Atitit http 1 2.0 3.0 的区别与不同

目录

[2. ，这组RFC又叫“HTTP/2 预览版” 3](#_Toc15034)

[2.1. TLS协议。 3](#_Toc4669)

[2.2. 传统的短链接。 4](#_Toc5247)

[2.3. 可重用TCP的的长链接模型。 4](#_Toc20422)

[2.4. 服务端push的模型。 4](#_Toc640)

[2.5. WebSocket模型。 4](#_Toc31892)

[3. HTTP/2基本上解决了之前的这些性能问题， 4](#_Toc22466)

[4. 于是HTTP/3破天荒地把HTTP底层的TCP协议改成了UDP！ 5](#_Toc10604)

HTTP (Hypertext transfer protocol) 翻译成中文是超文本传输协议，是互联网上重要的一个协议，由欧洲核子研究委员会CERN的英国工程师 [Tim Berners-Lee](https://en.wikipedia.org/wiki/Tim_Berners-Lee" \o ") v发明的，同时，他也是WWW的发明人，最初的主要是用于传递通过HTML封装过的数据。在1991年发布了HTTP 0.9版，在1996年发布1.0版，1997年是1.1版，1.1版也是到今天为止传输最广泛的版本（初始[RFC 2068](https://tools.ietf.org/html/rfc2068) 在1997年发布， 然后在1999年被 [RFC 2616](https://tools.ietf.org/html/rfc2616) 取代，再在2014年被 [RFC 7230](https://tools.ietf.org/html/rfc7230) /[7231](https://tools.ietf.org/html/rfc7231)/[7232](https://tools.ietf.org/html/rfc7232)/[7233](https://tools.ietf.org/html/rfc7233)/[7234](https://tools.ietf.org/html/rfc7234)/[7235](https://tools.ietf.org/html/rfc7235)取代），2015年发布了2.0版，其极大的优化了HTTP/1.1的性能和安全性，而2018年发布的3.0版，继续优化HTTP/2，激进地使用UDP取代TCP协议，目前，HTTP/3 在2019年9月26日 被 Chrome，Firefox，和Cloudflare支持，所以我想写下这篇文章，简单地说一下HTTP的前世今生，让大家学到一些知识，并希望可以在推动一下HTTP标准协议的发展。

#### **HTTP/1.1**

HTTP/1.1 主要解决了HTTP 1.0的网络性能的问题，以及增加了一些新的东西：

* 可以设置 keepalive 来让HTTP重用TCP链接，重用TCP链接可以省了每次请求都要在广域网上进行的TCP的三次握手的巨大开销。这是所谓的“****HTTP 长链接****” 或是 “****请求响应式的HTTP 持久链接****”。英文叫 HTTP Persistent connection.
* 然后支持pipeline网络传输，只要第一个请求发出去了，不必等其回来，就可以发第二个请求出去，可以减少整体的响应时间。（注：非幂等的POST 方法或是有依赖的请求是不能被pipeline化的）
* 支持 Chunked Responses ，也就是说，在Response的时候，不必说明 Content-Length 这样，客户端就不能断连接，直到收到服务端的EOF标识。这种技术又叫 “****服务端Push模型****”，或是 “****服务端Push式的HTTP 持久链接****”
* 还增加了 cache control 机制。
* 协议头注增加了 Language, Encoding, Type 等等头，让客户端可以跟服务器端进行更多的协商。
* 还正式加入了一个很重要的头—— [HOST](https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Host" \t "https://coolshell.cn/articles/_blank)这样的话，服务器就知道你要请求哪个网站了。因为可以有多个域名解析到同一个IP上，要区分用户是请求的哪个域名，就需要在HTTP的协议中加入域名的信息，而不是被DNS转换过的IP信息。
* 正式加入了 OPTIONS 方法，其主要用于 [CORS – Cross Origin Resource Sharing](https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS" \t "https://coolshell.cn/articles/_blank) 应用。

# ，这组RFC又叫“HTTP/2 预览版”

。其中影响HTTP发展的是两个大的需求：

一个需要是加大了HTTP的安全性，这样就可以让HTTP应用得广泛，比如，使用

## TLS协议。

另一个是让HTTP可以支持更多的应用，在HTTP/1.1 下，HTTP已经支持四种网络协议：

## 传统的短链接。

## 可重用TCP的的长链接模型。

## 服务端push的模型。

## WebSocket模型。

自从2005年以来，整个世界的应用API越来多，这些都造就了整个世界在推动HTTP的前进，我们可以看到，****自2014的HTTP/1.1 以来，这个世界基本的应用协议的标准基本上都是向HTTP看齐了，也许2014年前，还有一些专用的RPC协议，但是2014年以后，HTTP协议的增强，让我们实在找不出什么理由不向标准靠拢，还要重新发明轮子了。****

# HTTP/2基本上解决了之前的这些性能问题，

其和HTTP/1.1最主要的不同是：

* HTTP/2是一个二进制协议，增加了数据传输的效率。
* HTTP/2是可以在一个TCP链接中并发请求多个HTTP请求，移除了HTTP/1.1中的串行请求。
* HTTP/2会压缩头，如果你同时发出多个请求，他们的头是一样的或是相似的，那么，协议会帮你消除重复的部分。这就是所谓的HPACK算法（参看[RFC 7541](https://tools.ietf.org/html/rfc7541" \t "https://coolshell.cn/articles/_blank) 附录A）
* HTTP/2允许服务端在客户端放cache，又叫服务端push，也就是说，你没有请求的东西，我服务端可以先送给你放在你的本地缓存中。比如，你请求X，我服务端知道X依赖于Y，虽然你没有的请求Y，但我把把Y跟着X的请求一起返回客户端。

# ****于是HTTP/3破天荒地把HTTP底层的TCP协议改成了UDP！****

目前看下来，HTTP/3目前看上去没有太多的协议业务逻辑上的东西，更多是HTTP/2 + QUIC协议。但，HTTP/3 因为动到了底层协议，所以，在普及方面上可能会比 HTTP/2要慢的多的多。但是，可以看到QUIC协议的强大，细思及恐，QUIC这个协议真对TCP是个威胁，如果QUIC成熟了，TCP是不是会有可能成为历史呢？

未来十年，让我们看看UDP是否能够逆袭TCP……

HTTP的前世今生 | | 酷 壳 - CoolShell