Tema 1

El desarrollo web

MEDAC
Instituto Oficial de Formación Profesional

El desarrollo web

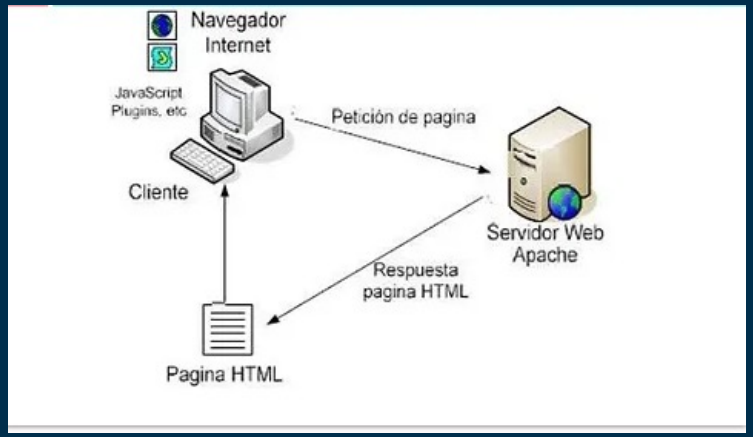
Se trata de entender los conceptos de Arquitectura y Desarrollo Web enfocándonos en uno de los elementos (el cliente) y comenzando a presentar las herramientas de desarrollo web.

Lo que vamos a aprender:

- Entender la arquitectura cliente / servidor
- Conocer los protocolos básicos de comunicación: HTTP, HTTPS
- Conocer las herramientas de desarrollo web

Fundamentos del desarrollo web

- Un sitio web es un conjunto de archivos que se almacenan en una máquina, normalmente denominada servidor.
- Cuando el servidor está conectado a Internet es posible que otras aplicaciones, conocidas como navegadores, puedan conectarse y visualizar el contenido
- Cada una de las máquinas que se conecta recibe el nombre de cliente

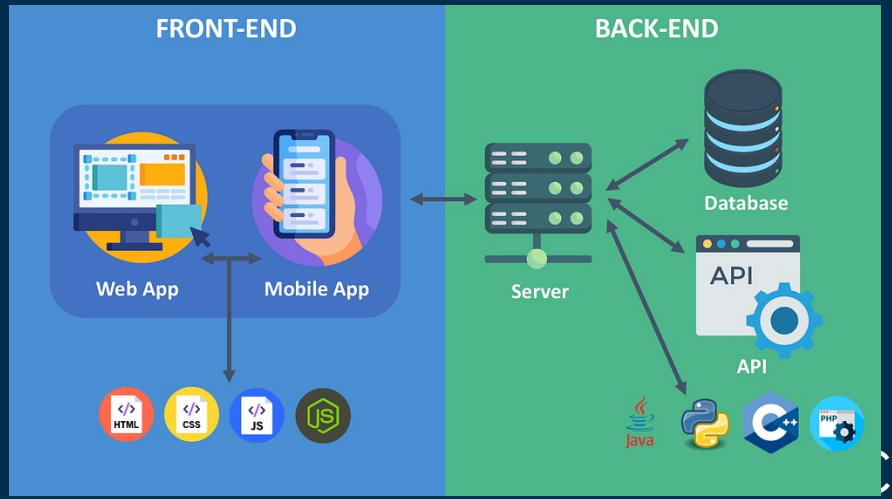


MEDAC

Fundamentos del desarrollo web

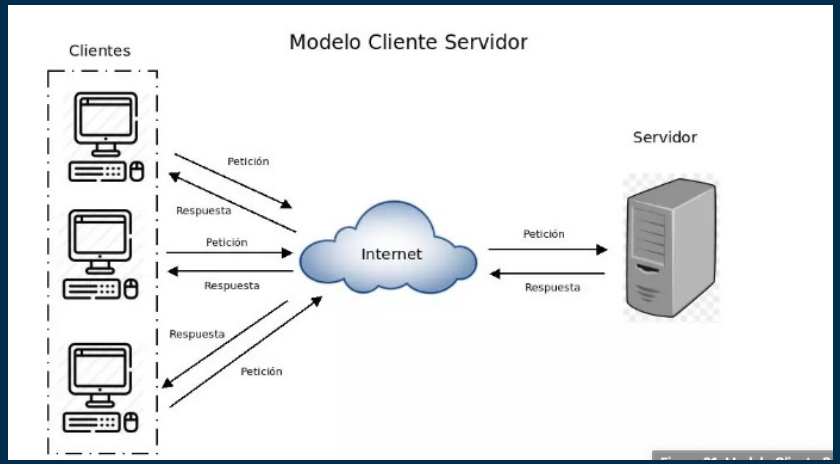
Como desarrollador web podemos trabajar en distintos ámbitos:

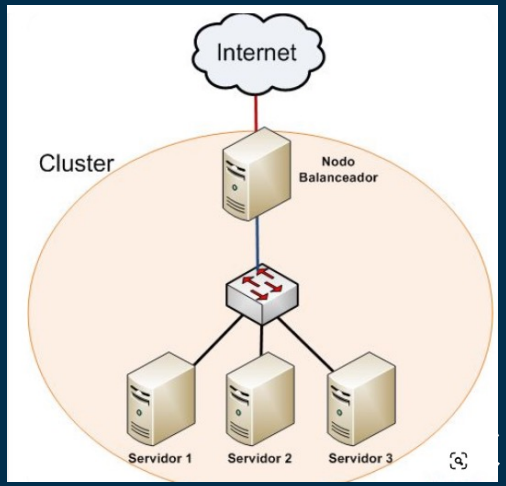
- En el servidor – Back-end: parte no visible de la web, como la base de datos o los scripts que se ejecutan en el servidor
- En el cliente – Front-end (debemos conocer HTML, CSS y JavaScript): parte visible de una web como las hojas de estilo, el código HTML y los scripts que se ejecutan en el lado del cliente. Creatividad y originalidad.
- En el cliente y servidor - Full stack



Arquitectura cliente / servidor

La principal importancia de este modelo es que permite conectar a varios clientes a los servicios que provee un servidor. Hoy en día, la mayoría de las aplicaciones y servicios tienen como gran necesidad poder ser consumidos por varios usuarios de forma simultánea.
El modelo cliente servidor se fundamenta en la realización de peticiones por parte del cliente y la generación de respuesta por parte del servidor.





Arquitectura cliente / servidor

- **Cliente**. El cliente es un dispositivo conectado a una red, cuya función es realizar peticiones al servidor. Estas peticiones pueden ser consultas o solicitudes de servicios, tales como la descarga de archivos o la transmisión de datos. Normalmente, los clientes se encuentran en dispositivos como PCs, teléfonos móviles y tabletas, entre otros.

- **Servidor**. El servidor es el encargado de responder a las peticiones del cliente y proporcionar los servicios que se le hayan solicitado. Un servidor también puede almacenar y administrar información para su posterior acceso por parte del usuario. Suelen trabajar con sistemas operativos como Linux o Windows Server.

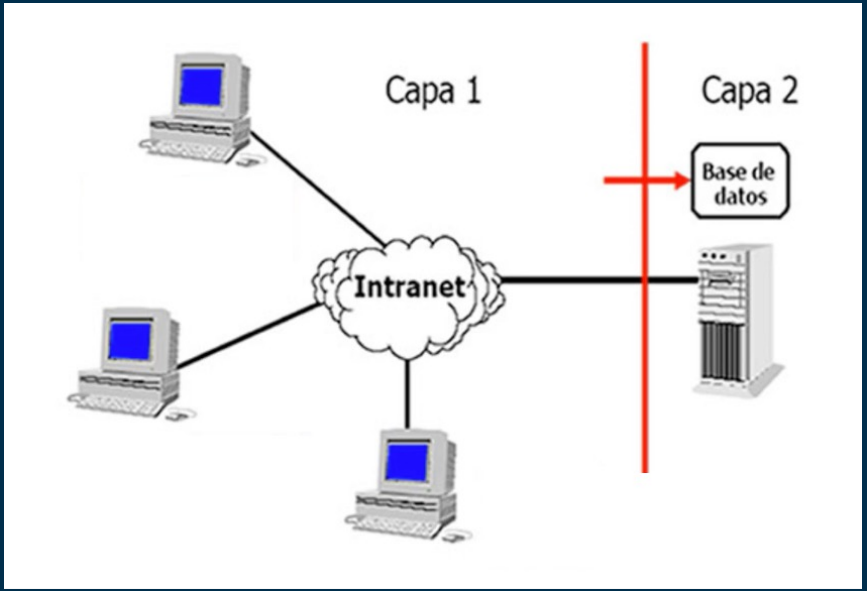
MEDAC

Tipos de arquitecturas

Dependiendo de las tareas que realice el servidor podemos tener distintas arquitecturas:

Arquitectura en dos capas.

Es el modelo por defecto. El servidor, al recibir una petición, es capaz de responder con sus propios recursos.. Esto significa que el servidor no requiere de una aplicación extra para proporcionar parte del servicio.

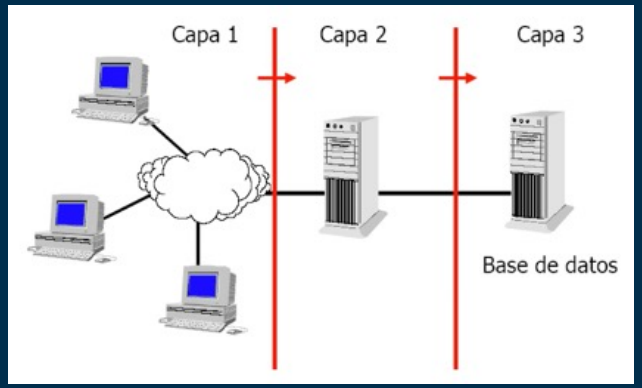


MEDAC

Tipos de arquitecturas

Arquitectura en tres capas.

En este caso el servidor requiere el acceso a un componente externo como puede ser un servidor de datos. Un cliente con su navegador o interfaz de usuario solicita un recurso. La capa del medio se encarga de proporcionar los recursos solicitados pero requiere de otro servidor para hacerlo. La última capa es el servidor de datos que proporciona al servidor de aplicaciones los datos necesarios para poder procesar y generar el servicio que solicitó el cliente en un principio. (cliente – servidor intermedio o de aplicaciones – servidor de datos)

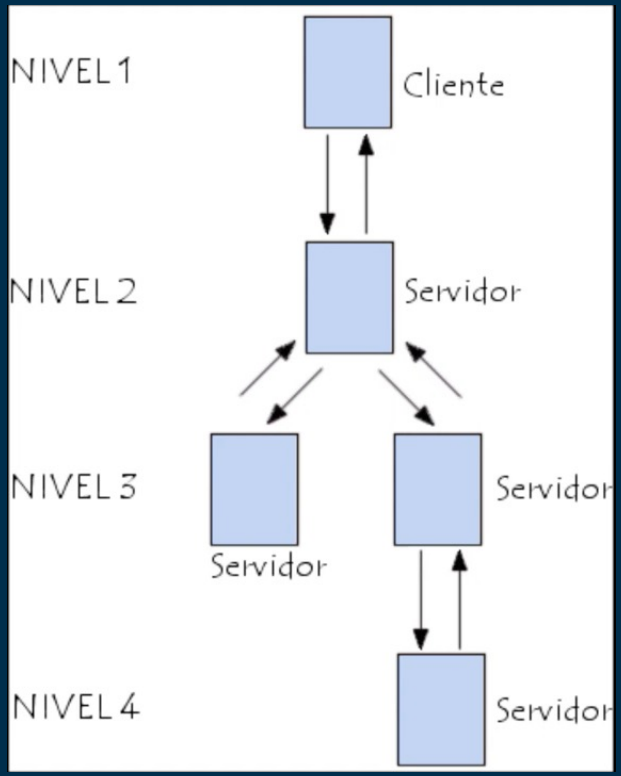


MEDAC

Tipos de arquitecturas

Arquitectura en N capas.

este escenario supondría la generalización del modelo en el que se podrían introducir tantas capas de datos de procesamiento como el problema requiera

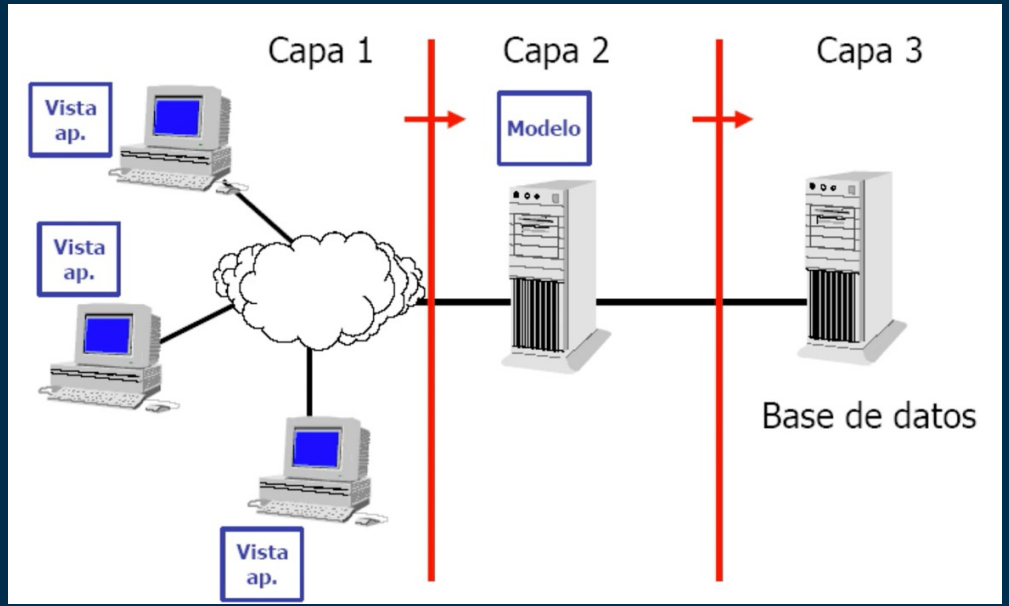


MEDAC

Arquitectura de aplicaciones web

Arquitectura en 3 capas.

- **Capa de presentación**: mostrar la información al usuario (cliente)
- **Capa de lógica de negocio**: procesa la información
- **Capa de acceso a datos**: almacena y recupera información



MEDAC

Ventajas y desventajas de la arquitectura cliente-servidor

Aunque la arquitectura cliente-servidor ofrece muchas ventajas, también hay algunas desventajas asociadas con ella.

1. Ventajas:
   a. **Mayor escalabilidad**. La capacidad de agregar más servidores a la red significa que puede manejar cargas de trabajo mucho mayores sin disminuir el rendimiento.
   b. **Independencia de plataforma**. Los usuarios pueden utilizar el sistema operativo que deseen.
   c. **Mejor seguridad**. Los datos se almacenan en servidores centralizados. En consecuencia, es más fácil protegerlos contra intrusiones externas.
   d. **Accesibilidad remota**. Los usuarios pueden acceder a los recursos y servicios desde cualquier lugar con conexión a Internet.
2. Desventajas:
   a. **Vulnerabilidad ante ataques externos**. Al estar todos los recursos concentrados en un solo lugar, puede ser más fácil para los hackers atacarlos todos a la vez.
   b. **Dependencia del ancho de banda**. Si el ancho de banda es bajo o inestable, el rendimiento general del sistema será menor.
   c. **Más coste**: puede resultar una solución costosa porque los servidores suelen requerir un alto nivel de hardware

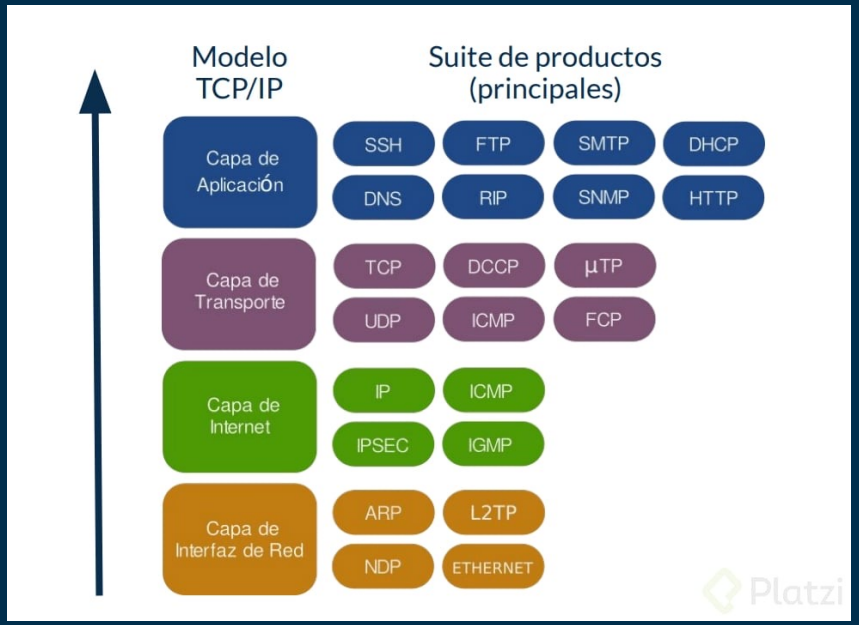
Ejemplos de aplicación de la arquitectura cliente-servidor

1. Ejemplos de aplicación de la arquitectura cliente-servidor son:

- **Correo electrónico**. En el caso del correo electrónico, el usuario utiliza su dispositivo cliente para conectarse al servidor que almacena todos los mensajes enviados y recibidos. El usuario puede leer y escribir mensajes desde su dispositivo sin necesidad de acceder directamente al servidor.
- **Navegación web**. La navegación web es un buen ejemplo de la arquitectura cliente-servidor donde el navegador web actúa como el dispositivo cliente para conectarse a los servidores donde se alojan las páginas web. Una vez que se establece la conexión, el navegador envía peticiones al servidor solicitando contenido que luego se muestra en pantalla.
- **Impresión remota**. Del mismo modo, es posible imprimir documentos desde cualquier lugar del mundo mediante la configuración de impresoras remotas. Lo que es posible gracias a la arquitectura cliente-servidor. Esta configuración permite que los documentos sean enviados directamente desde tu equipo hasta la impresora remota sin tener que instalar software adicional en tu equipo.
- **Compartición de archivos**. La compartición de archivos también forma parte del concepto de arquitectura Cliente-Servidor. Los usuarios acceden desde sus equipos a los archivos compartidos alojados en determinado servidor sin necesidad de transferir físicamente dichos archivos entre ellos mismos.

Protocolo de comunicaciones

Protocolo: Reglas necesarias para que unos dispositivos se comuniquen con otros. Estos protocolos se organizan en niveles o capas, cada uno con una función. En este modelo por niveles, cada uno de ellos usa las funcionalidades del nivel inmediatamente inferior.

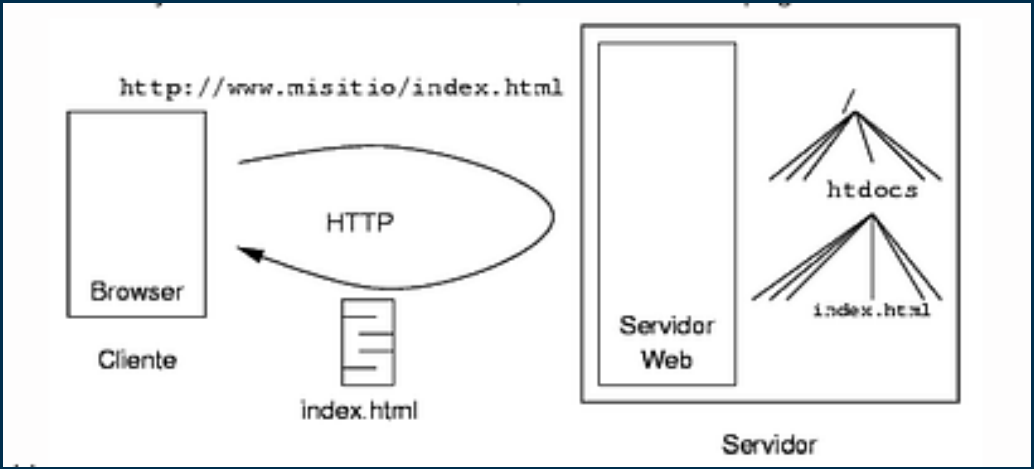


- **Nivel de red**: medio físico – routers, switches
- **Nivel de internet**: asegura que los datos lleguen a la dirección correcta
- **Nivel de transporte:** asegura el orden de los datos
- **Nivel de aplicación:** interacción directa entre clientes y servidores

MEDAC

Protocolo HTTP

En entornos web el protocolo utilizado para la transferencia de contenido web es HTTP (Hypertext Transfer Protocol). Este protocolo funciona mediante un sistema de peticiones respuesta dentro de una arquitectura cliente servidor.



MEDAC

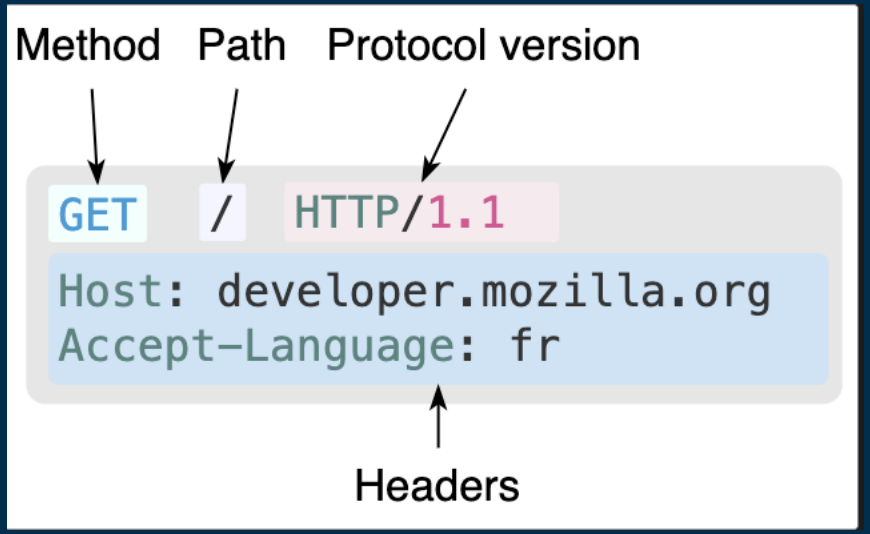
Solicitud HTTP (request)

La solicitud es lo que el cliente necesita del servidor. Ese mensaje incluye datos específicos para describir lo que se solicita. Los principales componentes de una solicitud son:

- Método (verbo): acción que quiere realizar el cliente
- Ruta: el recurso que estamos buscando
- Versión del protocolo HTTP
- Encabezados: información adicional para los servidores (codificación, idioma, tipo de contenido…)
- Cuerpo del mensaje: Es opcional, pero en algunos métodos como en el POST es donde va la información que queremos guardar.

MEDAC

Solicitud HTTP (request)



METODOS

- GET: obtener recursos
- POST: guardar datos en el servidor
- PUT: actualizar un registro existente
- DELETE: borrar un recurso

MEDAC

Solicitud HTTP (request)

```
GET /index.html HTTP/1.1
Host: www.example.com
Referer: www.google.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:45.0)
Gecko/20100101 Firefox/45.0
Connection: keep-alive
```

HTTP

```
1  POST /createCall HTTP/1.1
2  Host: 0.0.0.0:8080
3  Content-Type: application/json
4  Content-Length: 182
5
6  {"datetime":"5/3/2022 6:53 PM",
       "number":"123456789",
       "direction":"Outbound",
       "calltype":"Notanswered",
       "agent":"1000",
       "agentfirstname":"Carlos",
       "agentlastname":"Ros",
       "duration":"00:28"}
```

MEDAC

Respuesta HTTP (response)

La respuesta dada por el servidor contiene la información solicitada por el cliente o le informa si hay un error respecto a lo que ha solicitado. Contiene los siguientes elementos:

- Versión del protocolo HTTP
- Código de estado y mensaje de estado : indica cómo se ha procesado la petición
- Encabezados: información adicional para el cliente
- Cuerpo del mensaje: Opcional y contiene los datos sobre el recurso solicitado por un ejemplo en un GET sería obligatorio

Respuesta HTTP (response)

```
HTTP/1.1 200 OK
Date: Fri, 31 Dec 2003 23:59:59 GMT
Content-Type: text/html
Content-Length: 1221

<html lang="eo">
<head>
<meta charset="utf-8">
<title>Título del sitio</title>
</head>
<body>
<h1>Página principal de tuHost</h1>
(Contenido)
   .
   .
   .
</body>
</html>
```

Códigos de estado

- 1XX: Respuestas informativas (procesando)
- 2XX: Ejecución correcta
- 3XX: Redirección
- 4XX: Errores en el cliente
- 5XX Errores en el servidor

MEDAC

HTTP en la práctica

- Navegador: Herramientas de desarrollador/Inspeccionar – Network

- User-Agent: Contiene un string característico que será examinado por el protocolo de red para identificar el tipo de aplicación, sistema operativo, proveedor de software o versión del software del agente de software que realiza la petición Link

MEDAC

Práctica

Revisa qué significan los siguientes campos de cabecera y qué valores puede tener y responde a las preguntas

1.Accept
2.Content-Type
3.Cache-Control
4.Content-Length
5.Date
6.Referer
7.Accept-Encoding

Accede a una página web
1.¿Qué método de solicitud utilizaste?
2.¿Qué código de estado HTTP respondió?
3.¿Qué tipo de información acepta el cliente? MIME
4.¿Qué tipo de información está enviando el servidor? MIME
5.¿En qué fecha y a qué hora te respondió?
6.¿Está la información comprimida?
7.¿Ha utilizado algún tipo de caché?
8.¿Qué agente de usuario estás usando?

Link

MEDAC

Práctica

Accept: Informa al servidor sobre los diferentes tipos de datos que pueden enviarse de vuelta. Es de tipo MIME (tipo/subtipo). Link

Ejemplo:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7

MEDAC

Práctica

Content-Type: Indíca el tipo (MIME) del recurso que va en el body

Ejemplo: text/html; charset=UTF-8

MEDAC

Práctica

Cache-Control: Especifica directivas para los mecanismos de almacenamiento en caché, tanto para peticiones como para respuestas.

Ejemplo: private, max-age=0

- private: Indica que la respuesta es específica para un único usuario y no debe almacenarse en cachés compartidas
- public: Indica que la respuesta puede ser almacenada en cualquier caché, tanto en cachés de cliente (navegadores) como en cachés compartidas (proxies, CDN).
- no-store: No se debe almacenar la respuesta en absoluto, ni en el cliente ni en cachés compartidas. Es útil para datos sensibles, como transacciones financieras o datos personales.

Práctica

Content-Length: tamaño del cuerpo (body) del mensaje en bytes

Ejemplo: Content-Length: 62143

Date: Contiene la fecha y hora en que el mensaje de respuesta fue originado

Referer: Indica la dirección de la página web previa desde la cual un link nos ha redirigido a la actual.

MEDAC

Práctica

Accept-Encoding: Informa al servidor sobre el algoritmo de codificación, habitualmente un algoritmo de compresión, que puede utilizarse sobre el recurso que se envíe de vuelta en la respuesta.

Ejemplo:

Request: Accept-Encoding: gzip, deflate, br, zstd
Response Content-Encoding: br

MEDAC

Seguridad en la comunicación web

Hablamos de que un sistema de comunicaciones es seguro cuando la transmisión que se lleva a cabo entre dos entidades no puede ser interceptada por un tercer individuo. Además, durante este proceso de comunicación debemos asegurarnos de que el destinatario es realmente quien dice ser y que el mensaje que se recibe es el que realmente hemos enviado.

- HTTP: no es seguro, es decir la información que se envía o recibe se transmite sin cifrar por lo que cualquiera que estuviera espiando tu conexión podría ver esa información.
- **HTTPS**: es una versión segura de HTP. Funciona de la misma manera que HTTP pero cifra la información que se envía y recibe. Esto significa que incluso si alguien intercepta los datos no podrá leerlo ni entenderlo, porque están encriptados.

MEDAC

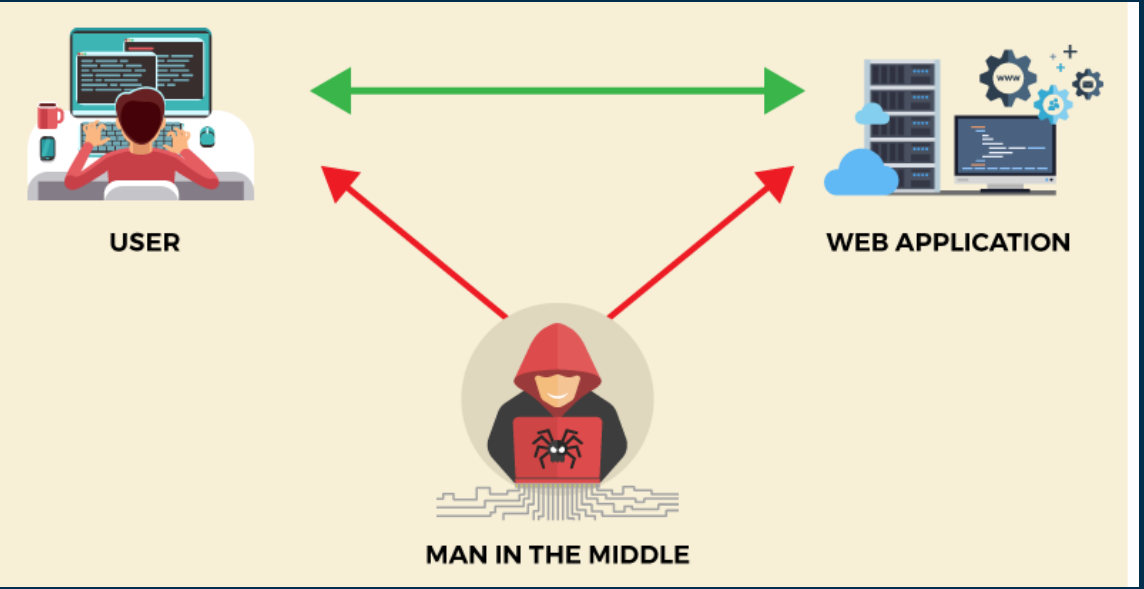
HTTPS

Para usar HTTPS, un sitio web necesita un certificado SSL o TLS. Este certificado garantiza que la información esté cifrada y asegura que el sitio web es legítimo (DigiCert, GlobalSign, GoDaddy…). Los sitios web que usan HTTPS son más confiables. Los navegadores suelen mostrar un candado verde en la barra de direcciones cuando estás en un sitio seguro, lo que te da una señal de que tus datos están protegidos.

Aunque sea un protocolo más seguro la tecnología y la capacidad de cómputo avanza y los ciberdelicuentes pueden llegar a detectar vulnerabilidades en los protocolos que poder atacar. Por ejemplo, se ha observado que SSL puede sufrir ataques de MID (Man in the Middel)

MEDAC

Man in the Middle



MEDAC

Otros protocolos usados en red

Otros protocolos que se utilizan en desarrollo de cliente son: FTP y SSH

- FTP (File Transfer Protocol): Es un protocolo para poder realizar transferencia de archivos entre un cliente y un servidor a través de la red. En el desarrollo web lo podemos utilizar para publicar la última versión de la web creada. Hay un cliente FTP en todos los sistemas operativos. (FileZilla, WinSCP)

- SSH (Secure Shell): Es un protocolo de red utilizado para acceder de manera segura a un dispositivo remoto, como un servidor. Suele usarse en línea de comandos para interactuar con el sistema remoto. Para conectar con este sistema se nos pedirá un nombre de usuario y contraseña. Un cliente SSH se encuentra disponible para todos los sistemas operativos de propósito general (PuTTY)

MEDAC

MEDAC

Instituto Oficial de Formación Profesional