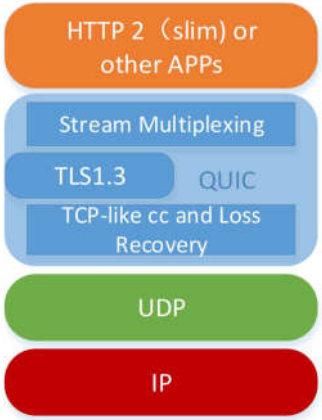
QUIC

Quick UDP Internet Connections

中国科学技术大学
自动化系　郑烃



提纲

1. QUIC概述
2. TCP及UDP之上的应用开发
3. HTTP协议及QUIC的历史
4. 在协议栈中的位置和主要特性
5. QUIC主要工作原理
6. QUIC应用及效果简介

提纲

1. QUIC概述
2. TCP及UDP之上的应用开发
3. HTTP协议及QUIC的历史
4. 在协议栈中的位置和主要特性
5. QUIC主要工作原理
6. QUIC应用及效果简介

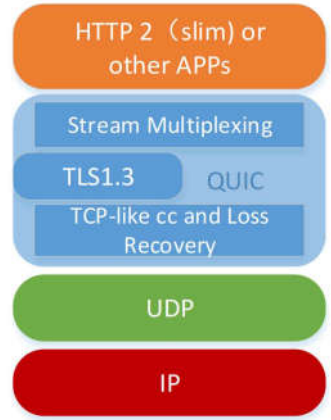
1. QUIC简介–来自HTTP的简史

HTTP/2在应用层实现多个高性能特性：
多流复用解决HOL阻塞，头部压缩和服务器推送等

为了破除在TLS/TCP上实现HTTP/2的固有限制
将在HTTP/2的功能整体搬移到UDP之上

补上TCP可靠传输，拥塞控制等特性，但重新采用高效方式实现，与TLS1.3协作实现安全性

凝练成用户态的传输协议QUIC，为HTTP提供服务，也可以为其他应用提供服务



1. QUIC简介-主要特性

提高HTTP的传输效率：握手延迟低、HOL堵塞避免、高效可靠恢复、模块化CC，连接迁移等

安全性：绝大部分头部和所有数据被加密，所有信息都是被认证的，每次会话新秘钥等，安全性得以提升

用户态，便于快速部署

避免网络middlebox干预

提纲

1. QUIC概述
2. TCP及UDP之上的应用开发
3. HTTP协议及QUIC的历史
4. 在协议栈中的位置和主要特性
5. QUIC主要工作原理
6. QUIC应用及效果简介

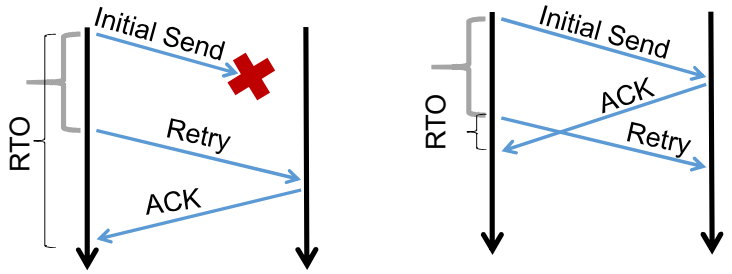
2. TCP及UDP之上的应用开发

传统架构上的应用开发：TCP和UDP之上二选一

TCP之上的应用及其优缺点：

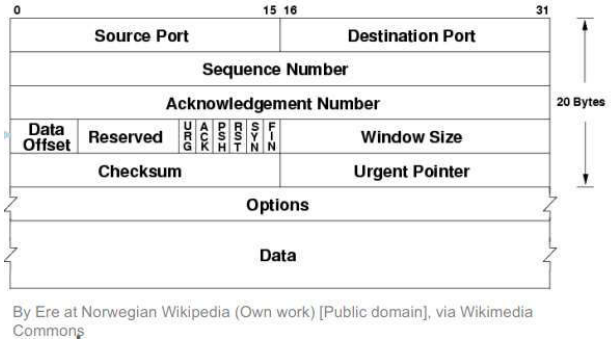
- TCP特性：应用区分、可靠、拥塞控制、流量控制等
- 好处：应用享受TCP之上开发的便利性，开发效率高
- 代价：忍受TCP本身问题：1）握手交互次数多、慢启动、基于丢失的拥塞控制吞吐抖动，TCP实现可靠性的恢复时间长；2）应用有性能天花板
- 例子：HTTP2.0在TLS/TCP上做了大量的工作（并行传输），无法突破TCP的固有问题和实现代价
- 为了实现安全：TLS/TCP，但是握手延迟更大、效率更低

2. TCP及UDP之上的应用开发-TCP主要问题



TCP协议主要问题之一：TCP 序号二义性
RTO计算有问题，丢失重传效率低

2. TCP及UDP之上的应用开发-TCP主要问题



By Ere at Norwegian Wikipedia (Own work) [Public domain], via Wikimedia Commons

TCP协议主要问题之一：**TCP头部长度（最长60B，选项最长40B）**
新特性如SACK、FAST OPEN、MPTCP，时戳等不够承载

2. TCP及UDP之上的应用开发-TCP主要问题

- 僵化：OS内核态方式+广泛部署的中间设备处理TCP协议方式固化
  - OS内核态实现，修改要求高，难度大，并且需要广泛部署才能够体现新特性，如：MPTCP
  - 中间设备对于TCP协议报文段的处理固化，如：一些有ECN选项的TCP段被防火墙过滤掉

2. TCP及UDP之上的应用开发

- UDP之上的应用及其优缺点：
  - UDP特性：区分进程，不可靠，无流量控制，无CC，无安全性等
  - 优点：UDP轻载，效率高，应用可以控制往网络中注入的速率，没有受制于TCP的性能天花板
  - 缺点：简陋、不安全，需要应用自己实现很多特性，开发工作量大，包括比较繁琐的可靠性恢复工作

2. TCP及UDP之上的应用开发

- 随着应用的需求越来越高，不管是在TCP还是UDP本身都无法完全满足应用的高要求：高效、可靠并且安全
- 同时，在OS中内核态升级TCP或者UDP本生存在着巨大的困难
  - OS内核态功能，开发难度大，要求高
  - 几十年的历史，比较成熟，全面部署难度大
  - 网络设备处理TCP或者UDP的IP分组的模式已固化，僵化，例如：ECN
- 大多数应用都是在用户态增强TCP或者UDP的服务，满足其效率提升和其他个性化特性的需求
  - QUIC：1）HTTP/2特性迁移到UDP之上；2）TCP特性的高效实现；3）与TLS1.3协作实现安全；4）迁移到UDP上后方便增加的一些新特性：连接迁移

提纲

1. QUIC概述
2. TCP及UDP之上的应用开发
3. HTTP协议及QUIC的历史
4. 在协议栈中的位置和主要特性
5. QUIC主要工作原理
6. QUIC应用及效果简介

3. HTTP协议及QUIC的历史

- HTTP 1.0（1996年），在0.9版本上增加了POST, DELETE, PUT, HEAD等方法，有报文头和报文体区分
  - 非持久连接：在TCP连接上只有一个对象的请求和响应，之后关闭，效率低下
- HTTP1.1（1999年），支持持久HTTP连接，在一个TCP连接上可以发送多个对象的请求和响应
  - 其他新特性：管道化、支持缓存、断点续传（对象偏移量）
  - 问题：有头端堵塞HTTP HOL问题

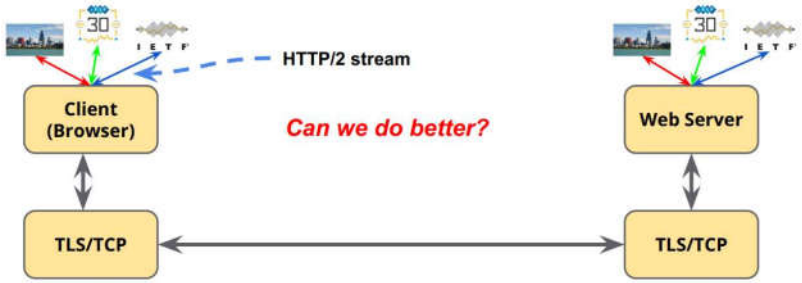
HTTP 1.1 单个TCP连接+流水线

HTTP1.1，单个TCP连接+流水线
HTTP HOL堵塞

HTTP1.1 并发TCP和流水线

HTTP 1.1，多个TCP连接+流水线
问题：HTTP HOL堵塞，服务器负担重（服务器一般支持来自一个客户端的6个连接）

HTTP/2上的多路复用



HTTP/2（2015年），1个TCP连接上多stream传，解决HTTP HOL堵塞问题
问题：TCP头端堵塞问题仍然存在，握手延迟大3RTT(TLS 1.2)

3.HTTP协议及QUIC的历史

- HTTP/2：2015年谷歌提出SPDY协议的标准化版本
- 主要特性：
  - 多路复用解复用：连接之上流，每个流相当于之前的请求/响应，C/S间连接数量不必开设很多，减轻服务器负担，解决了HTTP头部阻塞的问题
  - 二进制编码：不再编码成ASCII文本，直接二进制编码，省却了编码和解码时间
  - 头部压缩：采用HPACK技术，减少头部开销，同样的头部信息用编码表示
  - 服务器推送：将客户端可能要用的资源主动推送给客户端，避免客户端请求耽误时间
  - 支持重置、请求优先级
  - 安全性在TLS1.2之上

3.HTTP协议及QUIC的历史-HTTP/2的主要问题

- 握手交互次数多，延迟大（300ms左右）：TCP握手-1RTT，TLS-2RTT
- 存在TCP HOL堵塞问题：
  - 虽然可以有很多HTTP的流,最终落实在一个TCP连接上
  - 一旦某段出问题，需要重传，TCP接收方才能向上交付
  - 虽该段后面的一些流的数据是正确的，也无法交付
- 网络迁移问题，由于终端设备移动IP地址变化，或通过NAT重绑定设备的PORT变化，连接无法维护，HTTP应用无法正常使用
- TCP实现可靠恢复工作效率低等

3.HTTP协议及QUIC的历史- HTTP/2的问题分析

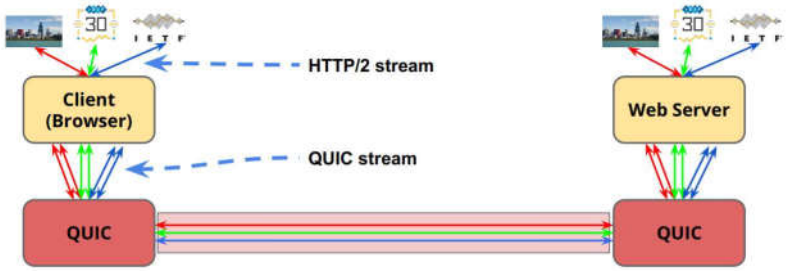
- 在TLS/TCP之上实现，忍受底层协议的实现代价和方式
  - TLS+TCP 握手时间，慢启动
  - 基于丢失的拥塞控制
  - TCP HOL堵塞，可靠恢复工作的低效率
- TCP协议本身的设计问题：序号二义性问题，RTT计算时间，头部选项有限支持新特性困难，
- 结论：无法靠HTTP/2本身来解决

3.HTTP协议及QUIC的历史

■HTTP/3的思路

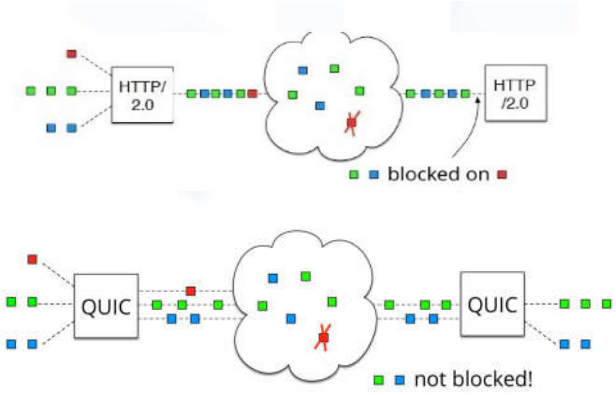
- ■为破除TLS/TCP之上实现HTTP/2的制约，整体搬迁到UDP之上
- ■将HTTP/2多路复用功能，二进制传输，头部压缩等好的工作积累下沉到一个单独的层次QUIC，而不是在HTTP上做，也可以为其他应用服务
- ■**将部分TCP功能上移到QUIC：**移到UDP之上实现，它不具备TCP所具备的可靠性、拥塞控制、流量控制工作，需要实现（而且需要高效实现）
- ■基于UDP，实现连接迁移等新特性
- ■和TLS1.3配合实现安全性，协作关系不是包含关系
  - ■使用TLS1.3的握手认证对方身份和交换秘钥等，握手1-RTT
  - ■TLS1.3使用QUIC传输报文
- ■QUIC在OS用户态实现，便于部署推广

QUIC基于UDP上的多路复用



HTTP/3 over QUIC，1个QUIC连接上多stream传
优点：解决HTTP HOL堵塞问题,不存在TCP HOL问题
握手延迟小0-RTT~1RTT(with TLS 1.3)

3.HTTP协议及QUIC的历史

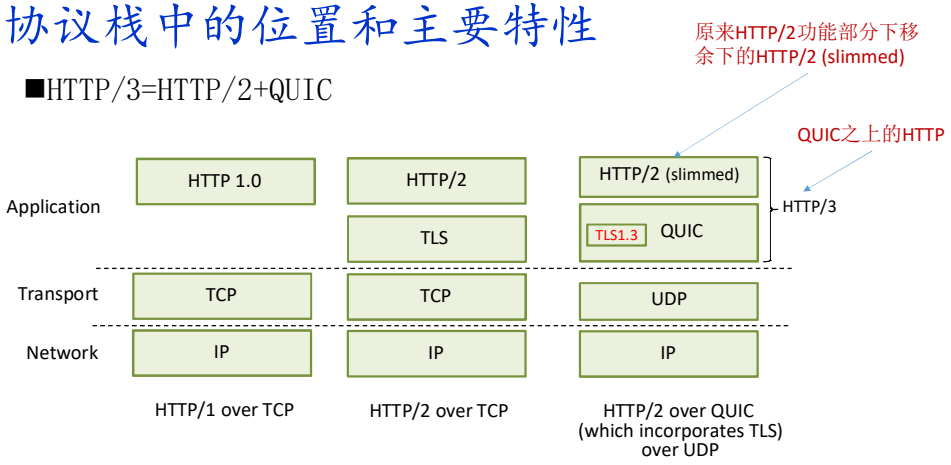


提纲

1. QUIC概述
2. TCP及UDP之上的应用开发
3. HTTP协议及QUIC的历史
4. 在协议栈中的位置和主要特性
5. QUIC主要工作原理
6. QUIC应用及效果简介

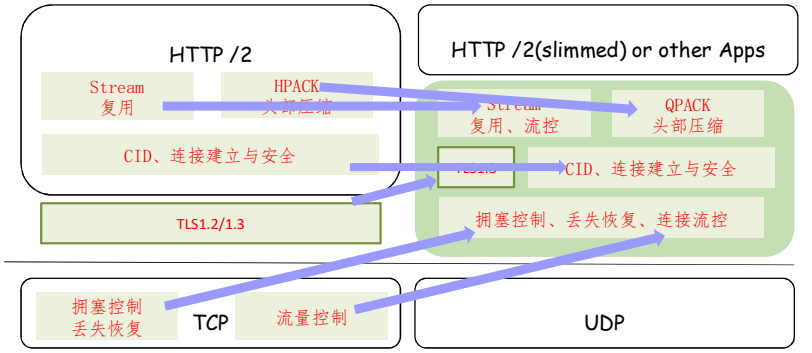
4. 协议栈中的位置和主要特性

■HTTP/3=HTTP/2+QUIC



4. 协议栈中的位置和主要特性

■HTTP/3=HTTP/2+QUIC



更快、更安全、可演化

4. 协议栈中的位置和主要特性

■继承自HTTP/2(并升级)
- ■一个连接上多流复用
  - ■over UDP消除头端堵塞
  - ■减轻服务器负担
  - ■支持各流优先级
- ■QPACK头部压缩;二进制分帧传输;FEC;服务器推送

■QUIC新特性
- ■0-RTT快速握手
- ■网络连接迁移

■继承自TCP（并升级）
- ■高效的差错恢复，无状态重试
- ■可插拔拥塞控制cc机制
- ■流量控制

■与TLS1.3协作实现安全
- ■QUIC握手与TLS1.3整合
- ■所有报文头可认证的
- ■报文载荷是加密的
- ■Dos保护

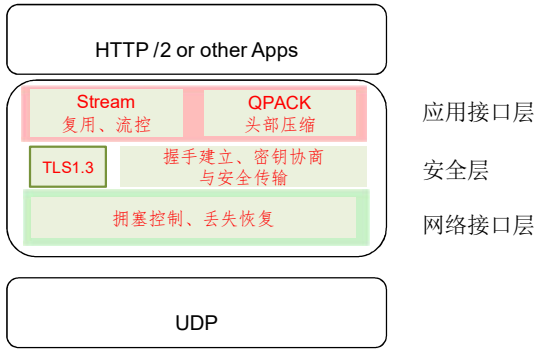
提纲

1. QUIC概述
2. TCP及UDP之上的应用开发
3. HTTP协议及QUIC的历史
4. 在协议栈中的位置和主要特性
5. QUIC主要工作原理
6. QUIC应用及效果简介

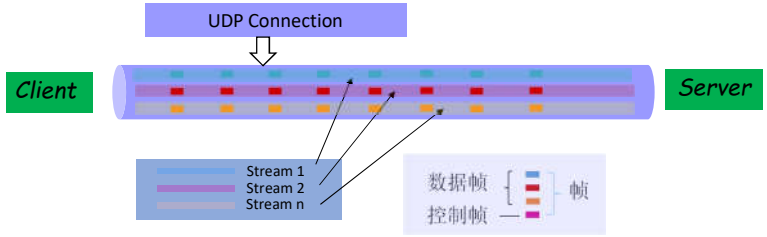
5.QUIC主要工作原理

- ■系统架构
- ■主要概念
- ■多路复用解复用
- ■快速握手
- ■连接迁移
- ■拥塞控制和流量控制
- ■丢失恢复的高效实现
- ■安全性
- ■头部压缩
- ■二进制分帧传输
- ■服务器推送

5.1 QUIC系统架构

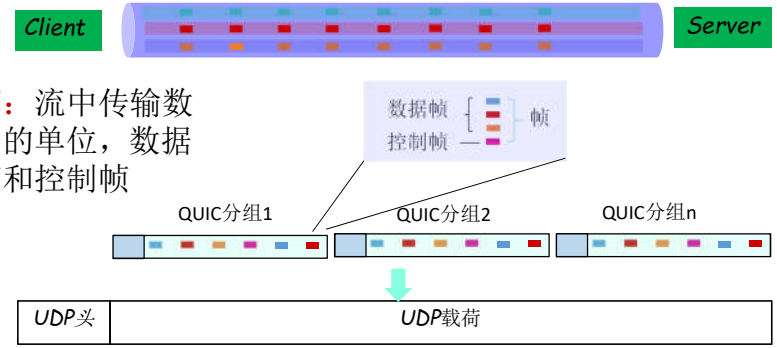


5.3 QUIC主要概念



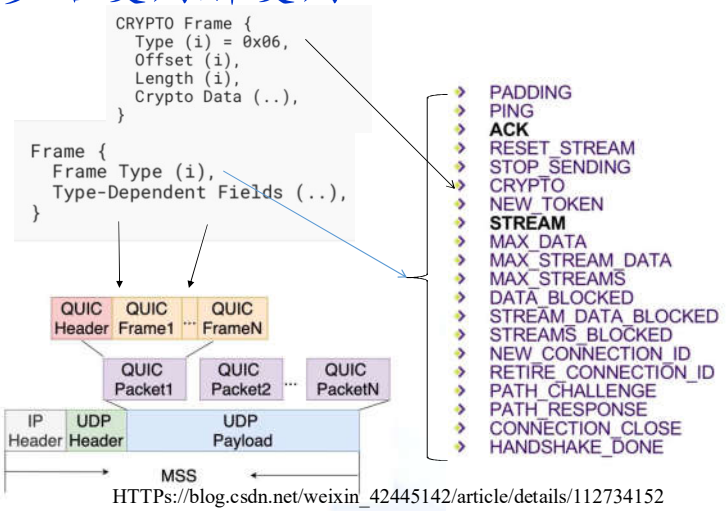
- 连接（connection）:C到S间2个端节点的一个会话
  - 每个端接点自己选择connection id
  - 用对方的connection id标识
- 流(stream)：一个完整的请求和响应字节流，轻量级、有序字节流的抽象
  - 连接上可以有多个流：单向流，双向流
  - 流由62位stream id标识，最低两位:1位标识单向双向，1位标识流的发起者
  - 如：一个连接上有多个对象的请求和响应，每个对象的请求/响应在一个流中

5.3 QUIC主要概念



- 帧：流中传输数据的单位，数据帧和控制帧
- 分组：连接传输的数据单位，UDP数据报载荷中有若干分组，分组包含了若干帧

5.3 多路复用解复用



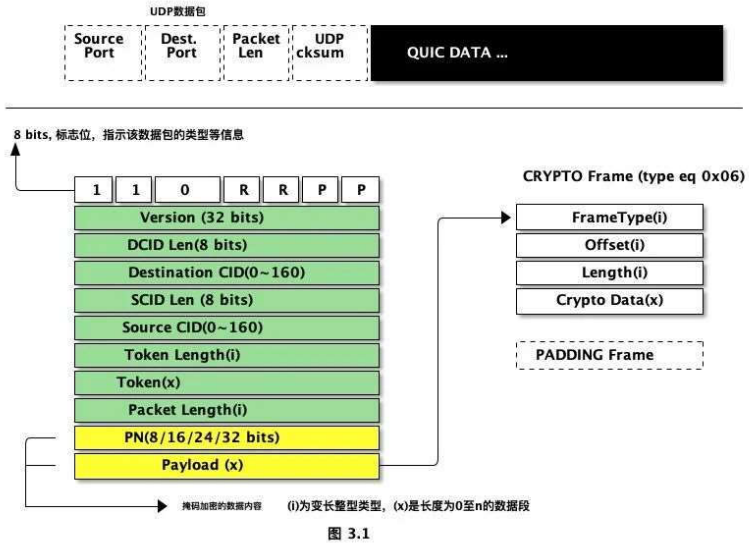
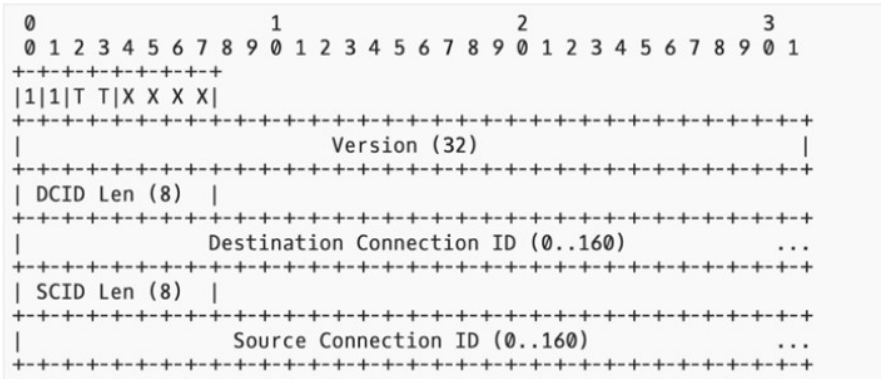


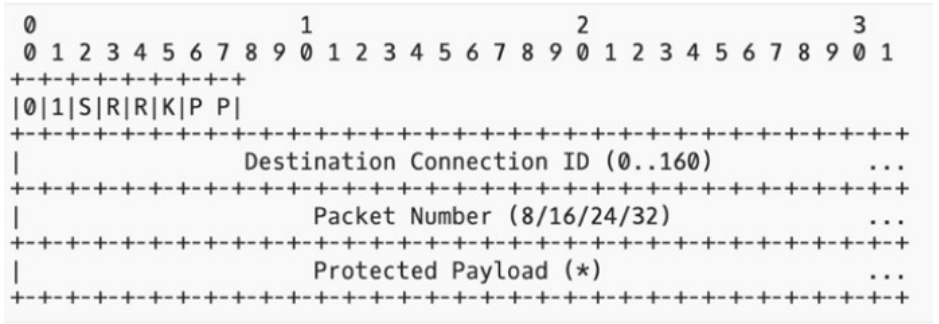
图 3.1

5.3 多路复用解复用



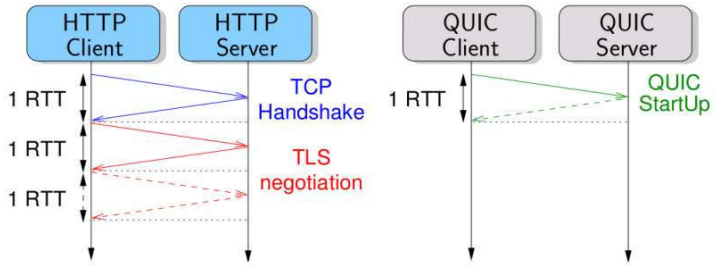
头部包括完整的DCID和SCID

5.3 多路复用解复用



头部经过压缩的，只有DCID，没有SCID

5.4 快速握手



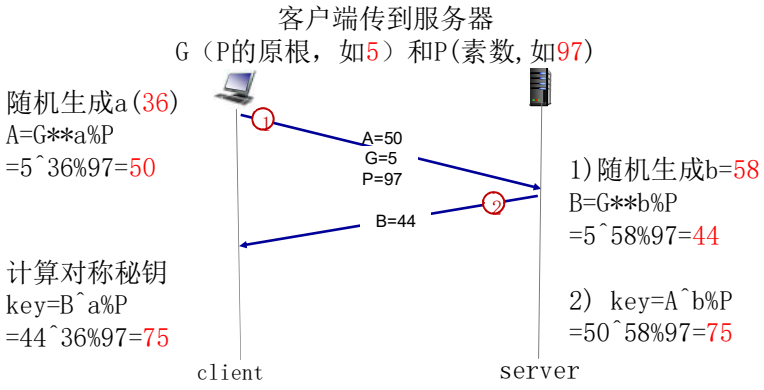
TCP+TLS1.2+HTTP/2　　QUIC+TLS1.3+HTTP/3

引用自：quicker_KevinPittevils_August2018

5.4 快速握手-TLS1.3

- TLS 1.3“加密套件cipher suite”：秘钥生成、加密、MAC、数字签名算法组合
- TLS 1.3可选择范围比TLS1.2 套件要少的多
  - 为防止降级攻击：只支持5种选择，而不是37种，
  - 通过Diffie-Hellman (DH)算法进行秘钥交换，不是RSA，交互次数少
  - 对数据实施加密和认证组合算法（“可认证加密”），而不是加密之后再认证的序列：4 based on AES
  - HMAC采用SHA (256 or 284)加密哈希函数

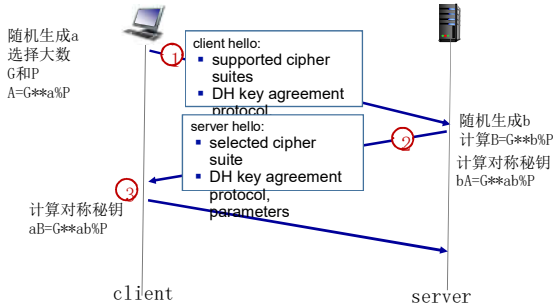
5.4 快速握手-Diffie-Hellman算法



握手和秘钥生成1RTT完成

5.4 快速握手

TLS 1.3 握手: 1 RTT



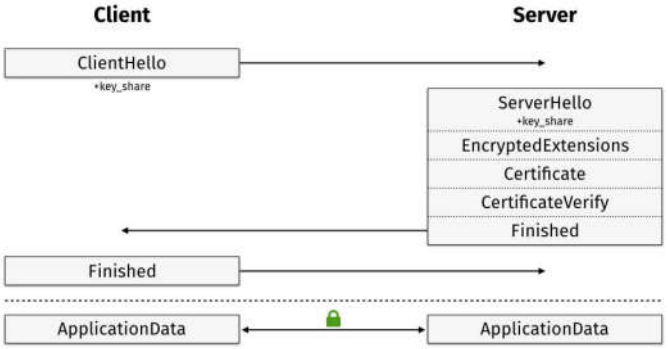
G是P的原根

① 客户段TLS hello 报文:
- 猜测秘钥交换协议，参数大数G、P和A=G**a%P
- 支持加密套件支持列表

② 服务器TLS hello 报文选择
- 秘钥协商协议，参数B=G**b%P
- 加密套件
- 签名了的证书

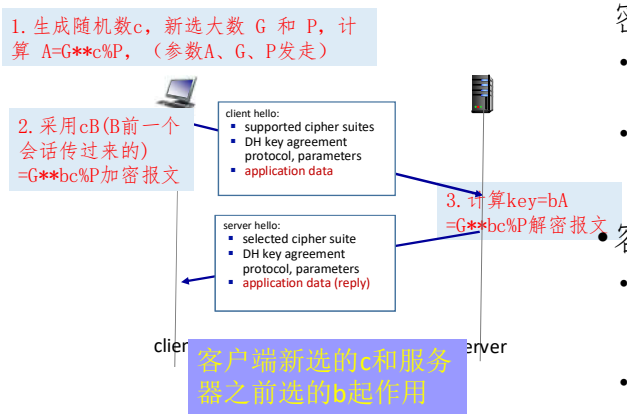
③ 客户端:
- 校验服务器的证书，认证对方身份
- 生成秘钥
- 可以发送应用报文了 (e.g., HTTPS GET)

5.4 快速握手-1RTT



1-RTT连接建立
（TLS1.3握手报文封装在QUIC的帧中传输）

5.4 快速握手（0RTT）

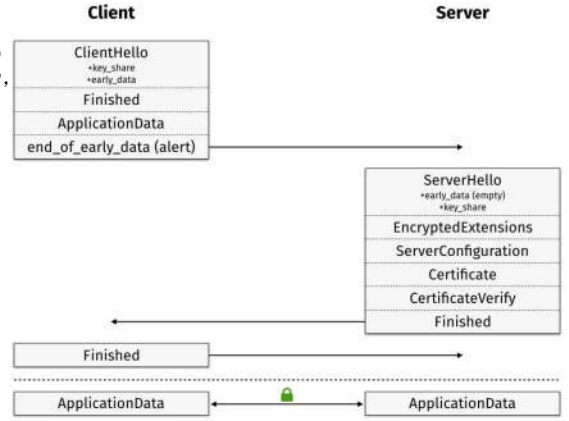


TLS 1.3 握手：0 RTT

- 初始化hello报文包含加密应用数据！
  - “恢复resuming”之前客户端和服务器之间的连接
  - 应用数据采用之前的连接所使用的“resumption master secret”进行加密
- 容易受到重放攻击
  - HTTP GET或者客户端请求不改变服务器的状态这种报文没问题的
  - 在QUIC协议中通过超时机制，几个小时过期

5.4 快速握手-0-RTT

0.客户端缓存ServerConfig（B=G**b%P）
1.随机生成c，算A=B**c%P，A，G，P 传走，tickets
2.key=c*B%P (G**bc*%P)加密报文



0.据ticket找b
1.算key=b*A%P=G**bc%P
2.G**bc%P解密报文

0-RTT连接恢复

5.5 连接迁移

■问题的出现：
- ■设备移动和NAT重新绑定，IP地址或者PORT变化
- ■TCP连接：socket与4元组捆绑（TCP,本地IP，本地端口，对方IP，对方端口）
- ■任何一端(对方)的IP地址变化或者PORT变化，TCP连接丢失

■HTTP/2基于TCP，因此设备IP地址或者端口号变化，通信终止

■QUIC连接迁移问题的解决
- ■基于UDP，OS内核不维护和对方通信的状态
- ■QUIC连接在用户态，用CID维护和对方通信状态
- ■CID对应（UDP，本地IP,本地UDP port,对方IP，对方UDP port）
- ■更新CID和本地IP地址、Port的映射关系
- ■只要对方IP地址和PORT不变，UDP能够将UDP数据报发给对方
- ■通信关系仍能维持

5.6 拥塞控制和流量控制

- ■拥塞控制
  - ■可插拔的模块化拥塞控制，支持演化
  - ■默认New Reno和Cubic算法
  - ■可以采用新的拥塞控制算法：如：BBR
- ■流量控制：两个级别的流量控制
  - ■连接级别的流量控制
  - ■流级别的流量控制

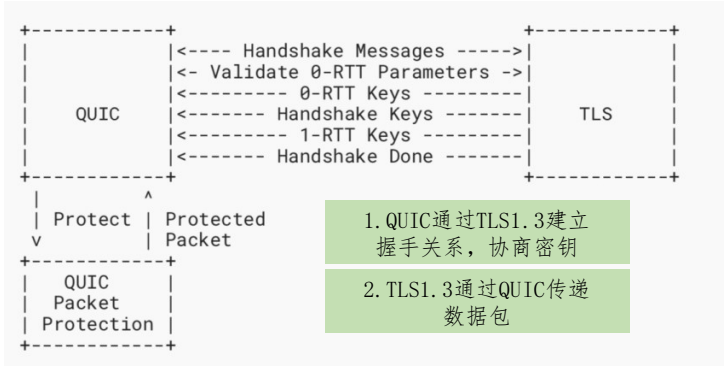
5.7 丢失恢复的高效实现

- ■ACK帧给出最多256个SACK（TCP最多3个）
  - ■精确给定了接收端哪些QUIC分组收到的情况
  - ■发送方有的放矢而不是盲目重传（浪费带宽，恢复时间短）
  - ■能避免99%以上的超时重传（时机太迟，可靠恢复时间长）
- ■PN单调递增，解决了序号二义性问题
  - ■规定PN单调递增，如递增1或者20、30
  - ■重传分组具备不同的PN，但是分组内可能是之前丢失的帧，具备相同帧号
  - ■RTO计算无二义性
- ■超时重传
  - ■ACK帧中给出处理时间
  - ■RTO计算扣除在接收方的处理时间
  - ■RTO计算更准

5.8 安全性

- ■与TLS1.3协作实现安全性
  - ■TLS1.3使用QUIC分组（帧）传送TLS报文
    - ■QUIC在CRYPTO帧中携带TLS握手数据，每个帧由偏移量和长度标识的连续握手数据块组成
    - ■帧被打包成QUIC数据包，并在当前的TLS加密级别下进行加密
  - ■QUIC通过TLS1.3 认证身份，实现秘钥共享
- ■QUIC的安全特性
  - ■所有的信息都是被认证的
  - ■大部分头部和所有的载荷都是加密的

5.8 安全性



QUIC与TLS1.3的协作关系

5.9 头部压缩

■经常使用的头部信息采用编码代替，而不是传输头部文本本身

■HTTP/2-HPACK

■QUIC-QPACK改进并适应

- ■采用动态表，静态表等编码方式
- ■大大减少头部信息传输的数量

5.10 二进制分帧传输

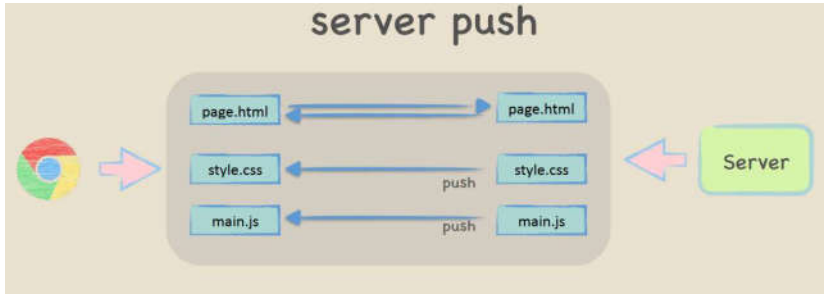
■采用二进制编码传输帧内容

■而不是ASCII本身

■减少了编码和解码的时间

■提升效率

5.11 服务器推送



- 当服务器收到资源请求做响应时，将客户端需要的资源主动推送给客户端
- Frame Type：PUSH_PROMISE，具备新建的Stream_ID
- 客户端接收

提纲

1. QUIC概述
2. TCP及UDP之上的应用开发
3. HTTP协议及QUIC的历史
4. 在协议栈中的位置和主要特性
5. QUIC主要工作原理
6. QUIC应用及效果简介

6.QUIC应用及效果简介

■2021年5月，历经9年经过30个草案变成了RFC 9000

■主流商家支持：国际（谷歌，苹果，微软，脸书，F5等），国内（阿里，腾讯，华为，新华三等）都已经落地、运行

■流量占比

- ■谷歌浏览器85%以上请求
- ■互联网7%以上

■性能提升

- ■握手时间短
- ■经历较少的丢失
- ■安全性高， 能够有效抑制DOS攻击