سوال 1) پروتکل HTTP چگونه کار میکند؟

پروتکل (HyperText Transfer Protocol) HTTP یک پروتکل ارتباطی است که برای انتقال داده‌ها در وب استفاده می‌شود. این پروتکل اساساً برای انتقال صفحات وب از سرور به مرورگر کاربر طراحی شده است.

در ان هر درخواست مستقل از درخواست‌های قبلی است؛ سرور اطلاعاتی درباره درخواست‌های قبلی کاربر نگه نمی‌دارد.

HTTP از هدرها (Headers) استفاده می‌کند که می‌توانند برای ارسال اطلاعات اضافی بین کلاینت و سرور استفاده شوند.

مراحل کارکرد ان به شرح زیر است:

1.درخواست(Request)

- مرورگر کاربر یک درخواست HTTP به سرور ارسال می‌کند. این درخواست معمولاً شامل یک URL (آدرس وب) و متد HTTP مانند GET، POST، PUT، DELETE و غیره است.
- متد :GET برای درخواست داده‌ها از سرور (مثلاً یک صفحه وب).
- متد :POST برای ارسال داده‌ها به سرور (مثلاً اطلاعات فرم).

2.پردازش درخواست توسط سرور

- سرور درخواست را دریافت کرده و آن را پردازش می‌کند. این پردازش ممکن است شامل خواندن فایل‌ها، اجرای اسکریپت‌ها، یا دسترسی به پایگاه داده باشد.

3.پاسخ(Response)

- سرور پس از پردازش درخواست، یک پاسخ HTTP به مرورگر کاربر ارسال می‌کند. این پاسخ شامل یک کد وضعیت (Status Code)و محتوای درخواست شده مانندHTML ، CSS، JavaScript، تصاویر و غیره است.
- کدهای وضعیت رایج:
  - 200 OK: درخواست موفقیت‌آمیز بود.
  - 404 Not Found: منبع درخواست شده یافت نشد.
  - 500 Internal Server Error: خطایی در سرور رخ داده است.

4.نمایش محتوا توسط مرورگر

- مرورگر پاسخ سرور را دریافت کرده و محتوای آن را نمایش می‌دهد. این محتوا ممکن است شاملHTML ، CSS، JavaScriptو سایر منابع باشد که به صورت جداگانه درخواست می‌شوند.

5.اتصال بسته می‌شود

- پس از ارسال پاسخ، اتصال بین مرورگر و سرور بسته می‌شود؛ مگر از HTTP/1.1 با اتصال پایدار استفاده شود.

سوال 2) عملیات رمزنگاری در HTTPS و TLS چگونه انجام میشود؟

HTTPS (HyperText Transfer Protocol Secure) نسخه امن پروتکل HTTP است که از **TLS** یا نسخه قدیمی‌تر آن **SSL (Secure Sockets Layer)** برای رمزنگاری ارتباطات استفاده می‌کند.

این رمزنگاری باعث می‌شود داده‌ها بین کلاینت (مرورگر) و سرور به صورت امن منتقل شوند و از دسترسی غیرمجاز جلوگیری شود.

مراحل و نحوه انجام عملیات رمزنگاری در HTTPS و TLS به شرح زیر است:

۱ برقراری ارتباط امن(Handshake)

اولین مرحله در HTTPS ، انجام فرآیند **TLS Handshake** است. این فرآیند بین کلاینت و سرور انجام می‌شود و شامل مراحل زیر است:

۱.۱ Client Hello

- کلاینت (مرورگر) یک پیام **Client Hello** به سرور ارسال می‌کند.
- شامل اطلاعاتی مانند نسخه TLS مورد پشتیبانی کلاینت، لیست الگوریتم‌های رمزنگاری (Cipher Suites) و یک عدد تصادفی (Client Random) است.

۱.۲ Server Hello

- سرور به پیام کلاینت پاسخ می‌دهد و یک پیام **Server Hello** ارسال می‌کند.
- این پیام شامل نسخه TLS انتخاب شده، الگوریتم رمزنگاری انتخاب شده(Cipher Suite) ، یک عدد تصادفی (Server Random)و **گواهی دیجیتال سرور** (Server Certificate)است.
- گواهی دیجیتال شامل کلید عمومی سرور و اطلاعات هویتی سرور است که توسط یک مرکز صدور گواهی (CA) امضا شده است.

۱.۳ احراز هویت سرور

- کلاینت گواهی دیجیتال سرور را بررسی می‌کند تا مطمئن شود سرور معتبر است.
- شامل تأیید امضای گواهی توسط یک مرکز صدورگواهی معتبر(CA) و بررسی تاریخ انقضای گواهی است.

۱.۴ ارسال کلید عمومی و پیش‌راز(Pre-Master Secret)

- کلاینت یک **Pre-Master Secret** تولید می‌کند و آن را با استفاده از کلید عمومی سرور رمزنگاری می‌کند.
- این داده رمزنگاری شده به سرور ارسال می‌شود.

۱.۵ تولید کلیدهای رمزنگاری

- سرور با استفاده از کلید خصوصی خود، **Pre-Master Secret** را رمزگشایی می‌کند.
- سپس، هم کلاینت و هم سرور از **Pre-Master Secret**، **Client Random** و **Server Random** برای تولید **کلیدهای جلسه (Session Keys)** استفاده می‌کنند.
- این کلیدها برای رمزنگاری و رمزگشایی داده‌ها در طول جلسه ارتباطی استفاده می‌شوند.

۱.۶ اتمامHandshake

- کلاینت و سرور پیام‌های **Finished**را به یکدیگر ارسال می‌کنند تا تأیید کنند که Handshake با موفقیت انجام شده است.
- از این مرحله به بعد، تمام داده‌ها با استفاده از کلیدهای جلسه رمزنگاری می‌شوند.

۲ رمزنگاری داده‌ها

پس از تکمیلHandshake ، ارتباط امن برقرار می‌شود و داده‌ها با استفاده از الگوریتم‌های رمزنگاری تعیین شده در Cipher Suiteرمزنگاری می‌شوند. این الگوریتم‌ها معمولاً شامل موارد زیر هستند:

۲.۱ رمزنگاری متقارن(Symmetric Encryption)

- از یک کلید مشترک (Session Key) برای رمزنگاری و رمزگشایی داده‌ها استفاده می‌شود.
- الگوریتم‌های رایج**AES (Advanced Encryption Standard)** :، **ChaCha20**.

۲.۲ رمزنگاری نامتقارن(Asymmetric Encryption)

- در مرحله Handshake از رمزنگاری نامتقارن (با استفاده کردن از کلید عمومی و خصوصی) برای انتقال ایمن **Pre-Master Secret**استفاده می‌شود.
- الگوریتم‌های رایج**RSA** :، **ECDSA**، **DH (Diffie-Hellman)**.

۲.۳ توابع درهم‌سازی(Hash Functions)

- برای تأیید یکپارچگی داده‌ها و ایجاد امضای دیجیتال استفاده می‌شوند.
- الگوریتم‌های رایج**SHA-256** :، **SHA-384**.

۳ انتقال داده‌ها

پس از برقراری ارتباط امن، داده‌ها بین کلاینت و سرور به صورت رمزنگاری شده منتقل می‌شوند. این داده‌ها شامل درخواست‌های) HTTP مانند GET یا (POST و پاسخ‌های سرور) مانندHTML ، JSONو غیره (هستند.

۴ اتصال بسته می‌شود

پس از اتمام انتقال داده‌ها، اتصال TLS بسته می‌شود. اگر کلاینت بخواهد دوباره به سرور متصل شود، فرآیند Handshake ممکن است تکرار شود یا از **Session Resumption**برای استفاده مجدد از کلیدهای جلسه قبلی استفاده شود.

سوال 3) ارتباط بین **کلاینت** و **سرور** در HTTP چگونه انجام میشود؟

ارتباط بین کلاینت (مانند مرورگر وب) و سرور در پروتکل **HTTP (HyperText Transfer Protocol)** از طریق یک سری درخواست‌ها (Requests) و پاسخ‌ها (Responses) انجام می‌شود. این ارتباط مراحل زیر را شامل می‌شود:

۱ .برقراری اتصال

- کلاینت (مرورگر) یک اتصال به سرور برقرار می‌کند. این اتصال معمولاً از طریق **پورت** ۸۰ برای HTTP یا **پورت** ۴۴۳ برای HTTPS انجام می‌شود.
- اگر از HTTPS استفاده شود، ابتدا یک فرآیند **TLS Handshake** برای رمزنگاری ارتباط انجام می‌شود.

۲ .ارسال درخواست(Request)

- کلاینت یک درخواست HTTP به سرور ارسال می‌کند. این درخواست شامل موارد زیر است:
  - **متد HTTP** : نوع عملیاتی که کلاینت انجام می دهد مانندGET ، POST، PUT، DELETE وغیره.
  - **URL**: آدرس منبعی که کلاینت درخواست می‌کند مانند /index.html
  - **هدرها**: اطلاعات اضافی مانند نوع مرورگر، کوکی‌ها، زبان مورد نظر و غیره.
  - **بدنه :** در برخی درخواست‌ها مانندPOST ، داده‌های اضافی در بدنه ارسال می‌شوند.

۳ .پردازش درخواست توسط سرور

- سرور درخواست کلاینت را دریافت کرده و آن را پردازش می‌کند. این پردازش ممکن است شامل موارد زیر باشد:
  - خواندن یک فایل از دیسک مانند یک صفحه HTML
  - اجرای یک اسکریپت سمت سرور مانند PHP یا Python
  - دسترسی به یک پایگاه داده برای دریافت اطلاعات.

۴ .ارسال پاسخ(Response)

- سرور پس از پردازش درخواست، یک پاسخ HTTP به کلاینت ارسال می‌کند. این پاسخ شامل موارد زیر است:
  - **کد وضعیت (Status Code)**: نشان‌دهنده نتیجه درخواست (مانند ۲۰۰ برای موفقیت، ۴۰۴ برای یافت نشدن منبع و غیره).
  - **هدرها (Headers)**: اطلاعات اضافی مانند نوع محتوا، طول محتوا، کوکی‌ها و غیره.
  - **بدنه (Body)**: محتوای درخواست شده مانندHTML ، JSON، تصاویر و غیره.

۵ .نمایش محتوا توسط کلاینت

- کلاینت (مرورگر) پاسخ سرور را دریافت کرده و محتوای آن را پردازش می‌کند. این محتوا شامل موارد زیر است:
  - **HTML**ساختار صفحه وب و **CSS** استایل‌های ظاهری صفحه.
  - **JavaScript** کدهای تعاملی.
  - **تصاویر، ویدئوها و سایر منابع** :که ممکن است در درخواست‌های جداگانه دریافت شوند.

۶ .بستن اتصال

- پس از ارسال پاسخ، اتصال بین کلاینت و سرور بسته می‌شود (مگر از **HTTP/1.1با اتصال پایدار** استفاده شود).
- در HTTP/1.1 ، اتصال می‌تواند برای چندین درخواست و پاسخ باز بماند تا از ایجاد تاخیر در برقراری اتصال جدید جلوگیری شود.

با این مکانیزم‌ها، ارتباط بین کلاینت و سرور در HTTP به صورت کارآمد و قابل اعتماد انجام می‌شود.

سوال 4) چه تفاوتی بین HTTP/1.1 و HTTP/2 وجود دارد؟

ویژگی	HTTP/2	HTTP/1.1
Multiplexing (چندبرابری درخواست‌ها)	✅ دارد	❌ ندارد
فشرده‌سازی هدرها	✅ دارد(HPACK)	❌ ندارد
اتصال پایدار(Keep-Alive)	✅ بهینه‌تر شده است	✅ دارد اما محدود است
اولویت‌بندی درخواست‌ها	✅ دارد	❌ ندارد
Head-of-Line Blocking	✅ خیر (رفع شده)	❌ بله (مشکل دارد)
پشتیبانی ازServer Push	✅ دارد	❌ ندارد
نوع ارتباط	باینری(Binary)	متنی(Text-based)
سرعت و بهینه‌سازی شبکه	سریع‌تر	کندتر

HTTP/2 نسخه‌ی بهینه شده‌ی HTTP/1.1 است که باعث کاهش تأخیر، افزایش سرعت بارگذاری صفحات و بهینه‌سازی مصرف منابع سرور می‌شود. در دنیای امروزی، بیشتر وب‌سایت‌ها و مرورگرهای مدرن از HTTP/2 پشتیبانی می‌کنند.