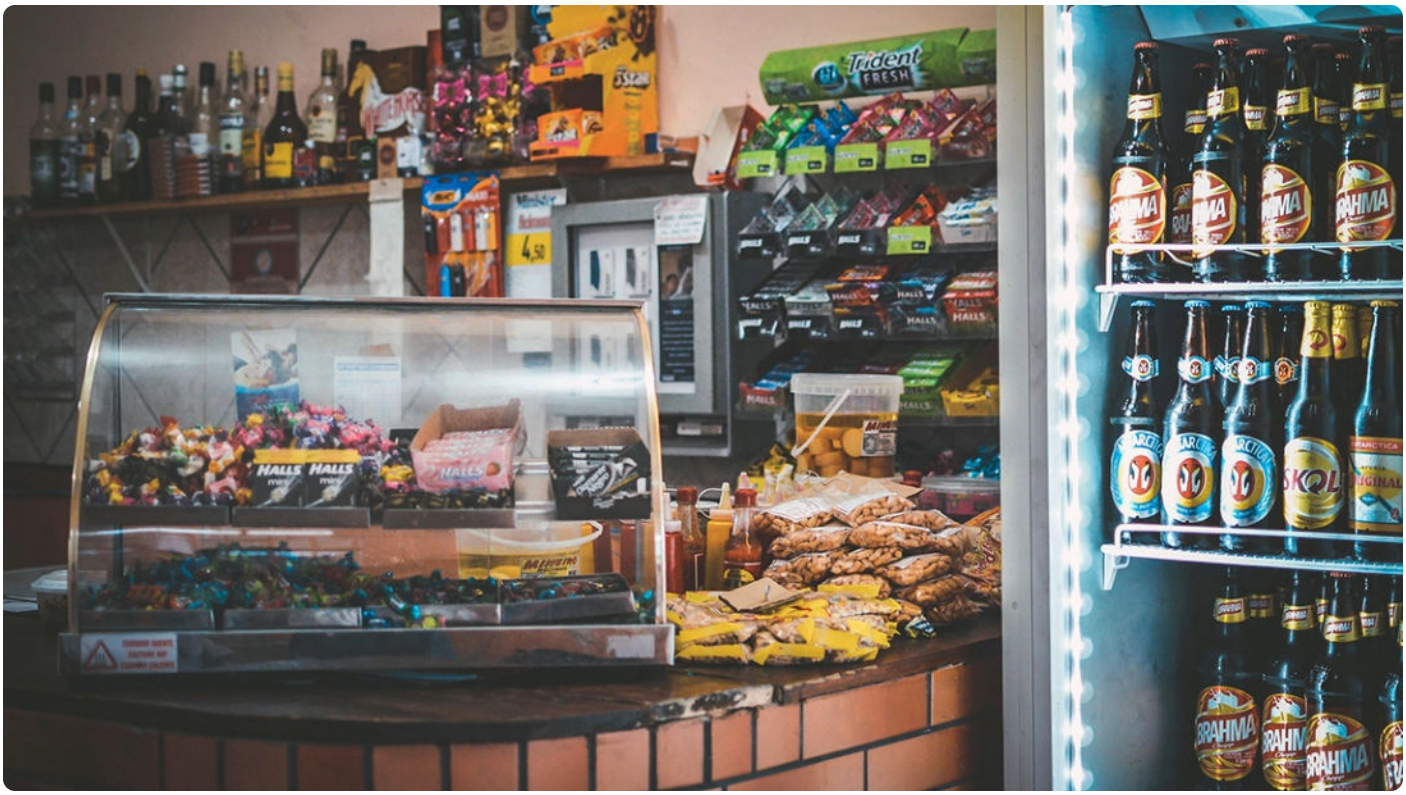
21 | 良心中间商：HTTP的代理服务

2019-07-15 Chrono

《透视HTTP协议》 课程介绍 >



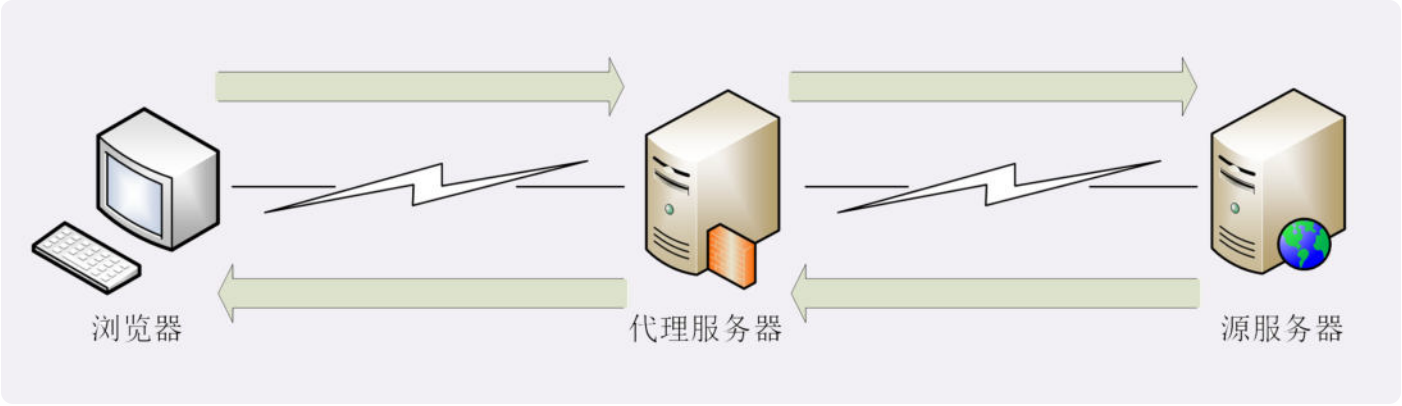
讲述：Chrono

时长 10:33 大小 12.09M

在前面讲 HTTP 协议的时候，我们严格遵循了 HTTP 的“请求 – 应答”模型，协议中只有两个互相通信的角色，分别是“请求方”浏览器（客户端）和“应答方”服务器。

今天，我们要在这个模型里引入一个新的角色，那就是HTTP 代理。

引入 HTTP 代理后，原来简单的双方通信就变复杂了一些，加入了一个或者多个中间人，但整体上来看，还是一个有顺序关系的链条，而且链条里相邻的两个角色仍然是简单的一对一通信，不会出现越级的情况。



链条的起点还是客户端（也就是浏览器），中间的角色被称为代理服务器（proxy server），链条的终点被称为源服务器（origin server），意思是数据的“源头”“起源”。

代理服务

“代理”这个词听起来好像很神秘，有点“高大上”的感觉。

但其实 HTTP 协议里对它并没有什么特别的描述，它就是在客户端和服务器原本的通信链路中插入的一个中间环节，也是一台服务器，但提供的是“代理服务”。

所谓的“代理服务”就是指服务本身不生产内容，而是处于中间位置转发上下游的请求和响应，具有双重身份：面向下游的用户时，表现为服务器，代表源服务器响应客户端的请求；而面向上游的源服务器时，又表现为客户端，代表客户端发送请求。

还是拿上一讲的“生鲜超市”来打个比方。

之前你都是从超市里买东西，现在楼底下新开了一家 24 小时便利店，由超市直接供货，于是你就可以在便利店里买到原本必须去超市才能买到的商品。

这样超市就不直接和你打交道了，成了“源服务器”，便利店就成了超市的“代理服务器”。

在第 4 讲中，我曾经说过，代理有很多的种类，例如匿名代理、透明代理、正向代理和反向代理。

今天我主要讲的是实际工作中最常见的反向代理，它在传输链路中更靠近源服务器，为源服务器提供代理服务。

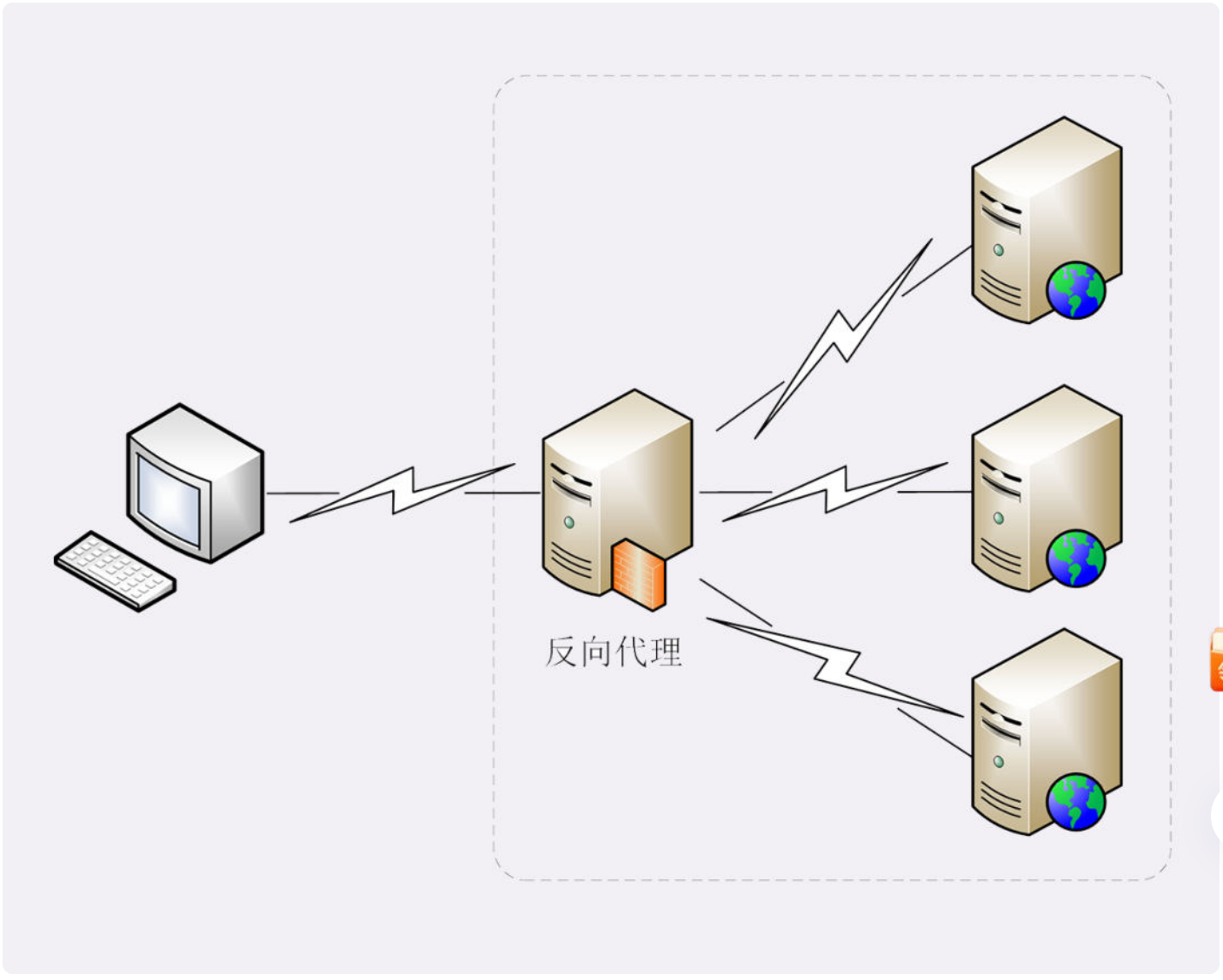
代理的作用

为什么要有代理呢？换句话说，代理能干什么、带来什么好处呢？

你也许听过这样一句至理名言：“计算机科学领域里的任何问题，都可以通过引入一个中间层来解决”（在这句话后面还可以再加上一句“如果一个中间层解决不了问题，那就再加一个中间层”）。TCP/IP 协议栈是这样，而代理也是这样。

由于代理处在 HTTP 通信过程的中间位置，相应地就对上屏蔽了真实客户端，对下屏蔽了真实服务器，简单的说就是“**欺上瞒下**”。在这个中间层的“小天地”里就可以做很多的事情，为 HTTP 协议增加更多的灵活性，实现客户端和服务器的“双赢”。

代理最基本的一个功能是**负载均衡**。因为在面向客户端时屏蔽了源服务器，客户端看到的只是代理服务器，源服务器究竟有多少台、是哪些 IP 地址都不知道。于是代理服务器就可以掌握请求分发的“大权”，决定由后面的哪台服务器来响应请求。



代理中常用的负载均衡算法你应该也有所耳闻吧，比如轮询、一致性哈希等等，这些算法的目标都是尽量把外部的流量合理地分散到多台源服务器，提高系统的整体资源利用率和性能。

在负载均衡的同时，代理服务还可以执行更多的功能，比如：

- **健康检查**：使用“心跳”等机制监控后端服务器，发现有故障就及时“踢出”集群，保证服务高可用；
- **安全防护**：保护被代理的后端服务器，限制 IP 地址或流量，抵御网络攻击和过载；
- **加密卸载**：对外网使用 SSL/TLS 加密通信认证，而在安全的内网不加密，消除加解密成本；
- **数据过滤**：拦截上下行的数据，任意指定策略修改请求或者响应；
- **内容缓存**：暂存、复用服务器响应，这个与第 20 讲密切相关，我们稍后再说。

接着拿刚才的便利店来举例说明。

因为便利店和超市之间是专车配送，所以有了便利店，以后你买东西就更省事了，打电话给便利店让它去帮你取货，不用关心超市是否停业休息、是否人满为患，而且总能买到最新鲜的。

便利店同时也方便了超市，不用额外加大店面就可以增加客源和销量，货物集中装卸也节省了物流成本，由于便利店直接面对客户，所以也可以把恶意骚扰电话挡在外面。

代理相关头字段

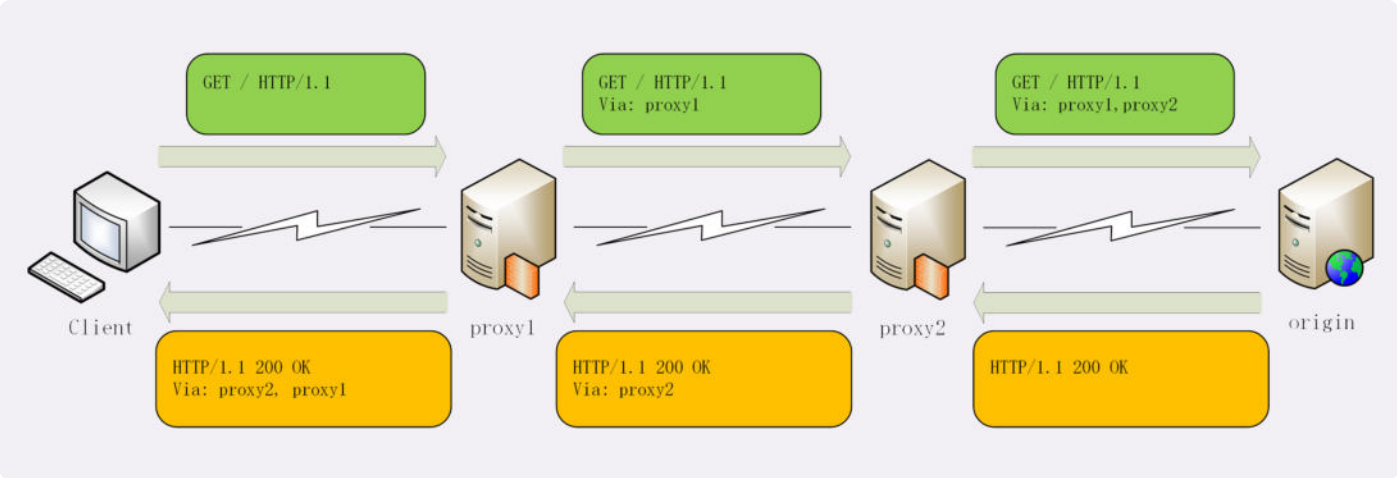
代理的好处很多，但因为它“欺上瞒下”的特点，隐藏了真实客户端和服务器，如果双方想要获得这些“丢失”的原始信息，该怎么办呢？

首先，代理服务器需要用字段“**Via**”标明代理的身份。

Via 是一个通用字段，请求头或响应头里都可以出现。每当报文经过一个代理节点，代理服务器就会把自身的信息追加到字段的末尾，就像是经手人盖了一个章。

如果通信链路中有很多中间代理，就会在 Via 里形成一个链表，这样就可以知道报文究竟走过了多少个环节才到达了目的地。

例如下图中有两个代理：proxy1 和 proxy2，客户端发送请求会经过这两个代理，依次添加就是“Via: proxy1, proxy2”，等到服务器返回响应报文的时候就要反过来走，头字段就是“Via: proxy2，proxy1”。



Via 字段只解决了客户端和源服务器判断是否存在代理的问题，还不能知道对方的真实信息。

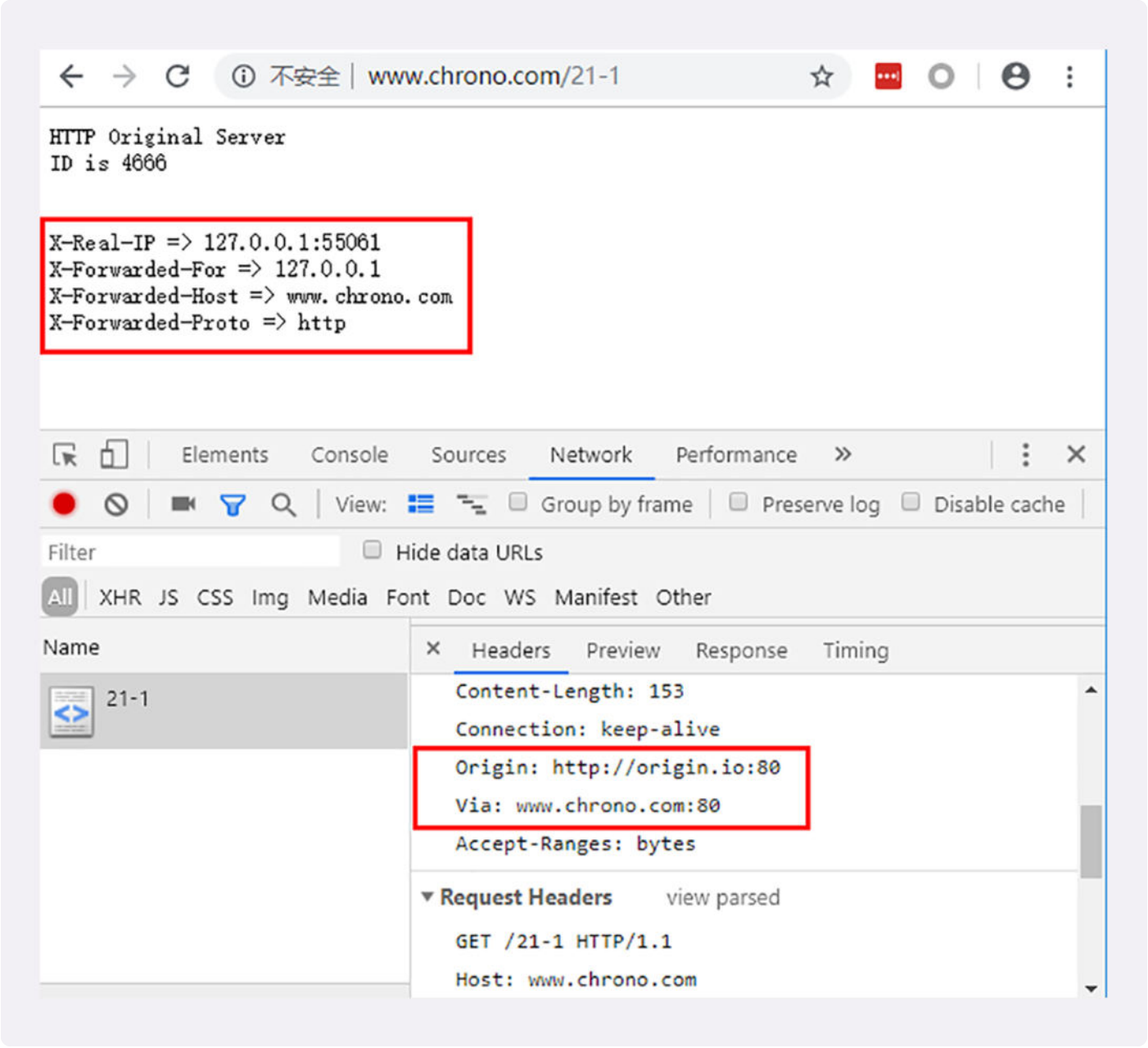
但服务器的 IP 地址应该是保密的，关系到企业的内网安全，所以一般不会让客户端知道。不过反过来，通常服务器需要知道客户端的真实 IP 地址，方便做访问控制、用户画像、统计分析。

可惜的是 HTTP 标准里并没有为此定义头字段，但已经出现了很多“事实上的标准”，最常用的两个头字段是“**X-Forwarded-For**”和“**X-Real-IP**”。

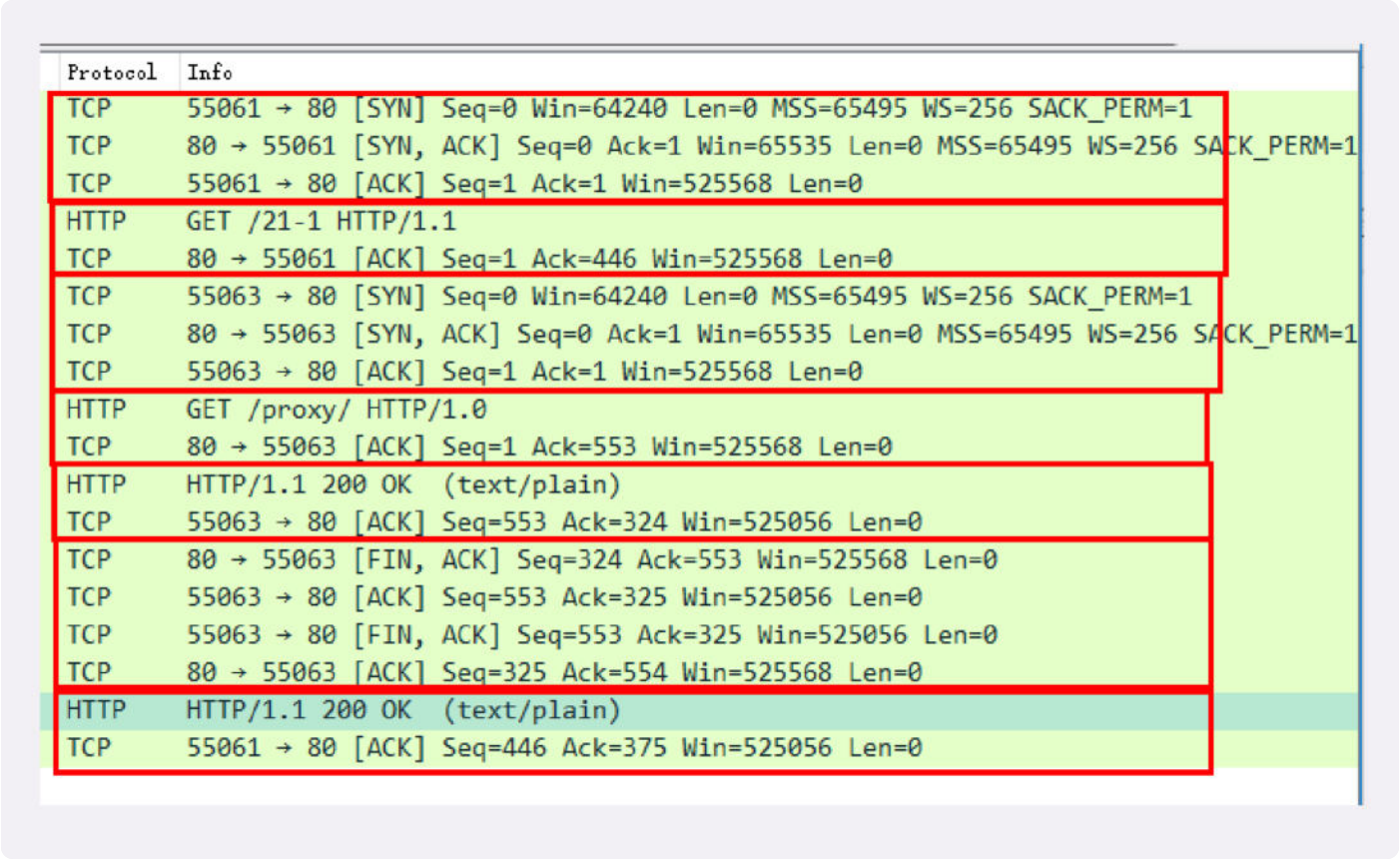
“X-Forwarded-For”的字面意思是“为谁而转发”，形式上和“Via”差不多，也是每经过一个代理节点就会在字段里追加一个信息。但“Via”追加的是代理主机名（或者域名），而“X-Forwarded-For”追加的是请求方的 IP 地址。所以，在字段里最左边的 IP 地址就是客户端的地址。

“X-Real-IP”是另一种获取客户端真实 IP 的手段，它的作用很简单，就是记录客户端 IP 地址，没有中间的代理信息，相当于是“X-Forwarded-For”的简化版。如果客户端和源服务器之间只有一个代理，那么这两个字段的值就是相同的。

我们的实验环境实现了一个反向代理，访问“http://www.chrono.com/21-1”，它会转而访问“http://origin.io”。这里的“origin.io”就是源站，它会在响应报文里输出“Via”“X-Forwarded-For”等代理头字段信息：



单从浏览器的页面上很难看出代理做了哪些工作，因为代理的转发都在后台不可见，所以我把这个过程用 Wireshark 抓了一个包：



Protocol	Info
TCP	55061 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=65495 WS=256 SACK_PERM=1
TCP	80 → 55061 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1
TCP	55061 → 80 [ACK] Seq=1 Ack=1 Win=525568 Len=0
HTTP	GET /21-1 HTTP/1.1
TCP	80 → 55061 [ACK] Seq=1 Ack=446 Win=525568 Len=0
TCP	55063 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=65495 WS=256 SACK_PERM=1
TCP	80 → 55063 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1
TCP	55063 → 80 [ACK] Seq=1 Ack=1 Win=525568 Len=0
HTTP	GET /proxy/ HTTP/1.0
TCP	80 → 55063 [ACK] Seq=1 Ack=553 Win=525568 Len=0
HTTP	HTTP/1.1 200 OK (text/plain)
TCP	55063 → 80 [ACK] Seq=553 Ack=324 Win=525056 Len=0
TCP	80 → 55063 [FIN, ACK] Seq=324 Ack=553 Win=525568 Len=0
TCP	55063 → 80 [ACK] Seq=553 Ack=325 Win=525056 Len=0
TCP	55063 → 80 [FIN, ACK] Seq=553 Ack=325 Win=525056 Len=0
TCP	80 → 55063 [ACK] Seq=325 Ack=554 Win=525568 Len=0
HTTP	HTTP/1.1 200 OK (text/plain)
TCP	55061 → 80 [ACK] Seq=446 Ack=375 Win=525056 Len=0

从抓包里就可以清晰地看出代理与客户端、源服务器的通信过程：

1. 客户端 55061 先用三次握手连接到代理的 80 端口，然后发送 GET 请求；
2. 代理不直接生产内容，所以就代表客户端，用 55063 端口连接到源服务器，也是三次握手；
3. 代理成功连接源服务器后，发出了一个 HTTP/1.0 的 GET 请求；
4. 因为 HTTP/1.0 默认是短连接，所以源服务器发送响应报文后立即用四次挥手关闭连接；
5. 代理拿到响应报文后再发回给客户端，完成了一次代理服务。

在这个实验中，你可以看到除了“X-Forwarded-For”和“X-Real-IP”，还出现了两个字段：“X-Forwarded-Host”和“X-Forwarded-Proto”，它们的作用与“X-Real-IP”类似，只记录客户端的信息，分别是客户端请求的原始域名和原始协议名。

代理协议

有了“X-Forwarded-For”等头字段，源服务器就可以拿到准确的客户端信息了。但对于代理服务器来说它并不是一个最佳的解决方案。

因为通过“X-Forwarded-For”操作代理信息必须要解析 HTTP 报文头，这对于代理来说成本比较高，原本只需要简单地转发消息就好，而现在却必须要费力解析数据再修改数据，会降低代理的转发性能。

另一个问题是“X-Forwarded-For”等头必须要修改原始报文，而有些情况下是不允许甚至不可能的（比如使用 HTTPS 通信被加密）。

所以就出现了一个专门的“代理协议”（The PROXY protocol），它由知名的代理软件 HAProxy 所定义，也是一个“事实标准”，被广泛采用（注意并不是 RFC）。

“代理协议”有 v1 和 v2 两个版本，v1 和 HTTP 差不多，也是明文，而 v2 是二进制格式。今天只介绍比较好理解的 v1，它在 HTTP 报文前增加了一行 ASCII 码文本，相当于又多了一个头。

这一行文本其实非常简单，开头必须是“PROXY”五个大写字母，然后是“TCP4”或者“TCP6”，表示客户端的 IP 地址类型，再后面是请求方地址、应答方地址、请求方端口号、应答方端口号，最后用一个回车换行（\r\n）结束。

例如下面的这个例子，在 GET 请求行前多出了 PROXY 信息行，客户端的真实 IP 地址是“1.1.1.1”，端口号是 55555。

复制代码

```
PROXY TCP4 1.1.1.1 2.2.2.2 55555 80\r\n
GET / HTTP/1.1\r\n
Host: www.xxx.com\r\n
\r\n
```

服务器看到这样的报文，只要解析第一行就可以拿到客户端地址，不需要再去理会后面的 HTTP 数据，省了很多事情。

不过代理协议并不支持“X-Forwarded-For”的链式地址形式，所以拿到客户端地址后再如何处理就需要代理服务器与后端自行约定。

小结

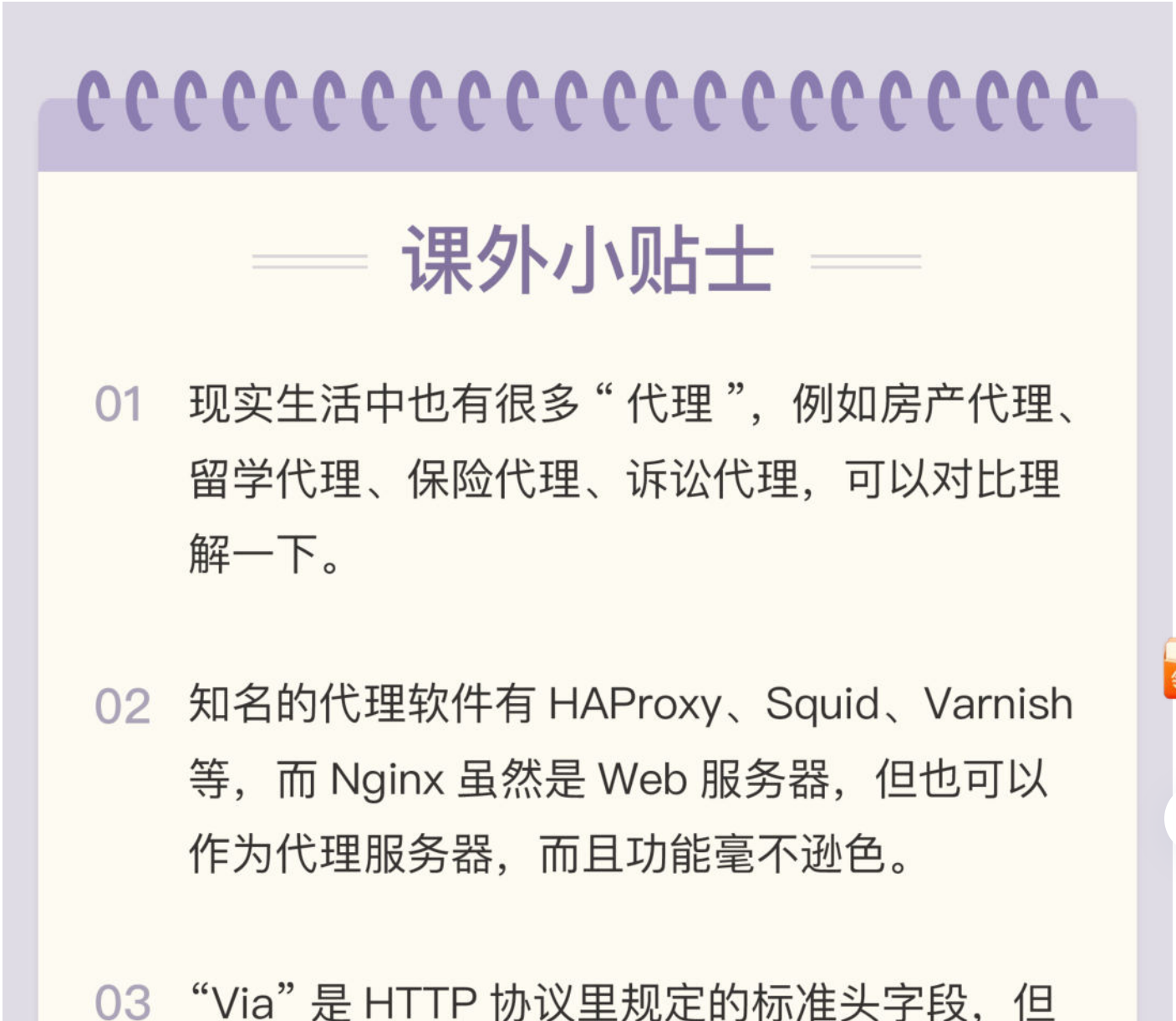
1. HTTP 代理就是客户端和服务器通信链路中的一个中间环节，为两端提供“代理服务”；

2. 代理处于中间层，为 HTTP 处理增加了更多的灵活性，可以实现负载均衡、安全防护、数据过滤等功能；
3. 代理服务器需要使用字段“Via”标记自己的身份，多个代理会形成一个列表；
4. 如果想要知道客户端的真实 IP 地址，可以使用字段“X-Forwarded-For”和“X-Real-IP”；
5. 专门的“代理协议”可以在不改动原始报文的情况下传递客户端的真实 IP。

课下作业

1. 你觉得代理有什么缺点？实际应用时如何避免？
2. 你知道多少反向代理中使用的负载均衡算法？它们有什么优缺点？

欢迎你把自己的学习体会写在留言区，与我和其他同学一起讨论。如果你觉得有所收获，也欢迎把文章分享给你的朋友。



有的服务器返回的响应报文里会使用“X-Via”，含义是相同的。

04 因为 HTTP 是明文传输，请求头很容易被窜改，所以“X-Forwarded-For”也不是完全可信的。

05 RFC7239 定义了字段“Forwarded”，它可以代替“X-Forwarded-For”“X-Forwarded-Host”等字段，但应用得不多。

赞 15　提建议

上一篇　20 | 生鲜速递：HTTP的缓存控制

下一篇　22 | 冷链周转：HTTP的缓存代理

精选留言 (47)

写留言

–W.LI–

2019–07–15

代理会增加链路长度，在代理上做一些复杂的处理。会很耗费性能，增加响应时间。
1.随机
2.轮询
3.一致性hash
4最近最少使用
5.链接最少

作者回复: great!

共 4 条评论>

45

BoyiKia

2020-05-11

老师，我发现前几节课，四次挥手的时候，是客户端主动先发 Fin信号， 今天实验结果，是源服务器，先给代理服务器发的 Fin信号。老师，我有点疑惑哈。到底是谁应该先发。还是说都可以呢。

作者回复: 这个是tcp协议的知识了，就是谁先断开连接的问题。

其实这个并没有强制要求客户端或者服务器先断开，通常都是客户端主动断开，但服务器也可以主动断开，比如超时、短连接、节约资源等等。

所以结论就是谁都可以，有空可以再补一下tcp的知识。

共 2 条评论>

15

Demon

2020-06-21

很多场景下，使用代理的目的就是为了匿名，不让对方知道请求/响应的来源在哪儿。除了在测试环境分析技术问题的场景，现实业务中有需要在报文中携带层层代理信息的应用case吗?

作者回复: 当然有了，互联网上很少有直连网站的，都要经过层层代理，这中间就免不了用代理协议。

很多代理并不是为了匿名，而是为了缓存。

8

火车日记

2019-07-16

1 补充几个，ip_hash 、最少连接数、最快连接数，根据场景应用
2 作为中转站，需要为上游和下游开启两个连接，大量并发请求，会出现性能瓶颈，应减少资源开销，加快响应速度，比如代理缓存，动静分离

作者回复: great!

7

Long

2020-02-23

老师好,文中
"服务器的 IP 地址应该是保密的，关系到企业的内网安全，所以一般不会让客户端知道。"
是不是可以认为,域名所对应的IP地址和真实服务器的IP地址是不一样的呢?因为真实服务器的地址一般都是私网的IP地址.

作者回复: 这个里面其实很复杂，首先网站外面会有cdn，然后入口会有反向代理，再后面才可能是真实的业务服务器。

服务器也可以安装多个网卡，一个网卡对外，一个网卡对内，这样有两个ip地址，分别对外对内。

4

lmingzhi
2019-07-15

老师，请问有什么检测http代理ip匿名性的手段?

是否只要检查请求头是否带有“X-Forwarded-For”和“X-Real-IP”及里面是否带有真实ip即可?

作者回复: 如果代理比较“善良”，就会用“X-Forwarded-For”和“X-Real-IP”告知客户端的真实ip，如果它是完全匿名，不提供这些字段，我们也没有办法，因为它就是一个真实的客户端。

4

sarah
2020-02-06

老师，对图中wireshark的抓包有个疑问: 每一次的http报文后面会跟着一个tcp报文，这个tcp报文是怎么产生的? 作用是什么? 例如，第一个http报文，HTTP GET/21-1 HTTP1.1后面的TCP 80—55061

作者回复: 这个是tcp协议的ack，表示收到报文的确认，如果你再多了解一些tcp的知识就会明白。

3

夏目
2019-12-03

老师，微服务里的网关算不算一个增强版的代理服务器呢

作者回复: 是的，可以算是一种微型的反向代理。

院长。
2019-07-16

老师后面会讲HTTP2.0吗

作者回复: 安全篇后的飞翔篇有http/2和http/3。

钱
2020-03-29

1：你觉得代理有什么缺点？实际应用时如何避免？
代理代理就是找她人代替你去打理一些事情，让他人代办事情你必须交代好沟通好，那效率自然会低一些，另外，如果代理出问题了，那你的事自然也办不成了，所以，可能存在单点问题，不过一般还好。

2：你知道多少反向代理中使用的负载均衡算法？它们有什么优缺点？
随机——简单，是否均匀看随机情况
轮询（一般轮询、加权轮询）——相对简单，也会考虑机器资源和性能的均衡性
哈希（一般哈希、一致性哈希、带虚拟节点的一致性哈希）——相对复杂，要求越公平就会越复杂，而且适当考虑了请求
哈希槽，和redis类似

只有能使请求尽可能的高效分发就行，请教一下VPN和代理，本质是否差不多？

作者回复:
1.对，代理的问题一个是成本，另一个就是信任。

2.最常用的就是这些了。

3.vpn和代理是两回事，它是一个虚拟的链路，有点像隧道，中间没有代理这样的角色，是直通的。

AKA三皮
2020-03-27

代理是个好东西，比如各种精细化的流量控制，灰度发布，同时微服务拆分后，服务治理的相关功能也可以下沉到代理去做，比如 限流、熔断。选个高性能的网络代理是王道，比如envoy

作者回复: 对，这个就是中间层的力量，也是软件开发的基本原则。

2

FF
2019-07-22

haproxy 那个代理协议那一行要客户端自己加上去的？如果客户端把这个加到 x-forward-for 里面，不用代理协议，那不是也可以解决代理去修改头部的问题？重点都是客户端先加上去这些信息 。这样看代理协议没啥优势啊，或者不是为了解决减少中间代理再去修改头的问题？盼复，感谢。

作者回复: 代理协议的那一行是代理服务器加的，客户端不需要参与。

代理协议的优势是简单，比http头好解析好处理，这对于代理服务器来说就能够提高转发效率。

你后面的理解基本正确。

2

Geek_6ea9af
2021-03-22

老师，请问在配置了正向代理之后，对于真正服务端的域名解析是发生在客户端还是代理端？该代理服务器仅做请求转发。

作者回复: 既然是代理，显然就帮客户端做所有事情了。客户端直接与代理通信，用不到域名解析，由代理实现对外收发信息。

或者反过来想一下，如果客户端解析域名，那么就拿到了真实ip地址，就会直连外网，也就不会走代理了。

1

xuan
2020-07-20

问题：
1."X-Forwarded-For”头的信息刚开始是客户端给的吗?
2.X-Forwarded-For在http头里，要修改就等于变动了原始的http报文，这个时候的修改动作发生在客户端还是代理服务器?
个人认为是客户端，两个动作应该都是在客户端

作者回复:
1.这个是代理服务器添加的，当然因为http协议很自由，客户端填也可以，但这就没有意义了。

2.这个头是为代理服务器准备的，含义是原始客户端，所以对于客户端来说就不需要这个头。

1

Maske
2020-06-17

1.a 代理服务器与上下游的通信机制也是http协议，因此增加了传输中的数据泄漏和篡改风险，可以使用https解决。b 如果代理服务器发生故障，会影响客户端的正常访问，可以增加代理服务器的数量，并配置代理服务器负载均衡算法。c 由于多了代理服务器的请求响应过程，增加了从源客户端和源服务器之间的来回时间。
2.轮询，加权轮询，随机法，加权随机法，源地址哈希法，最小连接数法

作者回复: 说的挺好，这段时间学习得很勤奋啊，也要适当休息。

共 2 条评论>

1

Aaron
2020-05-29

『因为通过“X-Forwarded-For”操作代理信息必须要解析 HTTP 报文头，这对于代理来说成本比较高，原本只需要简单地转发消息就好，而现在却必须要费力解析数据再修改数据，会降低代理的转发性能。』
问：代理协议的 PROXY 不也是一个头吗？同样需要对 header 的操作。它的优势是不是只在于操作的内容比 "X-Forwarded-For" 少一点而已？

『另一个问题是“X-Forwarded-For”等头必须要修改原始报文，而有些情况下是不允许甚至不可能的（比如使用 HTTPS 通信被加密）』
问：为什么“X-Forwarded-For”等头必须要修改原始报文呢？不是很理解。烦请老师解释一下，谢谢。

作者回复:
1.proxy头在第一行，结构很简单，而X-Forwarded-For在http头里，要有复杂的解析，特别是当http头很大的时候，成本就高了。

2.同样的原因，X-Forwarded-For在http头里，要修改就等于变动了原始的http报文，而proxy 协议是附加在外面的，不会改动原始报文。

1

张三
2020-04-11

HAProxy是不是就是MuleSoft…

作者回复: haproxy是专门的http代理软件，功能很强大，和Mulesoft的差距比较大吧。

1

Geek_f8a084
2020-02-06

转发是指的代理服务吗?

作者回复: 是的。

1

keep it simple
2020-01-23

学完了这一课，收获很大！给老师点赞~
第一个问题是：数据过滤——拦截上下行的数据，任意指定策略修改请求或者响应。这个不太理解。
第二个问题：X-Real-IP的例子，如果链路中有多个代理服务器，那只有第一个代理会加上X-Real-IP，后面的代理都不会再动这个字段了吧?

作者回复:
1.比如说，代理可以过滤某个关键字，如果出现有“密码”“银行卡”就把数据用xxx代替。

2.http报文可以随意修改，没有防窜改的手段，中间的代理可以任意修改，后面的也不知道，当然通常没有必要这么做，但不是说不可以改。

1

业余草
2019-07-24

落下了不少课，今天补上

作者回复: go on。

1