# 免流的研究报告

（空白页（其实没什么用））

2018/4/6

# 前言

其实最近快中考了，回顾了一下初中的日子，发现我好像就从来没有认真学习过。（浪费时间啊）

别人在搞算法，一整本《算法导论》下去，什么二叉树，二叉排序树，红黑树，二叉查找树，二叉平衡树，字典树，决策树， 哈夫曼树，AVL树，B树，B+树，最小生成树，哈密尔顿环，欧拉回路，A\*，DFSID，DBFS，Bellman-Ford（SPFA），Floyd-Warshall，Kneser，Prim，Kruskal，匈牙利算法，Ford-Fulkerson，并查集，带权并查集，双hash，hash优化，树状数组，线段树，线段树合并，Treap随机平衡二叉树，Splay伸展树，Scapegoat Tree替罪羊树，块状数组，块状链表，树套树，线段树套线段树，线段树套平衡树，平衡树套线段树，可并堆，左偏树，配对堆，KDtree，四分树，可持久化线段树（主席树），可持久化平衡树，可持久化块状数组，KMP，AC自动机（不是自动AC机(w)），后缀数组，后缀树，后缀自动机，manacher，最短路，次短路，K短路，连通分量（Tarjan）割点，最大流，最小割，费用流，分数规划，树上倍增，公共祖先，树链剖分，树的分治算法（点分治，边分治，\*动态？树分治），动态树（LCT，\*树分块），虚树，prufer编码，欧几里得算法，筛法，快速幂，斐蜀定理，更相减损术，lucas定理，乘法逆元，矩阵乘法，sg函数，中国剩余定理，单纯型线性规划，辛普森积分，模线性方程组，莫比乌斯反演，置换群，快速傅里叶变换（FFT+NTT），大步小步法（BSGS），EXBSGS，一般DP，背包DP，状压DP，区间DP，环形DP，树形DP，期望DP，数位DP，插头DP，子集DP，斜率优化，矩阵优化，决策单调性，数位动态规划，斯坦纳树，背包九讲，四边形不等式优化，环 + 外向树上的动态规划，插头动态规划，旋转卡壳，半平面交，pick定理，扫描线，莫队算法，\*树上莫队，随机增量法，二分，三分法（求偏导），分治，CDQ分治，离线（查询），ST表，Dinic，矩阵树定理，高斯消元，多项式求逆，K-D树，LCT，Pollard\_rho，带花树，2-SAT，ISAP，度限制生成树，最小乘积生成树，树上倍增，树链剖分LCA，Prufer编码，树的同构，LCT，Top-Tree，基环树，dominator tree，平面图转对偶图，单调栈，Lazy思想。ZKW线段树，划分树，卡特兰数，斯特林数，贝尔数……………………………………………..

但是总感觉初中搞过的东西就这样扔了也不好，所以就想着把最近的研究整理一下，就变成了这个东西。虽然这么说，免流也只是我最近在搞的一个东西，其实我以前还搞过更刺激的（和别人现场抢服务器啥的）。免流也是一个重要的东西（对于我这个看个视频流量就飞了的），而且最近免流也和谐的特别厉害，可能是GFW升级的同时顺便升级了三大运营商的服务器。

扯点看似不相关的。最近翻墙VPN和谐的特别厉害，而且GFW好像屏蔽了从国外到国内的所有连接（除80/8080等）。我这边很多国外的服务器都不能反向连回国内。而且国内这里也不能出去连国外的，不是开了防火墙就是被GFW干掉。Linux放在一边，windows的sharedaccess是绝对没有打开的。像之前那个台湾的还有一个泰国的都是这样，除了80其他什么端口都上不去，99%是被GFW干掉了。而且最近我还发现Lantern也不行了，它的服务器被墙掉了。所以现在翻墙的形势非常险峻。我现在用的T\*\*\*dVPN也只有ICMP能在4G上连接到，速度还只有40Kb/s。最重要的是GFW好像已经能扫SSR了。[详情见。](https://sber.us/%E4%B8%BA%E4%BD%95-shadowsocks-%E8%A6%81%E5%BC%83%E7%94%A8%E4%B8%80%E6%AC%A1%E6%80%A7%E9%AA%8C%E8%AF%81-ota/)

所以这个研究也可以说是很有意义的（别的方面）。

# 预备知识

#### 关于网络传输

数据是如何在网络中传输的呢？我相信很多人是不清楚的。首先，系统在传输网络的时候，最常用的是**TCP/IP**协议。你可能会在连接🡪属性里看到过这个东西。



网络适配器属性图

**TCP/IP** 全称 Transmission Control Protocol/Internet Protocol，中译名为传输控制协议/因特网互联协议，又名网络通讯协议，是**Internet最基本的协议**、Internet国际互联网络的基础。

TCP/IP很好的对应了网络的OSI模型。

简单的来说，TCP/IP协议在传输数据时可以分为以下几个部分。

就拿HTTP举例。





HTTP包

TCP包

IP包

以太帧

以太网在二层链路上通过**MAC地址**来唯一标识网络设备，并且实现局域网上网络设备之间的通信。MAC地址也叫物理地址，大多数网卡厂商把MAC地址烧入了网卡的ROM中。发送端使用接收端的MAC地址作为目的地址。以太帧封装完成后会通过物理层转换成比特流在物理介质上传输。

在这之下，IP包又分为很多种，主要有TCP、UDP、ICMP、ARP等协议。TCP、UDP属于高级网络协议，ICMP、ARP等属于基本网络协议。HTTP在传输的时候采用的就是**TCP协议**。

#### 关于HTTP格式

**HTTP**（HyperText Transfer Protocol）是网络传输**常用的协议**，如果在网址中出现 http:// 字样，则表明该网站使用的是**HTTP协议**。（HTTPS则是SSL加密安全的HTTP传输）（注：以下都为HTTP 1.0标准）我们可以从一次抓包过程中看出**HTTP传输**的过程。





HTTP最开始由**客户端发出请求**，**服务器处理请求**后返回数据，便**关闭TCP连接**。通常HTTP消息包括客户机向服务器的请求消息和服务器向客户机的响应消息。这两种类型的消息由一个起始行，一个或者多个头域，一个只是头域结束的空行和可选的消息体组成。HTTP的头域包括通用头，请求头，响应头和实体头四个部分。每个头域由一个域名，冒号（:）和域值三部分组成。域名是大小写无关的，域值前可以添加任何数量的空格符，头域可以被扩展为多行，在每行开始处，使用至少一个空格或制表符。

请求消息

请求消息的第一行为下面的格式：

Method[SP]Request-URI[SP]HTTP-Version[CRLF]

Method表示对于Request-URI完成的方法，这个字段是大小写敏感的，包括OPTIONS、 GET、HEAD、POST、PUT、DELETE、TRACE。方法GET和HEAD应该被所有的通用WEB服务器支持，其他所有方法的实现是可选的。 **GET方法**取回由Request-URI标识的信息。HEAD方法也是取回由Request-URI标识的信息，只是可以在响应时，不返回消息体。 POST方法可以请求服务器接收包含在请求中的实体信息，可以用于提交表单，向新闻组、BBS、邮件群组和数据库发送消息。

[SP]表示空格。Request-URI遵循URI格式，在此字段为星号（\*）时，说明请求并不用于某个特定的资源地址，而是用于服务器本身。HTTP- Version表示支持的HTTP版本，例如为HTTP/1.1。[CRLF]表示换行+回车符。请求头域允许客户端向服务器传递关于请求或者关于客户机的附加信息。请求头域可能包含下列字段Accept、Accept-Charset、Accept-Encoding、Accept-Language、 Authorization、From、Host、If-Modified-Since、If-Match、If-None-Match、If- Range、If-Range、If-Unmodified-Since、Max-Forwards、Proxy-Authorization、 Range、Referer、User-Agent。对请求头域的扩展要求通讯双方都支持，如果存在不支持的请求头域，一般将会作为实体头域处理。

典型的请求消息：

GET /STH/somedata.exe HTTP/1.0

Host: www.ppp.com

Accept:\*/\*

Pragma: no-cache

Cache-Control: no-cache

Referer: http://wwwppp.com/download.php

User-Agent: Mozilla/4.04[en](Win95;I;Nav)

Range: bytes=554554-

其中

**Host头域指定请求资源的Intenet主机和端口号**，必须表示请求url的原始服务器或网关的位置。HTTP/1.1请求必须包含主机头域，否则系统会以400状态码返回。如果服务器上有多个网站，**Host字段是必须的**。

User-Agent头域的内容包含发出请求的用户信息。比如浏览器信息等。网站根据此字段决定返回什么样式的网页（用H-S就能把User-Agent换成iPhone的就能获取百度云真实下载地址）

响应消息

响应消息的第一行为下面的格式：

HTTP-Version[SP]Status-Code[SP]Reason-Phrase[CRLF]

HTTP-Version表示支持的HTTP版本，例如为HTTP/1.1。Status-Code是一个三个数字的结果代码。Reason-Phrase给 Status-Code提供一个简单的文本描述。Status-Code主要用于机器自动识别，Reason-Phrase主要用于帮助用户理解。 Status-Code的第一个数字定义响应的类别，后两个数字没有分类的作用。第一个数字可能取5个不同的值：

1xx:信息响应类，表示接收到请求并且继续处理

2xx:处理成功响应类，表示动作被成功接收、理解和接受

3xx:重定向响应类，为了完成指定的动作，必须接受进一步处理

4xx:客户端错误，客户请求包含语法错误或者是不能正确执行

5xx:服务端错误，服务器不能正确执行一个正确的请求

响应头域允许服务器传递不能放在状态行的附加信息，这些域主要描述服务器的信息和Request-URI进一步的信息。响应头域包含Age、 Location、Proxy-Authenticate、Public、Retry-After、Server、Vary、Warning、WWW- Authenticate。对响应头域的扩展要求通讯双方都支持，如果存在不支持的响应头域，一般将会作为实体头域处理。

典型的响应消息：

HTTP/1.0 200 OK

Date: Mon,31 Dec 2001 04:25:57 GMT

Server: Apache/1.3.14(Unix)

Content-type: text/html

Last-modified: Tue,17Apr200106:46:28GMT

Etag: "a030f020ac7c01:1e9f"

Content-length: 39725426

Content-range: bytes554554-40279979/40279980

你可以在Wiki上找到[详细的字段信息](https://en.wikipedia.org/w/index.php?title=List_of_HTTP_header_fields)。

# 二．流量的走向

#### 1. APN

 你可能会在手机上设置->其它->移动网络->APN中发现过**APN设置**，但是APN到底是什么呢？

APN指一种**网络接入技术**，是通过手机上网时必须配置的一个参数，它决定了手机通过哪种接入方式来访问网络。

对于手机用户来说，可以访问的外部网络类型有很多，例如：Internet、WAP网站、集团企业内部网络、行业内部专用网络。而不同的接入点所能访问的范围以及接入的方式是不同的，网络侧如何知道手机激活以后要访问哪个网络从而分配哪个网段的IP呢，这就要靠APN来区分了，即APN决定了用户的手机通过哪种接入方式来访问什么样的网络。



一般来说，你可能会在手机上看到的APN有三种，也是最重要的三种：**CTWAP、CTNET、CTLTE（电信为CT，移动为CM）**，CTWAP为以前上网时用的，CTNET是正常的网络访问，CTLTE是针对4G优化。

#### 2. 代理服务器

如果你接触过一点免流，你会对 **10.0.0.200** 这个IP非常熟悉（移动为 10.0.0.172 ）。因为这个IP是流量的出口（猜测），所有的流量都经过这个IP。

（【配置环境：连接Android共享网络， route add 10.0.0.0 mask 255.0.0.0 192.168.44.1】）

 使用4G时网络状况

如果你trace流量的走向，你会发现除了第一个出口其它都被屏蔽了（应该是电信的防护措施）。



 使用WiFi时的网络状况

在4G情况下流量的走向是无从得知的。但是不知道哪个大神找出了电信的**代理服务器地址**：10.0.0.200

（这个地址应该是**流量的出口**，真实的情况我们无从得知）



扫一下可以发现21/ftp、22/ssh、23/telnet、53/dns等端口都开着。（等哪天哪个大神破了密码就厉害了）

但是这都不是重点，重点是80和8080端口

试一下可以发现，80和8080在这里都是代理端口。但是能连接那个端口取决于APN的设置，如果是CTWAP的话能连接8080端口，否则只能连接80端口。（针对我的Android4.4手机？）

这里拿8080端口举例

#### 免流的原理

如果我们发送**正常的HTTP请求**

当然，得到的也是正常的回复，同时也会**扣流量**。（注：代理服务器自动添加请求头）

但是，电信有很多项目是**不用流量**的，比如说爱看4G，爱玩4G，你在访问这些内容是是不扣流量的，计费服务器怎么判断哪些是要流量哪些是不要流量的呢？（**据推测，计费服务器与代理服务器分开**）

最主要是通过判断**HTTP中的Host头**来做到这一点。（目前）如果Host字段指向电信的**免费流量**的服务器，则计费服务器就**不扣费**。我们知道Host字段指定了**当前访问的网站**，代理服务器判断Host来发送数据包来达到代理的效果。（路由器也是这个效果）一旦Host字段改变，代理服务器将要访问的网站也就改变。这种情况下我们**无法伪造Host字段**。**因为如果使Host字段指向电信的免流服务器，虽然不扣流量，但是我们也无法访问我们自己的网站。**

另外，如果要访问的不是网页，而是要使用像RDP（3389/tcp）或者DNS（53/udp）或者SSH（22/tcp）等等协议的时候，虽然有的用的也是TCP，但是它们不是HTTP，所以数据包内**不含HTTP中的Host字段**，代理服务器就不能通过判断HTTP中的Host字段的方法来判断流量的去向。这个时候流量在出口的时候就会转为判断**IP头**或Ethernet头来找到数据包的去向。

但是这个时候问题来了。

如下图所示



我们会发现，如果往10.0.0.200代理发送请求，会多出很多代理服务器自己加的头，而如果直接发送相同的内容至我们的服务器，则HTTP头就保持原样。

这也就决定了一点，**如果直接往除80外（我的Android上端口为80时，即使直接访问服务器，依然会被HTTP代理）端口发送数据，不经过代理服务器**（上图为电脑，与手机上使用不一样），**流量就会不经HTTP代理直接被发送**，尽管它是HTTP数据包。



但是，这并不意味着发出去的流量就不计费。

如果流量经过10.0.0.200代理（一般通过80、8080访问网站的流量都经过代理（上图为电脑，与手机不同）），计费服务器就可以（很好地）**通过Host判断流量的真实去向**，来决定要不要扣费。但是，如果流量并没有经过HTTP代理，而是直接被发送出去（意思是数据包不经任何修改和判断），那么问题就来了。这是代理服务器判断的是IP头，不在我们修改的数据包范围内（见一.1），所以计费服务器不知道这个流量是什么类型的，所以它就会判断这个流量是不是属于HTTP协议（判断是否Valid），是的话再判断Host字段。如果Host字段指向免流服务器它就不扣费。（到哪天开始判断IP头就完了）

这就成功达到了伪装的效果。**这样我们就可以通过直接发送数据包至服务器，并且把它伪装成HTTP数据包，且里面的Host字段指向电信的免流服务器，这样电信的计费服务器判断我们这个是HTTP数据包Host还是它的免流服务器所以它就不计费，而且电信的代理服务器认为我们这个可能不是HTTP数据包不采用HTTP代理不判断Host字段。**

这就是免流中**云免**的原理。

就是修改自己的数据包使它比较符合HTTP格式，且Host指向电信的免流服务器，但是里面的数据却是我们自己的。这样我们就可以访问自己的内容但是计费服务器又不扣费。

例如：

GET /111.apk HTTP/1.0

Host: cdn.4g.play.cn //电信的免流服务器

Cookie: OUR\_DATA……

但是怎么实现使数据包符合HTTP格式，里面的数据又是我们自己的呢？

最常用的有两款，OpenVPN和Shadowsocks(R)。

#### 附：免流的另一种方法

我们知道流量如果通过10.0.0.200会被代理，同时也会被计费服务器检测Host字段判断是不是免流服务器来决定是不是需要扣费。

但是其实没有那么简单。

我们知道HTTP的格式是多样的，代理服务器不一定要通过判断Host来决定流量的走向。

比如，我们可以在URL中加入地址， GET http://www.baidu.com/ HTTP/1.0，这也是一个合法的请求。如果URL和Host指向不同的服务器，那么代理服务器就会根据程序的写法（地区不同）决定流量走URL指向的地址还是Host指向的地址。

不仅如此，同时代理服务器还会在判断HTTP格式合法的同时截断HTTP内容。HTTP的标准换行为[CRLF]，即回车换行，也就是”\r\n”。代理服务器不会判断到”\r”时就停止这一行的读取，认为内容读取完，而是完整的读取”\r\n”。但是计费服务器不是这样，可能是为了节省时间，它判断”\r”就认为这一行结束，读取下一行。这是有决定性的差距。比如 Host: cdn.4g.play.cn\rHost: www.baidu.com\r\n 这一行，代理服务器会认为地址为www.baidu.com，而计费服务器会认为地址为cdn.4g.play.cn从而不扣流量，这也是一种免流的方法。主要是通过tiny、hap、snmp等软件代理10.0.0.200的8080端口的代理服务器，利用代理服务器和计费服务器对待一些特殊字符和特殊情况的差距来免流。

现有模式双H，\r截断，\t绕过等等等。不过都已失效。现在未有新的突破。

# 免流的方法

 要使流量符合HTTP标准，又隐藏自己的数据，普通的HTTP代理做不到。所以要借助一些软件。

#### OpenVPN

在这里要先纠正一下大部分人对于VPN的观念。

VPN（Virtual Private Network）即**虚拟专用网络**，虚拟专用网络的功能是：**在公用网络上建立专用网络，进行加密通讯。在企业网络中有广泛应用。VPN网关通过对数据包的加密和数据包目标地址的转换实现远程访问。VPN主要用途不是翻墙，只是它可以用来翻墙。**

让外地员工访问到内网资源，利用VPN的解决方法就是在内网中架设一台VPN服务器。外地员工在当地连上互联网后，通过互联网连接VPN服务器，然后通过VPN服务器进入企业内网。为了保证数据安全，VPN服务器和客户机之间的通讯数据都进行了加密处理。有了数据加密，就可以认为数据是在一条专用的数据链路上进行安全传输，就如同专门架设了一个专用网络一样，但实际上VPN使用的是互联网上的公用链路，因此VPN称为虚拟专用网络，其实质上就是利用加密技术在公网上封装出一个数据通讯隧道。有了VPN技术，用户无论是在外地出差还是在家中办公，只要能上互联网就能利用VPN访问内网资源，这就是VPN在企业中应用得如此广泛的原因。



VPN其实就是接入VPN服务器所创建的虚拟网段，模拟网卡来使用。但是又因为它可以代理所有的网络数据，又有加密，所以它可以拿来翻墙，当然也可以拿来当免流工具使用。

使用安卓端的OpenVPN可接入VPN网络。

VPN设置中有一项http-proxy，即通过HTTP代理来访问VPN服务器。我们可以根据这个达到免流的效果。OpenVPN的http-proxy是通过向代理服务器发送“CONNECT 服务器地址 HTTP/1.0 ”来实现访问VPN服务器。所以我们可以伪造VPN服务器为cdn.4g.play.cn，同时自己写个代理使其把连接转向真正的VPN服务器。可以达到免流的效果。（配置超麻烦的我就不配了）

#### Shadowsocks or ShadowsocksR

SS和SSR本来就是为了翻墙而产生的工具，所以具有可以模拟请求的功能。所以安装好SSR之后，就可以通过SSR配置中的混淆HTTP填入免流Host来达到伪装Host的目的。

数据包格式如下

 GET /%01%48%29… HTTP/1.0

 Host: cdn.4g.play.cn

 ENCODED\_DATAS

唯一的缺点是SSR把数据附加在HTTP结束后的地方，可能会被认为是损坏的HTTP数据包。

推荐的SSR配置：

root@west398:~# cat /etc/shadowsocks.json

{

 "server": "0.0.0.0",

 "server\_port": \*\*\*,

 "local\_address": "127.0.0.1",

 "local\_port": 1081,

 "password": "\*\*\*\*\*\*\*\*\*\*",

 "timeout": 120,

 "udp\_timeout": 60,

 "method": "aes-256-cfb",

 "protocol": "auth\_aes128\_md5",

 "protocol\_param": "",

 "obfs": "http\_simple",

 "obfs\_param": "",

 "dns\_ipv6": false,

 "connect\_verbose\_info": 0,

 "redirect": "",

 "fast\_open": false,

 "workers": 5

}

但是不巧的是最近电信把SSR和谐了，可能是检测了SSR的加密或者其他特征，所以裸的SSR免流已经不行了。

所以我自己写了一个转发的工具，可以实现流量的完全模拟。

ML-Proxy.cpp

见 [https://mnihyc.com/blog/archives/1617#cpp-ml-proxy](https://mnihyc.com/blog/archives/1617%22%20%5Cl%20%22cpp-ml-proxy)

编译环境

Android: C4droid

Windows: DevC++ with -lws2\_32

Linux: g++

#### NEWWORLD（别到处乱发）

V2ray是我最近才发现的工具，它是在SS被和谐，SSR也被和谐后出现的工具。主要用途也是翻墙，但是它有自己独特的传输协议——VMess，它可以有效地防止数据包被电信或者GFW检测。

详细的信息你可以去[官网](https://www.v2ray.com/)找到。总而言之，这是一个非常强大的工具，但是配置并不像SSR那么人性化。

首先，v2ray在github上有源代码，也有编译好的版本，可以根据系统选择适合的版本直接下载使用。在这里贴一下我的服务器配置。

<DELETED>

# Enjoy yourself !




跳点比较不稳就是了，总之一个月能用上40GB