2.2 La Web y HTTP 15-37.

TCP es un protocolo orientado a conexión que ofrece un servicio muy fiable.

UDP no está orientado a conexión y ofrece un servicio poco fiabl

1990 -apareció en escena una nueva aplicación importante: la **World Wide Web** (www)

Web fue la primera aplicación de Internet que llamó la atención del público en general.

Web: información que se encuentra en una dirección determinada de internet (costo extremadamente bajo.). Sirve como protocolo de las páginas.

HTTP.

Protocolo de transferencia de hipertexto (HTTP, HyperText Transfer Protocol).

Definido en [RFC 1945], [RFC 7230] y [RFC 7540].

se implementa en dos programas: un programa cliente y un programa servidor.

Página web (también llamado documento) consta de objetos. Un objeto es simplemente un archivo, como un archivo HTML, una imagen JPEG, un archivo Javascrpt, etc. que se puede direccionar mediante una única URL.

Cada URL (Localizador de recursos uniforme) tiene **dos componente**s: el **nombre de hos**t del servidor que alberga el objeto y el **nombre de la ruta** del objeto.

Navegador web implementa al cliente y Servidores web implementan el lado del servidor.

Proceso:

- El cliente HTTP primero **inicia una conexión** TCP con el servidor. Una vez que se establece la conexión, el navegador y los procesos del servidor **acceden a TCP** a través de sus interfaces de socket.
- La **interfaz de socket** es la **puerta** entre el proceso del cliente y la conexión TC.
- En el lado del servidor, es la puerta entre el proceso del servidor y la conexión **TCP**.
- HTTP no necesita preocuparse por la pérdida de datos o los detalles de cómo TCP se recupera de la pérdida o el reordenamiento de datos dentro de la red.

HTTP es un protocolo sin estado (**No almacena info**).
La versión original de HTTP se llama **HTTP / 1.0** y se remonta a principios de la década de 1990.

servidores web también admiten una nueva versión de **HTTP llamada HTTP / 2**.

Conexiones persistentes y no persistentes.

Cuando esta **interacción cliente-servidor** se lleva a cabo a **través de TCP**, el desarrollador de la aplicación debe tomar una decisión importante, en caso de que cada par de solicitud / respuesta se envíe a través de un separar (separate) Conexión TCP, o todas las solicitudes y sus correspondientes respuestas deben enviarse a través del mismo Conexión TCP. En el primer enfoque, se dice que la aplicación utiliza **conexiones no persistentes**; y en el último enfoque, **conexiones persistentes**.

HTTP usa conexiones persistentes en su modo predeterminado

no persistentes: cada conexión TCP se cierra después de que el servidor envía el objeto; la conexión no persiste para otros objetos.

El uso de conexiones paralelas acorta el tiempo de respuesta.

Round-trip time (RTT) – Tiempo de viaje de un paquete de cliente a servidor y regreso.

Protocolo de enlace de tres vías: el cliente envía un pequeño segmento de TCP al servidor,

el servidor reconoce y responde con un pequeño segmento de TCP y, finalmente, el cliente

responde al servidor.

Desventajas de no persistentes: se debe establecer y mantener una nueva conexión para cada objeto solicitado, se debe establecer y mantener una nueva conexión para cada objeto solicitado (genera una carga en el servidor) y también un retraso en la entrega de dos RTT.

Persistentes:

HTTP / 1.1: el servidor deja abierta la conexión TCP después de enviar una respuesta.

Las solicitudes y respuestas posteriores entre el mismo cliente y servidor se pueden enviar a través de la misma conexión.

Se pueden enviar varias páginas web que residen en el mismo servidor al mismo cliente a través de una única conexión TCP persistente.

Canalización (pipelining): solicitudes de objetos se pueden realizar una tras otra, sin esperar las respuestas a las solicitudes pendientes.

el servidor HTTP cierra una conexión **cuando no se utiliza durante un tiempo** determinado.

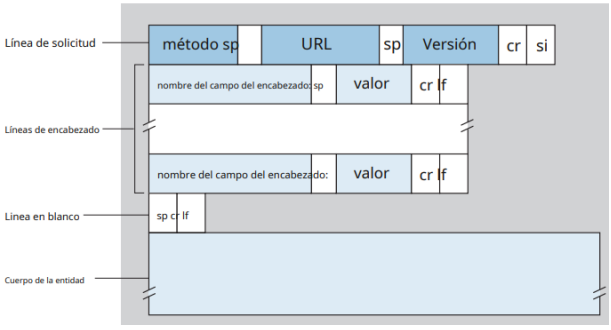
Formato de mensaje HTTP.

Hay **dos tipos** de **mensajes** HTTP, mensajes de solicitud y mensajes de respuesta.

```
GET /somedir/page.html HTTP/1.1
Host: www.someschool.edu
Connection: close
User-agent: Mozilla/5.0
Accept-language: fr
```

La **primera línea** de un mensaje de solicitud HTTP se llama **línea de solicitud** (request line), las líneas siguientes se denominan **líneas de encabezado** (header lines).

La **línea de solicitud tiene tres campos**: el campo de método, el campo de URL y el campo de versión HTTP. El campo de método puede tomar varios valores diferentes, incluidos OBTENER, PUBLICAR, DIRIGIR, PONER, y ELIMINAR (GET, POST, HEAD, PUT, and DELETE).



La **línea de encabezado especifica** el agente de usuario, es decir, el tipo de navegador que realiza la solicitud al servidor. Finalmente, el Aceptar-idioma.

El **código de estado** y la frase asociada indican el resultado de la solicitud.

- 200 OK: La solicitud se realizó
- 301 Movido Permanentemente: El objeto solicitado se ha movido de forma permanente; la nueva URL se especifica enLocalización: encabezado del mensaje de respuesta.
- 400 petición Incorrecta: Este es un código de error el servidor no pudo entender la solicitud.
- 404 No encontrado: El documento solicitado no existe en este servidor.
- 505 Versión HTTP no admitida: El servidor no admite la versión del protocolo HTTP solicitada.

La especificación HTTP define muchas, muchas más líneas de encabezado que los navegadores, servidores web y servidores de caché de red pueden insertar.

Un navegador generará líneas de encabezado en función del tipo y la versión del navegador, la configuración del usuario del navegador y si el navegador tiene actualmente una versión en caché, pero posiblemente desactualizada, del objeto.

Interacción usuario-servidor: cookies.

A menudo es deseable que un sitio web identifique a los usuarios, ya sea porque el servidor desea restringir el acceso de los usuarios o porque quiere servir contenido en función de la identidad del usuario, para esto HTTP utiliza cookies.

Cookies: permiten que los sitios realicen un seguimiento de los usuarios.

Cookies tiene cuatro **componentes**:

- línea de encabezado de cookie en el mensaje de respuesta HTTP.
- una línea de encabezado de cookie en el mensaje de solicitud HTTP.
- archivo de cookies guardado en el sistema final del usuario y administrado por el navegador del usuario.
- una base de datos back-end en el sitio web.

El servidor crea un número de identificación único y crea una entrada en su base de datos de back-end que está indexada por el número de identificación.

Ejemplo: Set-cookie: 1678.

1. Cuando el navegador de Susan recibe el mensaje de respuesta HTTP, ve el Conjunto de cookies.
2. el navegador agrega una línea al archivo de cookie especial que administra.
3. Esta línea incluye el nombre de host del servidor y el número de identificación en elConjunto de cookies: encabezamiento.
4. Mientras Susan continúa navegando por el sitio de Amazon, cada vez que solicita una página web, su navegador consulta su archivo de cookies, extrae su número de identificación para este sitio y coloca una línea de encabezado de cookie que incluye el número de identificación en la solicitud HTTP.
5. De esta manera, el servidor de Amazon puede rastrear la actividad de Susan.

6. Amazon no necesariamente conoce el nombre de Susan, ¡sabe exactamente qué páginas visitó el usuario 1678, en qué orden y en qué momento.
7. Si Susan también se registra en Amazon, proporcionando el nombre completo, la dirección de correo electrónico, la dirección postal y la información de la tarjeta de crédito, Amazon puede incluir esta información en su base de datos, asociando así el nombre de Susan con su número de identificación.
8. Así es como Amazon y otros sitios de comercio electrónico ofrecen “compras con un clic”

Las cookies **se pueden utilizar para** crear una capa de sesión de usuario sobre HTTP sin estado.

Aunque las cookies a menudo simplifican la experiencia de compra en Internet para el usuario, son controvertidas porque también pueden considerarse una invasión de la privacidad.

Almacenamiento en caché web.

Caché web—También llamado **servidor proxy**: Es una entidad de red que satisface las solicitudes HTTP en nombre de un servidor web de origen

Tiene su propio almacenamiento en disco y mantiene copias de los objetos solicitados recientemente

Proceso:

1. El navegador establece una conexión TCP a la caché web y envía una solicitud.
2. La caché web comprueba si tiene una copia del objeto almacenada localmente. Si es así, la caché web devuelve el objeto dentro de un mensaje de respuesta HTTP al navegador del cliente.
3. Si la caché web no tiene el objeto, la caché web abre una conexión TCP al servidor de origen, es decir, a www.someschool.edu.
4. luego envía una solicitud HTTP para el objeto en la conexión TCP de caché a servidor.
5. el servidor de origen envía el objeto dentro de una respuesta HTTP a la caché web.
6. Cuando la caché web recibe el objeto, almacena una copia en su almacenamiento local y envía una copia, dentro de un mensaje de respuesta HTTP

Una **caché es un servidor y un cliente** al mismo tiempo. Cuando recibe solicitudes y envía respuestas a un navegador, es un servidor. Cuando envía solicitudes y recibe respuestas de un servidor de origen, es un cliente

un ISP compra e instala una caché web. Por ejemplo, una universidad puede instalar un caché en la red de su campus y configurar todos los navegadores del campus para que apunten al caché.

El almacenamiento en caché web se ha **implementado** en Internet **por dos razones**:

- una caché web puede reducir sustancialmente el tiempo de respuesta.
- los cachés web pueden reducir sustancialmente el tráfico en el enlace de acceso de una institución a Internet.

Muchos cachés utilizan software de dominio público que se ejecuta en PC de bajo costo.

Redes de distribución de contenido (CDN), Los cachés web juegan un papel cada vez más importante en Internet. Una empresa de CDN instala muchos cachés distribuidos geográficamente en Internet, localizando así gran parte del tráfico.

El GET condicional.

Problema de almacenamiento en cache: la copia de un objeto que reside en la caché puede estar obsoleta. es posible que el objeto alojado en el servidor web se haya modificado desde que la copia se almacenó en caché en el cliente.

GET condicional: mecanismo que permite que una caché **verifique que sus objetos estén actualizados**.

Proceso ejemplo:

1. en nombre de un navegador solicitante, un caché proxy envía un mensaje de solicitud a un servidor web.
2. el servidor web envía un mensaje de respuesta con el objeto solicitado a la caché.
3. La caché reenvía el objeto al navegador solicitante, pero también almacena el objeto localmente. Es importante destacar que el caché también almacena la fecha de la última modificación junto con el objeto.
4. una semana después, otro navegador solicita el mismo objeto a través del caché y el objeto todavía está en el caché. Dado que este objeto puede haber sido modificado en el servidor web la semana pasada, la caché realiza una verificación actualizada emitiendo un GET condicional.
5. Este GET condicional le dice al servidor que envíe el objeto solo si el objeto ha sido modificado desde la fecha especificada.

HTTP / 2.

estandarizado en 2015, fue la primera versión nueva de HTTP desde HTTP / 1.1

objetivos principales: **reducir la latencia** percibida al permitir la multiplexación de solicitudes y respuestas en un único Conexión TCP, proporciona **priorización de solicitudes** y envío al servidor, y proporciona una **compresión eficiente de los campos de encabezado** HTTP

HTTP / 2 cambia el formato y el transporte de los datos entre el cliente y el servidor.

Utiliza conexiones TCP persistentes. Al tener solo una conexión TCP por página web, se reduce el número de sockets en el servidor y cada página web transportada obtiene una parte justa del ancho de banda de la red.

Bloqueo de cabecera de línea (HOL): Retraso por línea persistente, debido a esto, el **control de la congestión de TCP** proporciona a los navegadores un incentivo involuntario para utilizar múltiples conexiones TCP paralelas en lugar de una única conexión persistente

objetivos principales de HTTP / 2 es **deshacerse** de (o al menos reducir el número de) **conexiones TCP paralelas** para transportar una sola página web, por esto, HTTP / 2 requiere mecanismos cuidadosamente diseñados para evitar el bloqueo HOL.

Encuadre HTTP / 2 - HTTP/2 Framing

La solución HTTP / 2 para el bloqueo HOL es dividir cada mensaje en pequeños marcos e intercalar los mensajes de solicitud y respuesta en la misma conexión TCP.

- El entramado se realiza mediante la subcapa de entramado del protocolo HTTP / 2. Cuando un servidor desea enviar una respuesta HTTP, la respuesta es procesada por la subcapa de encuadre, donde se divide en marcos. Las tramas de la respuesta son luego intercaladas por la subcapa de tramas en el servidor con las tramas de otras respuestas y enviadas a través de la única conexión TCP persistente.
- A medida que los marcos llegan al cliente, primero se vuelven a ensamblar en los mensajes de respuesta originales en la subcapa de encuadre y luego el navegador los procesa como de costumbre.

Priorización de mensajes de respuesta y envío al servidor.

Permite a los desarrolladores **personalizar la prioridad** relativa de las solicitudes para **optimizar mejor el rendimiento** de la aplicación.

puede priorizar las respuestas que solicita asignando un peso entre 1 y 256 a cada mensaje. El número más alto indica una prioridad más alta.

el **cliente también establece la dependencia** de cada mensaje de otros mensajes especificando el ID del mensaje del que depende.

Otra característica de HTTP / 2 es la capacidad de un servidor para **enviar múltiples respuestas para una sola solicitud de cliente** (el servidor puede empujar objetos adicionales al cliente, sin que el cliente tenga que solicitar cada uno.). Esto es posible ya que la página base HTML indica los objetos que serán necesarios para representar completamente la página web.

QUIC tiene varias características que son deseables para HTTP, como multiplexación de mensajes (**entrelazado**), control de flujo por flujo y establecimiento de conexión de baja latencia.

HTTP / 3.

es un nuevo protocolo de "transporte" que se implementa en la capa de aplicación sobre el protocolo UDP básico.

aún no se ha estandarizado por completo.

2.5 Distribución de archivos de igual a igual (Peer-to-Peer). 56-72.

En cambio, pares (pairs) de **hosts** conectados intermitentemente, llamados pares, se comunican directamente entre sí. **Los pares (peers)** no son propiedad de un proveedor de servicios, sino que son PC, portátiles y dispositivos inteligentes controlados por los usuarios.

P2P: **distribuir** un archivo grande desde un solo **servidor a una gran cantidad de hosts** (llamados pares).

En la distribución de archivos P2P, **cada par puede redistribuir cualquier parte del archivo** que haya recibido a otros pares, ayudando así al servidor en el proceso de distribución.

El protocolo de distribución de archivos **P2P más popular** es **BitTorrent** por Bram Cohen**.**

Escalabilidad de arquitecturas P2P.

El servidor y los pares están conectados a Internet con enlaces de acceso.

Tiempo de distribución: es el momento se necesita para obtener una copia del archivo a todos N Peers.

Cliente - servidor:

- observaciones: El servidor debe transmitir una copia del archivo a cada uno de los N peers (El servidor debe transmitir NF bits).
- el tiempo de distribución aumenta linealmente con el número de pares.

arquitectura P2P:

Donde cada par puede ayudar al servidor a distribuir el archivo. cuando un par (peer) recibe algunos datos de archivo, puede usar su propia capacidad de carga para redistribuir los datos a otros pares.

Calcular el tiempo de distribución para la arquitectura P2P es algo más complicado, ya que el **tiempo de distribución** depende de cómo cada par distribuye partes del archivo a los otros pares.

Al comienzo de la distribución, solo el servidor tiene el archivo. Para obtener este archivo en la comunidad de pares, el servidor debe enviar cada bit del archivo al menos una vez en su enlace de acceso.

la capacidad de carga total del sistema en su conjunto es igual a la tasa de carga del servidor más las tasas de carga de cada uno de los compañeros, es decir, utotal = us + u1 + g+ uN.

BitTorrent.

Protocolo P2P popular para la distribución de archivos

En la jerga de BitTorrent, la colección de todos los pares que participan en la distribución de un archivo en particular se denomina Torrent.

Peers in a torrent download equal-size chunks of the file from one another, with a typical chunk size of 256 KBytes.

Cuando un peer se une por primera vez a un torrent, no tiene fragmentos.

Una vez que un par ha **adquirido el archivo completo**, **puede (egoístamente) dejar el torrent, o** (altruistamente) permanecer en el torrent **y continuar subiendo fragmentos** a otros pares.

Cada torrente tiene un nodo de infraestructura llamado rastreador. Cuando un par se une a un torrent, se registra en el **rastreador** y periódicamente informa al rastreador que todavía está en el torrent.

el **rastreador (tracker)** realiza un seguimiento de los pares que participan en el torrent.

técnica **llamada más raro primero (**rarest first**)**: La idea es determinar, de entre los trozos que no tiene, los trozos que son más raros entre sus vecinos (es decir, los trozos que tienen menos copias repetidas entre sus vecinos) y luego solicitar esos trozos más raros primero.

otra aplicación de P2P, a saber, **Distributed Hast Table (DHT)**. Una tabla hash distribuida es una base de datos simple, con los registros de la base de datos distribuidos entre los pares en un sistema P2P.

2.6 Video Streaming and Content Distribution Networks.

La transmisión de video, incluidos Netflix, YouTube y Amazon Prime, representa aproximadamente el 80% del tráfico de Internet en 2020.

implementan utilizando **protocolos a nivel de aplicación y servidores** que funcionan de alguna manera como un **caché**.

Una característica importante del video **es que se puede comprimir**, por lo que se compensa la calidad del video con la tasa de bits. Los algoritmos de compresión disponibles en la actualidad pueden comprimir un video a prácticamente cualquier tasa de bits deseada.

cuanto mayor sea la tasa de bits, mejor será la calidad de la imagen y mejor será la experiencia de visualización general del usuario.

El video de Internet comprimido generalmente varía desde 100 kbps para video de baja calidad hasta más de 4 Mbps para transmisión de películas de alta definición.

Transmisión HTTP y DASH.

- En la **transmisión HTTP**, el video simplemente **se almacena en un servidor HTTP como un archivo normal** con una URL específica.
- el servidor envía el archivo de video, dentro de un mensaje de respuesta HTTP, tan rápido como lo permitan los protocolos de red subyacentes y las condiciones del tráfico.
- En el lado del **cliente**, los bytes **se recopilan en un búfer de la aplicación** del cliente.
- Una vez que el número de bytes en este búfer excede un umbral predeterminado, la aplicación cliente comienza a reproducir, la aplicación de transmisión de video captura periódicamente fotogramas de video del búfer de la aplicación cliente, descomprime los fotogramas y los muestra en la pantalla del usuario.

transmisión de vídeo: muestra vídeo a medida que recibe y almacena en búfer los fotogramas correspondientes a las últimas partes del vídeo.

Transmisión dinámica adaptativa a través de HTTP (DASH): El **video se codifica en varias versiones diferentes**, y cada versión tiene una tasa de bits diferente y, en consecuencia, un nivel de calidad diferente.

El cliente selecciona diferentes fragmentos uno a la vez con mensajes de solicitud HTTP GET-

DASH también **permite que un cliente se adapte** al ancho de banda disponible si el ancho de banda de extremo a extremo disponible cambia durante la sesión.

HTTP también tiene un **archivo de manifiesto (**manifest file**)**, que **proporciona una URL para cada versión** junto con su tasa de bits.

DASH permite al cliente cambiar libremente entre diferentes niveles de calidad.

Redes de distribución de contenido.

Transmitir todo este tráfico a ubicaciones de todo el mundo mientras se proporciona una reproducción continua y una alta interactividad es claramente una tarea desafiante

El enfoque más sencillo para brindar servicio de transmisión de video es construir **un único centro de datos masivo**, almacenar todos y transmitir los videos directamente desde el centro de datos a todo el mundo.

tres problemas principales con este **enfoque**:

- si el cliente está lejos: el rendimiento de extremo a extremo también estará por debajo de la tasa de consumo, lo que resultará en molestos retrasos por congelación para el usuario.
- es probable que un video popular se envíe muchas veces a través de los mismos enlaces de comunicación (desperdicia el ancho de banda de la red).
- un solo centro de datos representa un solo punto de falla: si el centro de datos o sus enlaces a Internet se caen, no podría distribuir ninguna secuencia de video.

casi todas las principales empresas de transmisión de video utilizan **Redes de distribución de contenido (CDN):** gestiona servidores en múltiples ubicaciones distribuidas geográficamente, e intenta dirigir cada solicitud de usuario a una ubicación CDN que proporcionará la mejor experiencia de usuario.

CDN **privado**: propiedad del propio proveedor de contenido.

CDN **de terceros**: distribuye contenido en nombre de múltiples proveedores de contenido.

dos filosofías de ubicación de servidor diferentes:

- **Entra en lo profundo** (Enter Deep): implementación de clústeres de servidores en los ISP de acceso de todo el mundo.
  El objetivo es acercarse a los usuarios finales, mejorando así la demora y el rendimiento percibidos por el usuario al disminuir el número de enlaces y enrutadores entre el usuario final y el servidor CDN del que recibe el contenido.
- **Traer a casa** (Bring Home): llevar a los ISP a casa mediante la construcción de grandes grupos en un número menor (por ejemplo, decenas) de sitios.
  Colocan sus clústeres en Puntos de Intercambio de Internet (IXP).
  resulta en una menor sobrecarga de mantenimiento y administración, pero mayor demora y un menor rendimiento.

muchas CDN no envían videos a sus clústeres, sino que utilizan una estrategia de extracción simple: **si un cliente solicita un video de un clúster** que no almacena el video, entonces el **clúster recupera el video** y **almacena una copia localmente** mientras transmite el video al cliente al mismo tiempo. cuando el almacenamiento de **un clúster se llena, elimina los videos** que **no se solicitan con frecuencia**.

Operación CDN.
Funcionamiento de CDN:

1. host **recibe instruccion**es para recuperar un video específico
2. CDN debe **interceptar la solicitud**
3. **determinar un clúster** de servidor CDN adecuado
4. **redirigir la solicitud** del cliente **a un servidor** en ese clúster

Estrategias de selección de conglomerados.

Mecanismo para **dirigir dinámicamente a los clientes** a un clúster de servidores o un centro de datos dentro de la CDN.

- estrategia simple es la de asignar al cliente al clúster que está geográficamente más cercano.
- Para determinar el mejor clúster para un cliente en función de la Actual condiciones del tráfico, las CDN pueden r**ealizar operaciones periódicas mediciones en tiempo real de retraso y pérdida** de rendimiento entre sus clústeres y clientes.

Estudios de caso: Netflix y YouTube.

Netflix

tiene dos componentes principales: la **nube de Amazo**n y **su propia infraestructura de CDN** privada.

Netflix tiene un sitio web que maneja numerosas funciones que se ejecutan completamente en servidores de Amazon en la nube de Amazon.

Amazon maneja las siguientes funciones:

- Ingestión de contenido: ingerir y procesar la película.
- Procesamiento de contenido: crean muchos formatos diferentes para cada película
- Subiendo versiones a su CD: os hosts en la nube de Amazon cargan las versiones en su CDN.

Netflix distribuye enviando los videos a sus servidores CDN durante las horas de menor actividad.

1. Cuando un usuario selecciona una película para reproducir, el software de Netflix, que se ejecuta en la nube de Amazon, primero determina cuál de sus servidores CDN tiene copias de la película.
2. software determina entonces el "mejor" servidor.
3. Netflix envía al cliente la dirección IP del servidor específico, así como un archivo de manifiesto, que tiene las URL de las diferentes versiones de la película solicitada.
4. El cliente y ese servidor CDN interactúan directamente utilizando una versión propietaria de DASH
5. Netflix CDN utiliza el almacenamiento en caché de inserción en lugar de extracción en caché (el contenido se inserta en los servidores en horarios programados fuera de las horas pico).

YouTube

Al igual que Netflix, Google **usa su propia CDN** privada para distribuir videos de YouTube y ha **instalado clústeres de servidores en mucho**s cientos de **ubicaciones** diferentes de IXP e ISP

Google utiliza el **almacenamiento en caché de extracción**.

YouTube emplea la **transmisión HTTP**, a menudo haciendo que una pequeña cantidad de versiones diferentes estén disponibles para un video.

YouTube **no emplea transmisión adaptativa** (como DASH), sino que requiere que el usuario **seleccione manualmente una versión**.

Para **ahorrar ancho de banda** YouTube usa la **solicitud de rango de bytes HTTP** para limitar el flujo de datos transmitidos después de que se precargue una cantidad de video de destino.

YouTube **procesa cada video** que recibe, lo **convierte a un formato de video** de YouTube y crea **múltiples versiones a diferentes** velocidades de bits. Este procesamiento se lleva a cabo íntegramente en los centros de datos de Google.