172.17.0.13:36880       172.17.0.13:3306        ESTABLISHED 28812/java
tcp6       0      0 172.17.0.13:3306        172.17.0.13:36156       ESTABLISHED 5515/mariadbd
tcp6       0      0 172.17.0.13:3306        172.17.0.13:36930       ESTABLISHED 5515/mariadbd
tcp6       0      0 172.17.0.13:3306        172.17.0.13:36662       ESTABLISHED 5515/mariadbd
[root@VM-0-13-centos ~]#
```

可以看到，我在机器上运行了很多的应用程序，他们目前处于LISTEN监听状态，如Redis占用的是默认的6379端口，mariadb数据库占用了3307和3306端口，还有ssh的22端口，nginx监听着80和443端口对外提供HTTP或HTTPS服务。

我也可以进行筛选，比如我只要展示前10行即可，可以输入`netstat -antp | head -n 10`：

```
[root@VM-0-13-centos ~]# netstat -antp | head -n 10
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 172.17.0.13:6379        0.0.0.0:*               LISTEN      29974/./bin/redis-s
tcp        0      0 127.0.0.1:3307          0.0.0.0:*               LISTEN      15030/mariadbd
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      18582/nginx: worker
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      8132/sshd
tcp        0      0 0.0.0.0:443             0.0.0.0:*               LISTEN      18582/nginx: worker
tcp        0      0 0.0.0.0:389             0.0.0.0:*               LISTEN      4344/slapd
tcp        0      0 172.17.0.13:22          115.57.127.137:52518    ESTABLISHED 3751/sshd: unknown
tcp        0      0 172.17.0.13:39208       169.254.0.55:5574       ESTABLISHED 18656/YDService
[root@VM-0-13-centos ~]#
```

也可以查询比如3306端口目前的占用情况，输入`netstat -antp | grep :3306`：

```
[root@VM-0-13-centos ~]# netstat -antp | grep :3306
tcp6       0      0 :::3306                 :::*                    LISTEN      5515/mariadbd
tcp6       0      0 172.17.0.13:36978       172.17.0.13:3306        ESTABLISHED 28812/java
tcp6       0      0 172.17.0.13:36662       172.17.0.13:3306        ESTABLISHED 28812/java
tcp6       0      0 172.17.0.13:3306        172.17.0.13:36978       ESTABLISHED 5515/mariadbd
tcp6       0      0 172.17.0.13:3306        172.17.0.13:36880       ESTABLISHED 5515/mariadbd
tcp6       0      0 172.17.0.13:36930       172.17.0.13:3306        ESTABLISHED 28812/java
tcp6       0      0 172.17.0.13:36880       172.17.0.13:3306        ESTABLISHED 28812/java
tcp6       0      0 172.17.0.13:3306        172.17.0.13:36930       ESTABLISHED 5515/mariadbd
tcp6       0      0 172.17.0.13:37882       172.17.0.13:3306        ESTABLISHED 28812/java
tcp6       0      0 172.17.0.13:3306        172.17.0.13:36662       ESTABLISHED 5515/mariadbd
tcp6       0      0 172.17.0.13:3306        172.17.0.13:37882       ESTABLISHED 5515/mariadbd
[root@VM-0-13-centos ~]#
```

通过这个就可以看出来3306是被mariadb进程监听占用中。

此外，也可以通过lsof命令查看端口被哪个进程监听占用中，因为在linux中，一切皆文件，例如输入命令 `lsof -n -P -i:80` 查看80端口是被哪个进程监听的：

```
[root@VM-0-13-centos ~]# lsof -n -P -i:80
COMMAND   PID USER   FD   TYPE    DEVICE SIZE/OFF NODE NAME
nginx   18582 root    6u  IPv4 239527480      0t0  TCP *:80 (LISTEN)
nginx   29632 root    6u  IPv4 239527480      0t0  TCP *:80 (LISTEN)
[root@VM-0-13-centos ~]#
```

可以看到，80端口是被nginx所监听占用中。

六、检查对方端口是否可正常访问

在工作中，我们会经常需要查看对方端口是否打开，以确认网络是否可达，比如调试接口时，我们发现接口调用超时，这个时候我们如何手工验证对方端口是否打开或网络是否可达呢?

我们可以使用nc或telenet命令非常方便的查看到对方端口是否打开或者网络是否可达。

```
[root@VM-0-13-centos ~]# nc -v 220.181.38.251 443
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Connected to 220.181.38.251:443.
^C
[root@VM-0-13-centos ~]#
[root@VM-0-13-centos ~]#
[root@VM-0-13-centos ~]#
[root@VM-0-13-centos ~]#
[root@VM-0-13-centos ~]#
[root@VM-0-13-centos ~]#
[root@VM-0-13-centos ~]#
[root@VM-0-13-centos ~]#
[root@VM-0-13-centos ~]#
[root@VM-0-13-centos ~]#
[root@VM-0-13-centos ~]# telnet 220.181.38.251 80
Trying 220.181.38.251...
Connected to 220.181.38.251.
Escape character is '^]'.
```

当对方网络未打开或网络不通时会返回"Connection refused" 错误或“Connection timed out”等错误，此时就需要提供这样的截图告诉运维人员：呐，IP为XX主机到XX主机端口不通，应该有网络限制或路由不通，烦请检查!