هدیه پورقاسم – ۹۷۳۳۰۱۵

سوال ۱:

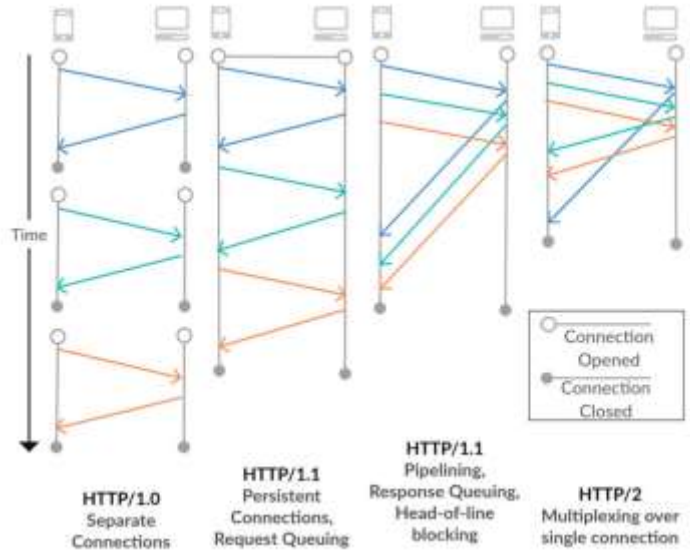
HTTP 1.0: non-persistent connection ، متدهای GET, HEAD, POST را ساپورت میکند، response آن برخلاف http 0.9 به hypertext محدود نیست و این محدودیت به کمک هدر content-type برداشته شده است. header هایی برای پیام های Request و response در نظر گرفته شده (HTTP version number, status code, content type).

HTTP 1.1: persistent and pipelined connection، متدهای GET, HEAD, POST, DELETE, PUT, TRACE, OPTIONS را ساپورت میکند.

مشکل pipeline در این ورژن:
پیاده سازی تقریبا غیرممکن
با اینکه درخواست ها موازی ارسال میشوند عملا جواب ها موازی برنمیگردند و باید برای دریافت هر جواب به نوبت نتظر ماند.

HTTP 2: persistent but with multiplexing and server push، پیام ها در فرمت باینری اند.

HTTP 3: به جای TCP از طریق QUIC اجرا می شود. QUIC یک پروتکل لایه انتقال است که سریعتر و ایمن تر از TCP است (QUIC ّراساس UDP است).



استفاده از TCP در HTTP2 مشکل head of line blocking problem را ایجاد میکند که این مشکل با استفاده از UDP حل میشود و سرعت افزایش میابد. همچنین ارتباط UDP مرحله ی connection setup را ندارد و کانکشن ها سریعتر است.

توضیح مشکل head of line blocking problem: با استفاده از TCP در http2 میتوان با درخواست های موازی روی یک ارتباط throughput و پهنای باند را به حداکثر رساند و در نتیجه TCP میتواند با سرعت زیادی کار کند. اما مشکل آنجا شروع میشود که یک قطعی کوچک در شبکه رخ دهد یا یک packet گم شود. در چنین حالتی چون TCP تضمین می‌کند که ترتیب ارسال بسته‌ها به ترتیبی است که توسط برنامه دریافت می‌شود - بنابراین اگر یک بسته از دست برود، همه چیز باید متوقف شود تا آن بسته خاص دوباره ارسال شود. اگر چندین درخواست را روی یک اتصال TCP مالتیپلکس کنیم، تمام این درخواست‌ها باید متوقف شوند و منتظر بمانند، حتی اگر بسته از دست رفته تنها بر یکی از آنها تأثیر بگذارد.

سوال ۲:

الف)پروتکل ب) http یک پروتکل stateless است و با استفاده از cookie و یا JWT ها مشکلات مربوط به آن را حل میکنند.

سوال ۳:

الف) در این کاربردها عموما سرور همواره د حال مانیتور کردن چیزی است و می‌خواهد در صورت پیش آمدن شرایط خاصی پیامی را برای کلاینت ارسال کند(مثلا alert بدهد) اما چون از سمت کلاینت ارتباطی شروع نشده و سرور هم نمیتواند requestای به سمت کلاینت ارسال کند چون در این کاربردها سمت کلاینت معمولا سروری وجود ندارد که درخواست را دریافت کند برای ارسال پیام از سمت سرور به کلاینت در اتصال http به چالش میخوریم.

ب) یکی از راه ها استفاده از websocket ها است. websocket یک پروتکل برای ارتباط کلاینت/سروری است مانند http با این تفاوت که bidirectional و full-duplex است همچنین stateful است و ارتباط بین کلاینت و سرور را نگه میدارد تا زمانی که ارتباط توسط کلاینت یا سرور بسته شود. از این پروتکل ابتدا از سمت کلاینت برای سرور request ارسال میشود و سپس سرور یک پیام handshake در جواب میدهد و یک connection جدید تشکیل میشود که به آن websocket میگویند.

در این صورت اگر یک با کلاینت به سرور درخوات داده باشد، دیگر سرور میتواند هر زمانی برای alert دادن به کلاینت پیام ارسال کند و چالش قسمت قبل از بین میرود.

سوال ۴:

ss://asghar:1234!!@ss.myproxy.com:1234\#shadowSocks1

<protocol (scheme)>:// <user> : <pass> @ <host> : <port>/<path>?<query>#<frag>

ss ------> scheme

Asghar : 1234 ------> <user> : <pass>

ss.myproxy.com:1234 ------> <host> : <port>

shadowSocks1 ------> <frag>

سوال ۵:

۱. اگر ما از مشکل باخبر باشیم و در واقع سرور خطا نداده باشد صرفا available نباشد، کد ۵۰۳ مناسب است و اگر سرور خطا داشته باشد(internal server error) باشد کد ۵۰۰ مناسب است.

۲. کد ۴۰۱: unauthenticated که مشابه ۴۰۳ Forbidden است، اما مخصوصاً برای استفاده در مواقعی که احراز هویت مورد نیاز است و ناموفق بوده یا هنوز ارائه نشده است.

۳. اگر جابجایی دامنه موقت در نظر گرفته شده باشد کد ۳۰۷ و اگر به طور دائم جابجا شده باشد کد۳۰۱ یا ۳۰۸

یا اگر قرار باشد کلاینت redirect شود و در هدر location آدرس جدید گذاشته شود کد ۳۰۲ داده میشود

۴. کد ۴۲۹: too many request، کاربر درخواست های بیش از حدی در یک بازه زمانی (rate limiting)داشته است

۵. کد ۲۰۰ یا ۲۰۱ که به ترتیب نشان دهنده موفقیت درخواست، ساخته شدن یک resource جدید است.

۶. کد ۴۰۳: forbidden ، اجازه‌ی دسترسی به ریسورس وجود نداره

سوال ۶:

۱.forward proxy

۲.Reverse proxy

۳.reverse proxy