24 React 常考进阶知识点 HTTP/2 很好的解决了当下最常用的 HTTP/1 所存在的一些性能问题,只需要升级到该协议就可 学习时长: 5分25秒 以减少很多之前需要做的性能优化工作,当然兼容问题以及如何优雅降级应该是国内还不普遍使 用的原因之一。 25 监控 已学完 学习时长: 2分48秒 虽然 HTTP/2 已经解决了很多问题,但是并不代表它已经是完美的了,HTTP/3 就是为了解决 26 UDP HTTP/2 所存在的一些问题而被推出来的。 已学完 学习时长: 3分11秒 HTTP/2 27 TCP 已学完 学习时长: 10分1秒 HTTP/2 相比于 HTTP/1, 可以说是大幅度提高了网页的性能。 28 HTTP 及 TLS 已学完 学习时长: 10分38秒 在 HTTP/1 中,为了性能考虑,我们会引入雪碧图、将小图内联、使用多个域名等等的方式。这 一切都是因为浏览器限制了同一个域名下的请求数量(Chrome 下一般是限制六个连接),当页 29 HTTP/2 及 HTTP/3 面中需要请求很多资源的时候,队头阻塞(Head of line blocking)会导致在达到最大请求数量 已学完 学习时长: 5分18秒 时,剩余的资源需要等待其他资源请求完成后才能发起请求。 30 输入 URL 到页面渲染的整个流程 在 HTTP/2 中引入了多路复用的技术,这个技术可以只通过一个 TCP 连接就可以传输所有的请 已学完 学习时长: 2分48秒 求数据。多路复用很好的解决了浏览器限制同一个域名下的请求数量的问题,同时也间接更容易 实现全速传输,毕竟新开一个 TCP 连接都需要慢慢提升传输速度。 大家可以通过 该链接 感受下 HTTP/2 比 HTTP/1 到底快了多少。 HTTP/1.1 HTTP/2 Latency: 179ms Latency: 145ms Load time: 11.96s Load time: 2.83s 在 HTTP/1 中,因为队头阻塞的原因,你会发现发送请求是长这样的 ■ tile... 20 pr ... 1. 2. ■ tile... 20 pr ... 3. 2.4 ■ tile... 20 pr ... 2. 2.4 ■ tile... 20 pr ... 2. 3.0 ■ tile... 20 pr ... 2. 2.5 ■ tile... 20 pr ... 1. 2. ■ tile... 20 pr ... 3. 2.5 ■ tile... 20 pr ... 1. 3.0 ■ tile... 20 pr ... 4. 3.0 ■ tile... 20 pr ... 3. 3. ■ tile... 20 pn ... 2. 3. tile... 20 pr ... 3. 3. ■ tile... 20 pr ... 1. 3.: ■ tile... 20 pr ... 1. 3.1 ■ tile... 20 pr ... 2. 3.: m tile... 20 pr ... 4. 3.: ■ tile... 20 pr ... 1. 3.: ■ tile... 20 pr ... 1. 3.4 ■ tile... 20 pr ... 3. 3.4 i tile... 20 pr ... 4. 3.√ ■ tile... 20 pr ... 3. 3. ■ tile... 20 pr ... 1. 3.(stile... 20 pr ... 3. 3.(■ tile... 20 pr ... 1. 3.0 ■ tile... 20 pr ... 1. 3.0 ■ tile... 20 pr ... 1. 3.0 ■ tile... 20 pr ... 2. 3. ■ tile... 20 pr ... 2. 3. ■ tile... 20 pr ... 1. 3.4 ◎時ず掘金技术社区 ■ tile... 20 pr ... 1. 3.4 在 HTTP/2 中,因为可以复用同一个 TCP 连接,你会发现发送请求是长这样的 ■ tile... 20 pr ... 2. 1.5 ■ tile... 20 pr ... 1.1.1 ■ tile... 20 pr ... 1.1.1 ■ tile... 20 pr ... 1. 1.1 ■ tile... 20 pr ... 1. 2.0 ■ tile... 20 pr ... 1, 2.0 ■ tile... 20 pr ... 1. 2.4 ■ tile... 20 pr ... 1. 2.0 pa... 20 sc ... (fr 1.2 二进制传输 HTTP/2 中所有加强性能的核心点在于此。在之前的 HTTP 版本中,我们是通过文本的方式传输 数据。在 HTTP/2 中引入了新的编码机制,所有传输的数据都会被分割,并采用二进制格式编 码。 Application (HTTP 2.0) HTTP 1.1 POST /upload HTTP/1.1 **Binary Framing** Host: www.example.org Content-Type: application/json Content-Length: 15 Session (TLS) (optional) {"msg":"hello"} Transport (TCP) HTTP 2.0 **HEADERS** frame Network (IP) **DATA** frame ○統十組会技术計 多路复用 在 HTTP/2 中,有两个非常重要的概念,分别是帧(frame)和流(stream)。 帧代表着最小的数据单位,每个帧会标识出该帧属于哪个流,流也就是多个帧组成的数据流。 多路复用,就是在一个 TCP 连接中可以存在多条流。换句话说,也就是可以发送多个请求,对 端可以通过帧中的标识知道属于哪个请求。通过这个技术,可以避免 HTTP 旧版本中的队头阻 塞问题,极大的提高传输性能。 HTTP 2.0 Connect Stream2 Stream3 Stream3 Stream1 Header 压缩 在 HTTP/1 中,我们使用文本的形式传输 header,在 header 携带 cookie 的情况下,可能每次 都需要重复传输几百到几千的字节。 在 HTTP /2 中,使用了 HPACK 压缩格式对传输的 header 进行编码,减少了 header 的大小。 并在两端维护了索引表,用于记录出现过的 header ,后面在传输过程中就可以传输已经记录过 的 header 的键名,对端收到数据后就可以通过键名找到对应的值。 服务端 Push 在 HTTP/2 中,服务端可以在客户端某个请求后,主动推送其他资源。 可以想象以下情况,某些资源客户端是一定会请求的,这时就可以采取服务端 push 的技术,提 前给客户端推送必要的资源,这样就可以相对减少一点延迟时间。当然在浏览器兼容的情况下你 也可以使用 prefetch。 HTTP/3 虽然 HTTP/2 解决了很多之前旧版本的问题,但是它还是存在一个巨大的问题,虽然这个问题并 不是它本身造成的,而是底层支撑的 TCP 协议的问题。 因为 HTTP/2 使用了多路复用,一般来说同一域名下只需要使用一个 TCP 连接。当这个连接中 出现了丢包的情况,那就会导致 HTTP/2 的表现情况反倒不如 HTTP/1 了。 因为在出现丢包的情况下,整个 TCP 都要开始等待重传,也就导致了后面的所有数据都被阻塞 了。但是对于 HTTP/1 来说,可以开启多个 TCP 连接,出现这种情况反到只会影响其中一个连 接,剩余的 TCP 连接还可以正常传输数据。 那么可能就会有人考虑到去修改 TCP 协议,其实这已经是一件不可能完成的任务了。因为 TCP 存在的时间实在太长,已经充斥在各种设备中,并且这个协议是由操作系统实现的,更新起来不 大现实。 基于这个原因,Google 就更起炉灶搞了一个基于 UDP 协议的 QUIC 协议,并且使用在了 HTTP/3 上,当然 HTTP/3 之前名为 HTTP-over-QUIC,从这个名字中我们也可以发现, HTTP/3 最大的改造就是使用了 QUIC,接下来我们就来学习关于这个协议的内容。 QUIC 之前我们学习过 UDP 协议的内容,知道这个协议虽然效率很高,但是并不是那么的可靠。QUIC 虽然基于 UDP,但是在原本的基础上新增了很多功能,比如多路复用、0-RTT、使用 TLS1.3 加 密、流量控制、有序交付、重传等等功能。这里我们就挑选几个重要的功能学习下这个协议的内 多路复用 虽然 HTTP/2 支持了多路复用,但是 TCP 协议终究是没有这个功能的。QUIC 原生就实现了这 个功能,并且传输的单个数据流可以保证有序交付且不会影响其他的数据流,这样的技术就解决 了之前 TCP 存在的问题。 并且 QUIC 在移动端的表现也会比 TCP 好。因为 TCP 是基于 IP 和端口去识别连接的,这种方 式在多变的移动端网络环境下是很脆弱的。但是 QUIC 是通过 ID 的方式去识别一个连接,不管 你网络环境如何变化,只要 ID 不变,就能迅速重连上。 0-RTT 通过使用类似 TCP 快速打开的技术,缓存当前会话的上下文,在下次恢复会话的时候,只需要 将之前的缓存传递给服务端验证通过就可以进行传输了。 纠错机制 假如说这次我要发送三个包,那么协议会算出这三个包的异或值并单独发出一个校验包,也就是 总共发出了四个包。 当出现其中的非校验包丢包的情况时,可以通过另外三个包计算出丢失的数据包的内容。 当然这种技术只能使用在丢失一个包的情况下,如果出现丢失多个包就不能使用纠错机制了,只 能使用重传的方式了。 小结 总结一下内容: • HTTP/2 通过多路复用、二进制流、Header 压缩等等技术,极大地提高了性能,但是还是存 在着问题的 • QUIC 基于 UDP 实现,是 HTTP/3 中的底层支撑协议,该协议基于 UDP,又取了 TCP 中的 精华, 实现了即快又可靠的协议 留言 输入评论 (Enter换行, # + Enter发送) 全部评论(10) 李乐勇 🍫 JY.1 1年前 TCP在只丢了一个包的情况下,为什么整个TCP都要重传?只重传丢失的那个包不就可以了吗? □ 点赞 □ 回复 程序员升职记 → → → 前端研发工程师 @ 帝都... 3年前 多路复用到底是怎么在工作中使用的 心点赞 🗇 1 (B) 用_户_名 2年前 。。。其实就是把网站的协议换成http2就行了,服务端或者运维干的事 心 点赞 🖵 回复 ⟨/⟩ Kaim ❤️жа FE 4年前 是服务器端决定用HTTP/1还是HTTP/2的么? 心点赞 🖵 2 **肤浅嘻** 3年前 我也想问这个问题 心 点赞 🖵 回复 金钟罩铁布衫 3年前 静态服务器端可以配置,比如nginx 心 点赞 🖵 回复 6月冰西瓜
※3/33
※3/4年前 "同时也接更容易实现全速传输"这句话感觉不太通顺。 心1 回复 易水寒YSP ❖ ͿΥ.1 4年前 受益匪浅 心 点赞 🖵 回复 rudy 🚧 💝 JY.3 4年前 心 点赞 🖵 回复 zy_ch 🚧 💝 🗸 🥒 外卖员 @ 美团 🔰 4年前 真好 心 点赞 🖵 回复

这一章节我们将来学习 HTTP/2 及 HTTP/3 的内容。 HTTP/2 很好的解决了当下最常用的 HTTP/1 所存在

HTTP/2及HTTP/3

❖ 稀土掘金 课程

22 Vue 常考进阶知识点

23 React 常考基础知识点

学习时长: 17分7秒

已学完 学习时长: 13分28秒

已学完 学习时长: 26分37秒

三 前端面试之道