

拉勾教育

— 互联网人实战大学 —

《前端高手进阶》

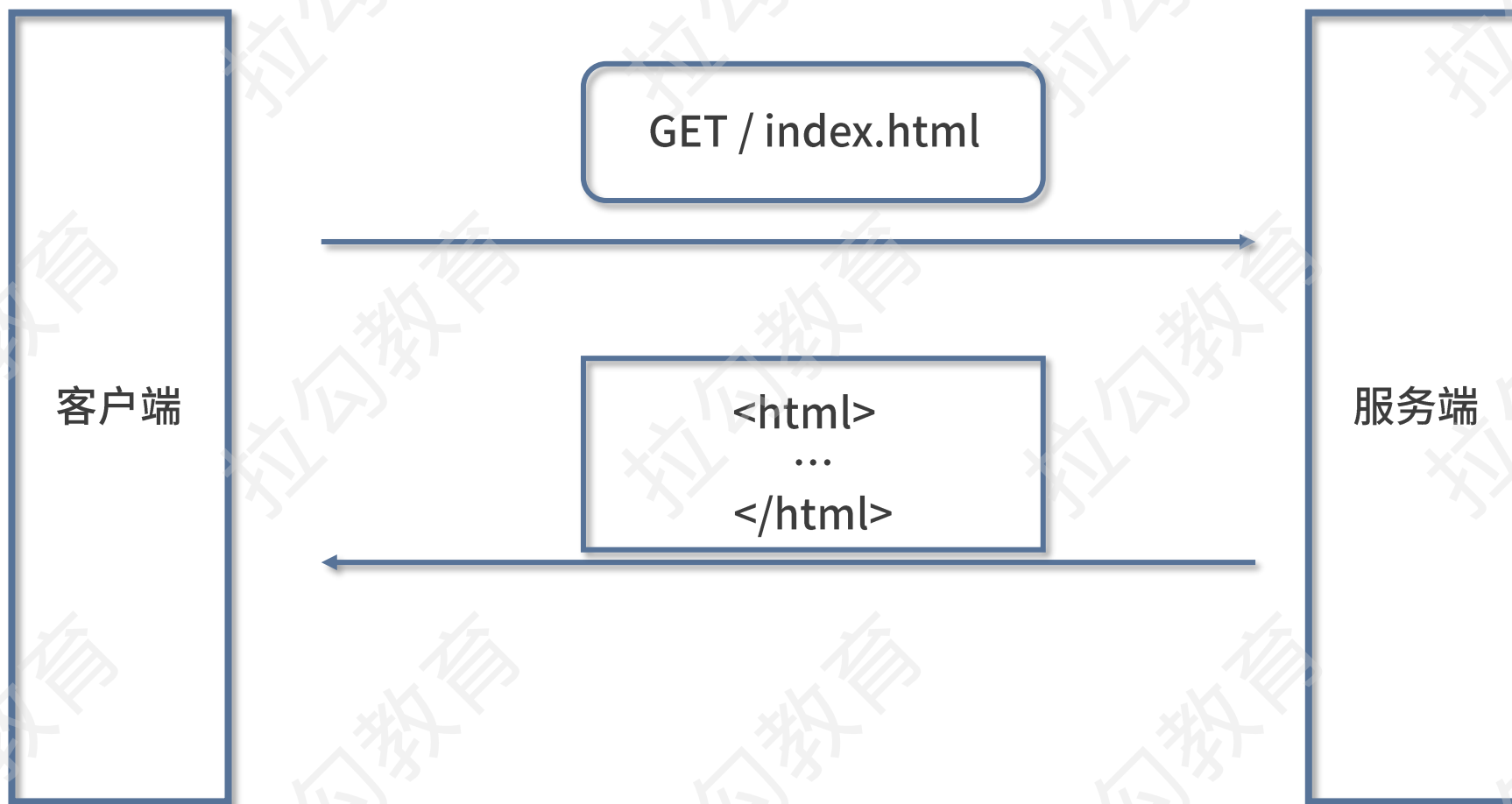
朱德龙 前中兴软创主任工程师

— 拉勾教育出品 —

第14讲：深入理解网络协议

1991 年 HTTP 正式诞生，当时的版本是 0.9

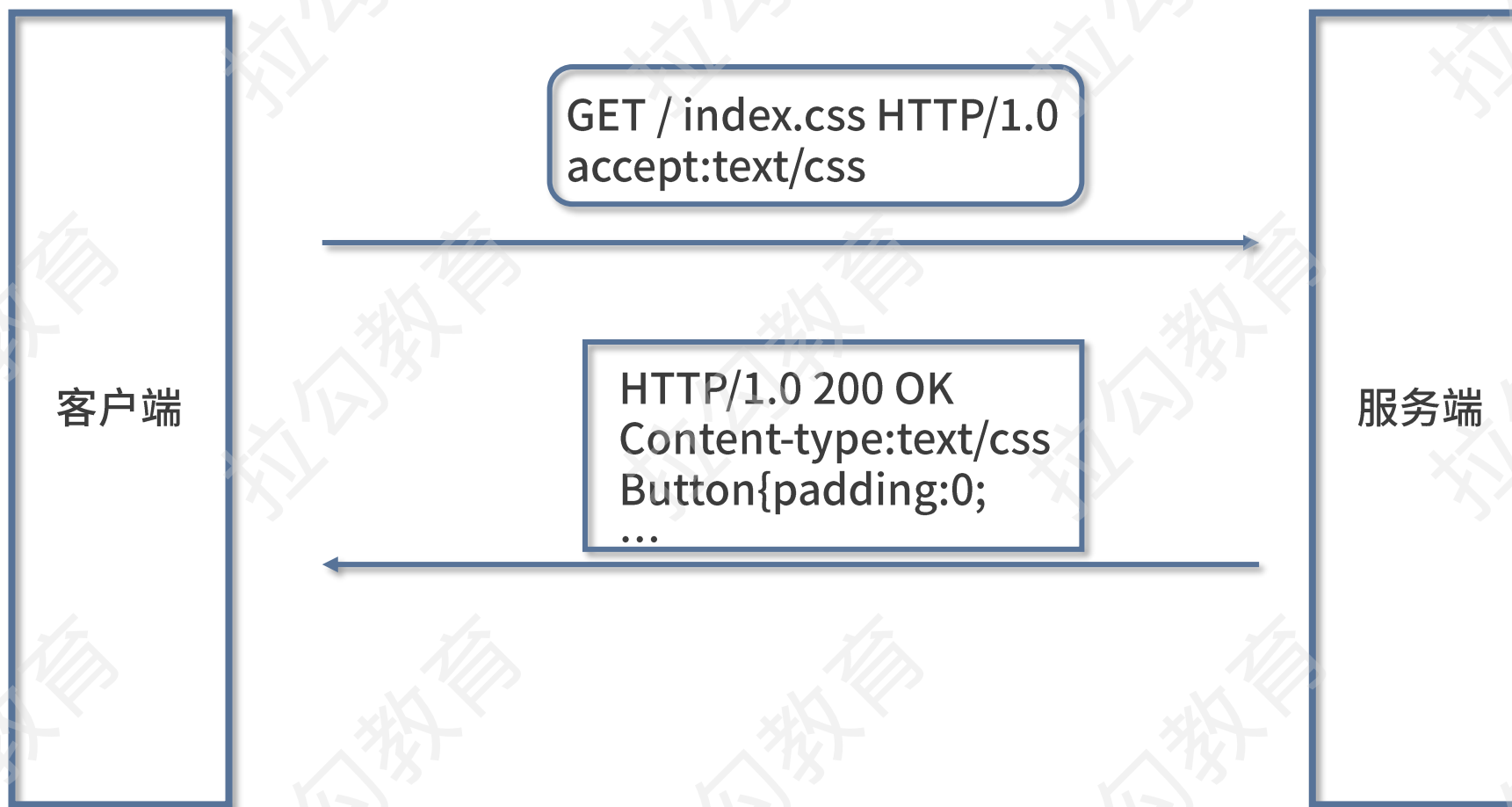
该协议的作用是传输超文本内容 HTML



浏览器希望通过 HTTP 来传输脚本、样式、图片、音频和视频等

不同类型的文件

在 1996 年 HTTP 更新的 **1.0** 版本中，针对上述问题，作出了重大改变



HTTP/1.0 每进行一次通信

都需要经历**建立连接**、**传输数据**和**断开连接**三个阶段

当一个页面引用了较多的外部文件时

这个建立连接和断开连接的过程就会**增加大量网络开销**

1999 年推出的 HTTP/1.1 版本增加了一个创建持久连接的方法

HTTP/1.1

拉勾教育

— 互联网人实战大学 —

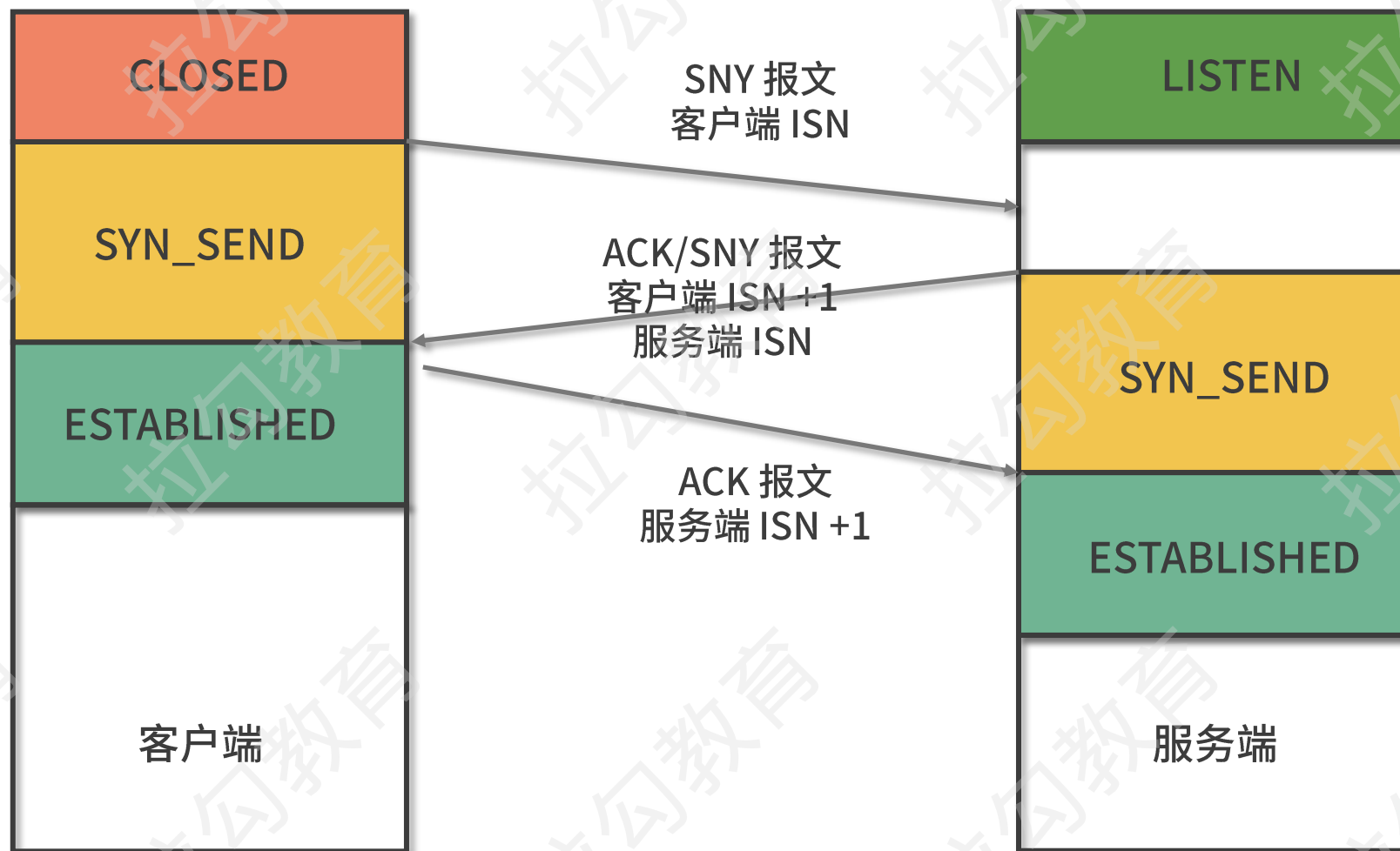


延申1: TCP 是怎样建立/断开连接的?

拉勾教育

— 互联网人实战大学 —

三次握手



延申1: TCP 是怎样建立/断开连接的?

拉勾教育

— 互联网人实战大学 —

为什么建立连接的时候需要进行三次握手呢?



第一次握手成功让服务端知道了客户端具有发送能力

第二次握手成功让客户端知道了服务端具有接收和发送能力

第三次握手成功让服务端知道客户端是否接收到了自己发送的消息

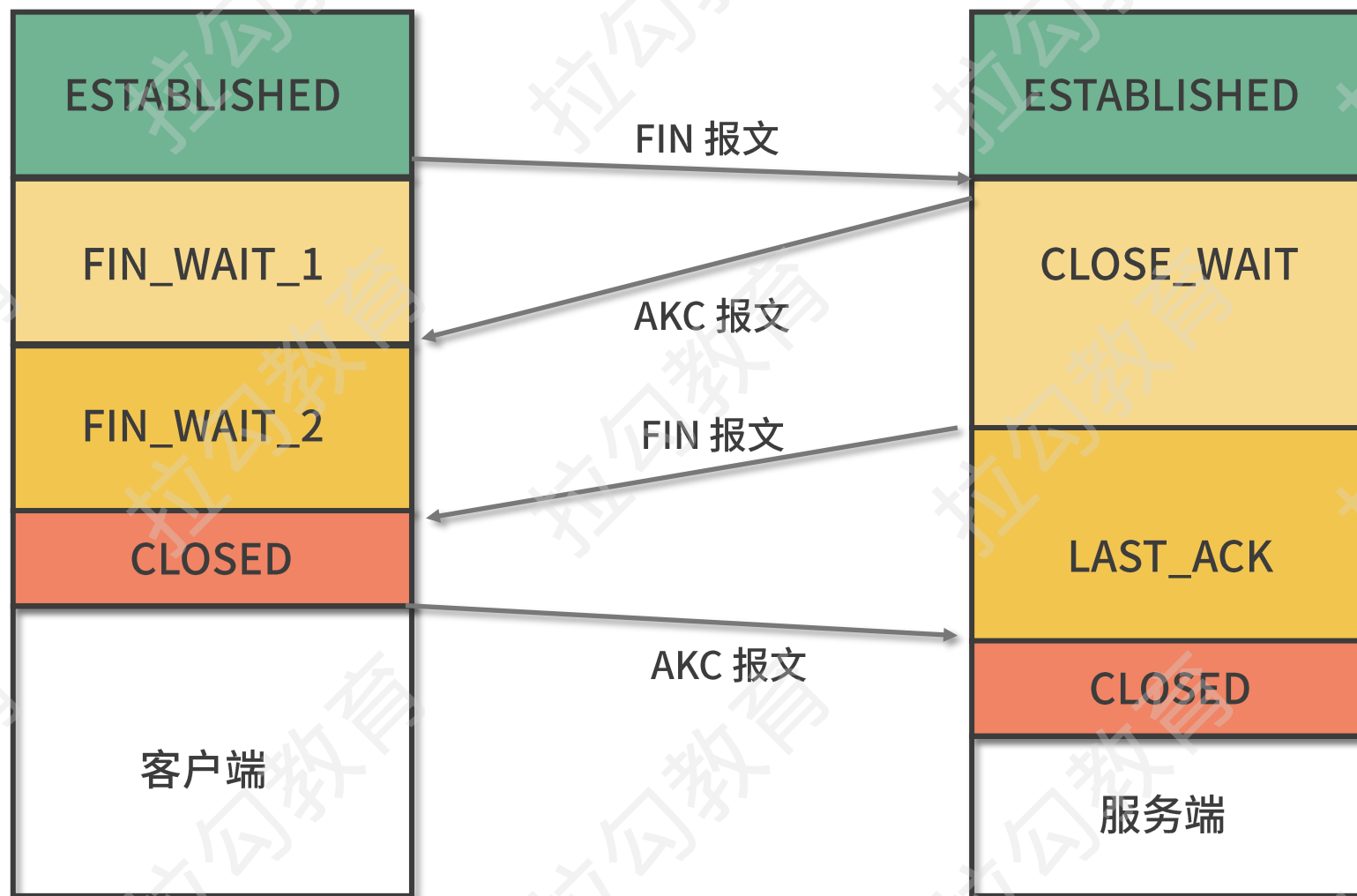
No.	Time	Source	Destination	Protocol	Length	Info
190...	139.149...	10.0.3.24	103.72.47.248	TCP	66	53898 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460
190...	139.179...	103.72.47.248	10.0.3.24	TCP	66	80 → 53898 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0
190...	139.179...	10.0.3.24	103.72.47.248	TCP	54	53898 → 80 [ACK] Seq=1 Ack=1 Win=65536 Len=0

延申1：TCP 是怎样建立/断开连接的？

拉勾教育

— 互联网人实战大学 —

四次挥手



延申1: TCP 是怎样建立/断开连接的?

拉勾教育

— 互联网人实战大学 —

为什么建立连接只通信了三次，而断开连接却用了四次



当服务端收到客户端的 FIN 报文后，发送的 ACK 报文只是用来**应答**的
并不表示服务端也希望立即关闭连接

当只有服务端把所有的报文**都发送完了**，才会发送 FIN 报文
告诉客户端可以断开连接了

266	2.686411	10.0.3.24	97.64.23.206	TCP	54 54108 → 443 [FIN, ACK] Seq=1 Ack=1 Win=255
285	2.904364	97.64.23.206	10.0.3.24	TCP	60 80 → 54109 [ACK] Seq=1 Ack=3 Win=237 Len=0
286	2.904824	97.64.23.206	10.0.3.24	TCP	60 443 → 54108 [FIN, ACK] Seq=1 Ack=2 Win=237
287	2.904890	10.0.3.24	97.64.23.206	TCP	54 54108 → 443 [ACK] Seq=2 Ack=2 Win=255 Len=0

HTTP/1.1 虽然通过长连接减少了大量创建/断开连接造成的性能消耗

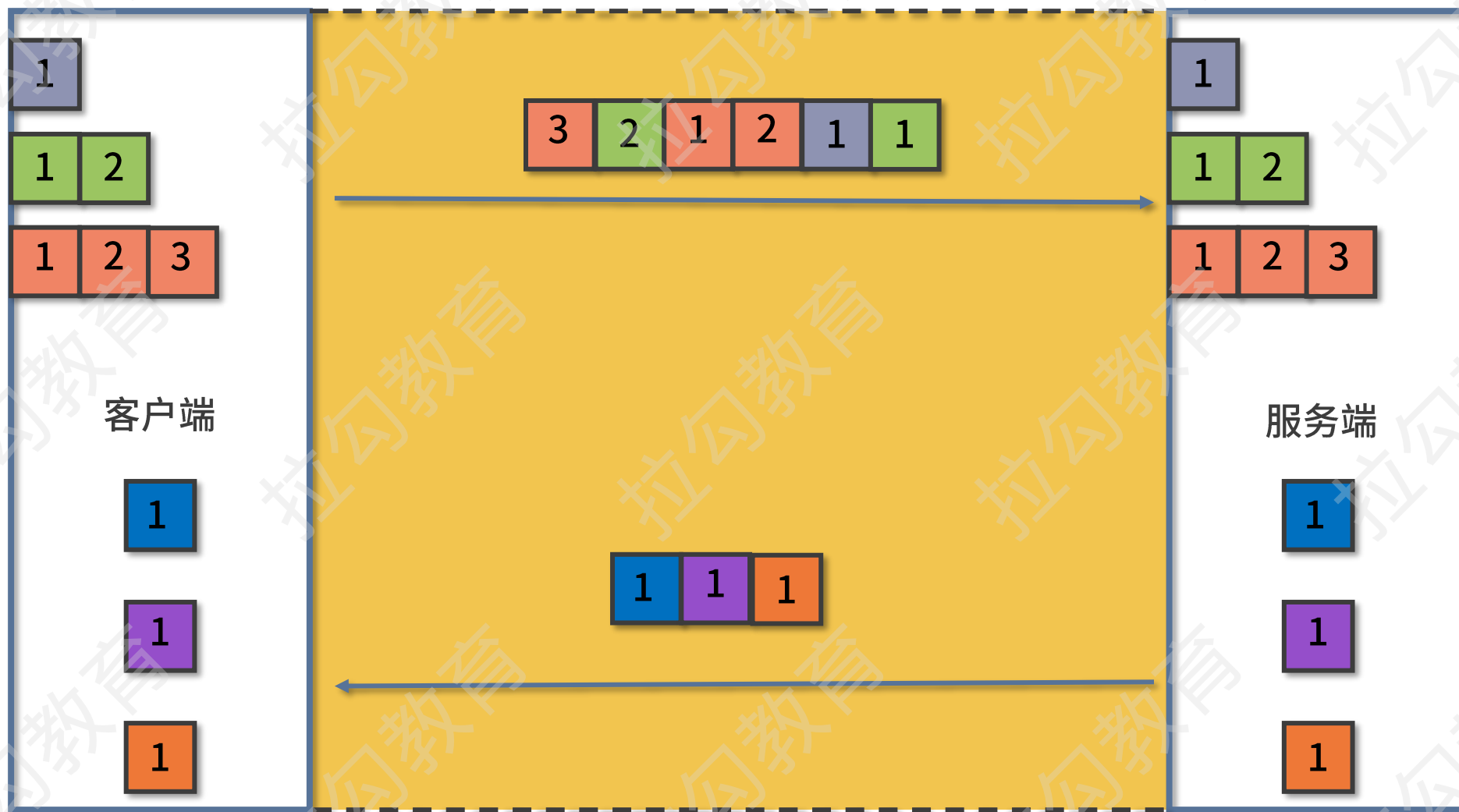
但由于它的并发能力受到限制，所以传输性能还有很大提升空间






- 浏览器为了减轻服务器的压力，限制了同一个域下的 **HTTP 连接数** 即 6 ~ 8 个，所以在 HTTP/1.1 下很容易看到资源文件等待加载的情况
对应优化的方式就是使用多个域名来加载图片资源
- HTTP/1.1 本身的问题
虽然 HTTP/1.1 中使用持久连接时，多个请求能共用一个 TCP 连接
但在一个连接中同一时刻只能处理一个请求
在当前的请求没有结束之前，其他的请求只能处于阻塞状态
这种情况被称为 “**队头阻塞**”



在 2015 年正式发布的 HTTP/2 中新增了一个二进制分帧的机制来提升传输效率



Name	Method	Status	Protocol
 kaiwu.lagou.com	GET	200	h2
 common.4f161b26.css	GET	200	h2
 vendor.424fdf064e46228ef60d.dll.js	GET	200	h2

延申2: HTTPS 原理

拉勾教育

— 互联网人实战大学 —

HTTP 虽然能满足客户端与服务端的通信需求

但这种使用明文发送数据的方式存在一定的安全隐患

因为通信内容很容易被通信链路中的第三方截获甚至篡改



对称加密

拉勾教育

— 互联网人实战大学 —

对称加密在加/解密过程中使用同一个密钥

而非对称加密使用不同的密钥进行加/解密

在**性能**方面，对称密钥更胜一筹，所以可以使用对称密钥



对称加密

拉勾教育

— 互联网人实战大学 —

但是肯定不能在每次通信中都使用同一个对称密钥

因为如果使用同一个密钥

任何人只要与服务端建立通信就能获得这个密钥

也就可以轻松解密其他通信数据了

所以应该是每次通信都要**随机生成**



非对称加密

拉勾教育

— 互联网人实战大学 —

由于不可能保证客户端和服务端同时生成一个相同的随机密钥
所以生成的随机密钥需要被传输

这样的话在传输过程中也会存在**被盜取**的风险

除了前面提到的对称加密，我们只有**非对称加密**这个选项了
比如客户端通过公钥来加密，服务端利用私钥来解密



密钥对生成后，该怎么分发呢？

如果在客户端生成密钥对，把私钥发给服务端

那么服务端需要为每个客户端保存一个密钥，这显然是不太现实的

所以只能由服务端生成密钥对

将公钥分发给需要建立连接的客户端

直接发送给客户端还是会被篡改

此时只能借助第三方来实现了，比如**证书机制**



具体来说就是把公钥放入一个**证书**中，该证书包含服务端的信息

比如颁发者、域名、有效期

为了保证证书是可信的

需要由一个可信的第三方来对证书进行签名

这个第三方一般是证书的颁发机构

也称 **CA (Certification Authority, 认证中心)**

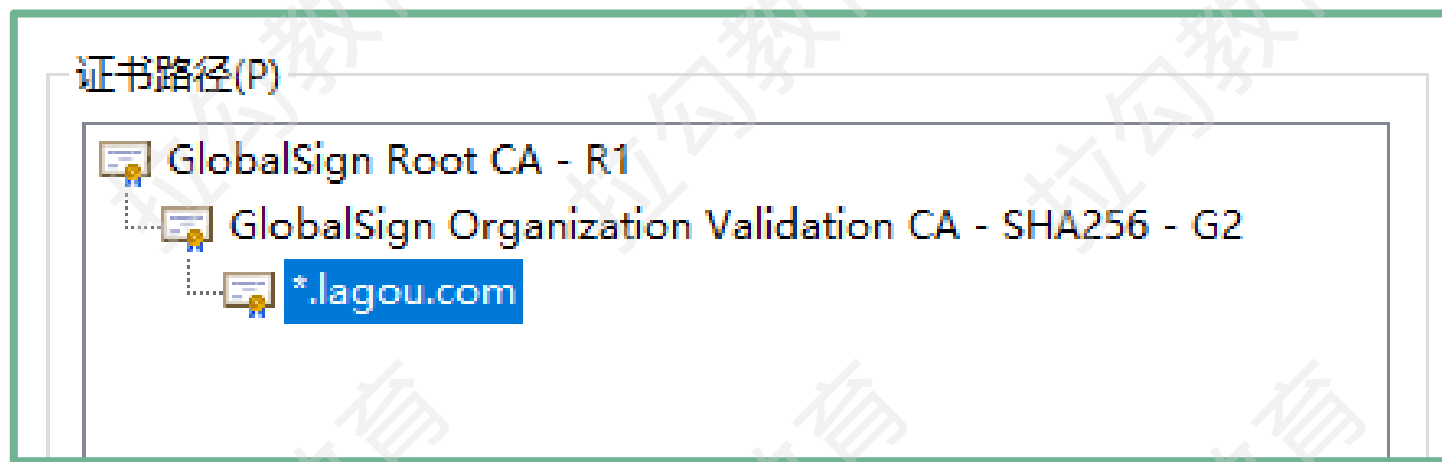


证书签名就是将证书信息进行 MD5 计算，获取唯一的**哈希值**
然后再利用证书颁发方的私钥对其进行加密生成

校验过程与之相反，需要用到证书颁发方的公钥对签名进行解密
然后计算证书信息的 MD5 值
将解密后的 MD5 值与计算所得的 MD5 值进行比对
如果**两者一致**代表签名是可信的



通过签名来颁发与校验证书的方式会形成一个可追溯的链，即**证书链**
处于证书链顶端的证书称为**根证书**，这些根证书被预置在操作系统的内部



HTTP/3

拉勾教育

— 互联网人实战大学 —

HTTP/2 也并非完美

如果客户端或服务端在通信时出现数据包丢失

或者任何一方的**网络出现中断**，那么整个 TCP 连接就会暂停



HTTP/3

拉勾教育

— 互联网人实战大学 —

HTTP/2 由于采用二进制分帧进行多路复用

通常只使用**一个 TCP 连接**进行传输

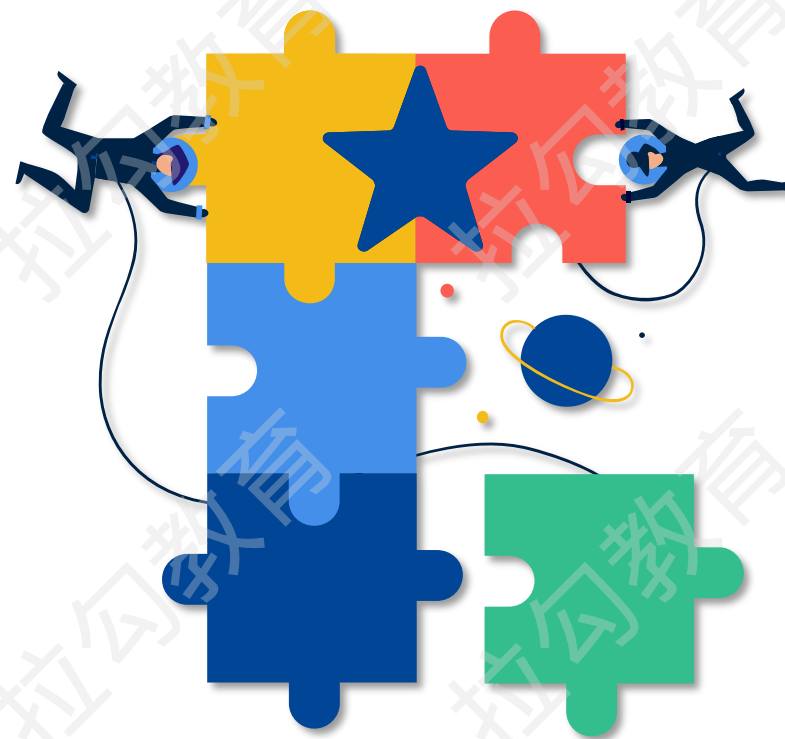
在丢包或网络中断的情况下后面的所有数据都被阻塞

但对于 HTTP/1.1 来说，可以开启**多个 TCP 连接**

任何一个 TCP 出现问题都不会影响其他 TCP 连接

剩余的 TCP 连接还可以正常传输数据

这种情况下 HTTP/2 的表现就不如 HTTP/1 了






2018 年 HTTP/3 将底层依赖的 TCP 改成 UDP

从而彻底解决了这个问题

UDP 相对于 TCP 而言最大的**特点**是传输数据时不需要建立连接

可以同时发送多个数据包

缺点就是没有确认机制来保证对方一定能收到数据

Name	Method	Status	Protocol	Type
 www.google.com	GET	200	h2	document
 ADGmq9RmtJkSJ82yxA4uVjieiQK8lVYzepVm...	GET	200	h2	png
 i1_1967ca6a.png	GET	200	h3-Q050	png

总结了 HTTP 各个版本的**核心改进以及解决的问题**

同时深入 HTTP 底层依赖的 **TCP**

讲解了 TCP 建立和断开连接的过程

分析了 HTTPS 如何通过证书机制以及加密方式来保障通信数据的安全



协议版本	解决的核心问题	解决方式
0.9	HTML 文件传输	确立了客户端请求、服务端响应的通信流程
1.0	不同类型文件传输	设立头部字段
1.1	创建/断开 TCP 连接开销大	建立长连接进行复用
2	并发数有限	二进制分帧
3	TCP 丢包阻塞	采用 UDP 协议

如果服务端要主动将数据推送到客户端

你知道有哪些解决方案吗?



Next: 第15讲: 《如何让浏览器更快地加载网络资源? 》

拉勾教育

— 互联网人实战大学 —



下载「拉勾教育App」
获取更多内容