Alberto Rodríguez Santana

T2. – Servicios y Protocolos de Aplicación en Internet

Protocolos TCP/IP

Los niveles más bajos (enlace y físico) no están implementados porque se diseñó para no depender de una red física concreta.

Capa de Red

La base de los protocolos es el nivel de red (IP – Internet Protocol). IP es un protocolo de conmutación de paquetes muy sencillo de tipo datagrama. Versiones IPv4 – Ipv6

Capa de Transporte

Implementa protocolos entre el nodo origen y el destino de la información (extremo a extremo). Se definen dos protocolos:

TCP (Transmission Control Protocol):

- Protocolo orientado a conexión.
- Control de errores.
- Se encarga del control de flujo.
- Fragmentado/reensamblado de flujo (secuenciamient)

UDP (User Datagram Protocol):

- Protocolo sin conexión.
- No realiza control de errores.
- No garantiza secuenciamiento
- Es muy rápido

Capa de Aplicación (Protocolos de Alto Nivel)

PING (ICMP), TELNET, SMTP, FTP, HTTP, RPC (basados en TCP) y SNMP, BOOTP, DNS, RPC, NFS (basados en UDP).

TCP y UDP al ser usuarios del protocolo IP no garantizan:

- Retardo Acotado.
- Fluctuaciones acotadas.
- Mínimo throughput
- Seguridad

Arquitectura Cliente – Servidor

Es una forma específica de diseño de aplicaciones. El cliente es la computadora que se encarga de efectuar una petición o solicitar un servicio a un servidor, que se encarga de evaluar la petición y decide aceptarla o rechazarla. Una vez aceptada la petición, el servidor envía la información al cliente.

Cliente y Servidor no tienen que porque estar en diferentes computadoras, pueden ser programas diferentes que se ejecutan en la misma computadora.

Se recomienda para redes que requieran un alto grado de fiabilidad.

VENTAJAS	DESVENTAJAS
- Recursos Centralizados: el servidor es el centro de la red y puede administrar los recursos comunes a todos los usuarios. **- Seguridad Mejorada** **- Administración al nivel del Servidor** **- Red Escalable:** es posible quitar o agregar clientes sin afectar el funcionamiento de la red.	**-Costo Elevado** **-Servidor es el eslabón débil:** debido a que toda la red está construida en torno a él, el servidor es el único eslabón débil. Es altamente tolerable a los fallos (RAID).

Servidor:

- Siempre en Funcionamiento
- IP Permanente y pública
- Agrupados en “granjas”.

Clientes:

- Funcionamiento Intermitente
- Pueden tener IP Dinámica y privada
- Se comunican con el servidor
- No se comunican entre sí

Procesos Cliente y Servidor

Proceso Cliente inicia la comunicación.

Proceso Servidor espera a ser contactado.

Proceso envía/recibe mensaje desde su **socket**.

Para recibir mensajes un proceso debe tener un **identificador (IP + Puerto).**

SOCKET: puerta de acceso entre la aplicación y los servicios de transporte. Cada tipo de socket define una serie de propiedades en función de las comunicaciones en las que está implicado. Fiabilidad de transmisión, conservación de orden de datos, no duplicación de datos, comunicación en modo conectado, conservación de los límites de mensajes, envío m.

Retardo en Cola (Servidor y Router)

$$R = \frac{\lambda \cdot (T_s)^2}{1 - \lambda \cdot T_s}$$

donde Ts (distribución exponencial) es el tiempo de servicio y
λ (Poisson) la ratio de llegada de solicitudes.

¿Qué define un protocolo?

- Tipo de Servicio: Orientado a conexión o no.
- Tipo de Mensaje: request, response.
- La Sintaxis: definición y estructura de "campos" en el mensaje
- La Semántica: significado de los campos.
- Las Reglas: cuándo los procesos envían/responden mensajes.

Tipos de Protocolos

Protocolos de dominio **público** – **propietarios**

Protocolos **In-band** – **out-of-band**

Protocolos **stateless – state-full (sin estado/con estado)**

-Stateless trata cada petición como una transacción independiente que no tiene relación con cualquier solicitud anterior.

-State-Full requiere el mantenimiento del estado interno en el servidor.

Protocolos **persistentes – no persistentes (1 conexión TCP vs multiples conexiones)**

Tendencia -> Hacer los protocolos flexibles con una **cabecera fija** y una **serie de "trozos".**

Características

Perdida de datos (Errores) -> tolerancia de perdida de datos.

Requisitos temporales -> Algunas apps requieren retardo acotado (delay) para ser efectiva

Ancho de banda (throughput) -> Algunas apps requieren envío a una tasa determinada

Seguridad -> Encriptacion, autenticación…

---

2. Servicio de nombres de dominio (DNS)

Es un esquema jerárquico que permite asignar nombres de dominio (cadena de hasta 255 caracteres formada por etiquetas separadas por un punto) a grandes conjuntos de máquinas y direcciones IP-> www.ugr.es ---- 150.214.204.25.

Utiliza una base de datos distribuida para implementarlos. Se utiliza mecanismo Cliente/Servidor -> (*resolvers / servidores de nombres*).

Cada Dominio es un índice en la BBDD.

El servicio DNS se puede situar en TCP/IP como protocolo de aplicación tanto sobre UDP como TCP en el puerto 53 de la capa de transporte.

¿Cómo funciona?

El DNS opera mediante **tres componentes** principales:

Los clientes DNS -> programa que hace petición de resolución de nombres (navegador web).

Los Servidores DNS -> servidores que contestan las consultas realizadas por los Clientes DNS. Según las características de la zona, los servidores se pueden clasificar en:

- **Primarios**: Almacena información de su zona en una base de datos local
- **Secundarios**: obtienen los datos de su zona desde otro servidor (transf. de zona)
- **Maestros**: son los que transfieren las zonas a los servidores secundarios
- **Locales**: no tienes autoridad sobre ningún dominio. Se limitan a contactar con otros servidores para resolver las peticiones de los clientes DNS.

Zonas de Autoridad -> almacena datos en los servidores DNS de los dominios.

Proceso de Resolución de Nombres de Dominio

Dos partes: Cliente (**Resolver**) – Servidor (**DNS**)

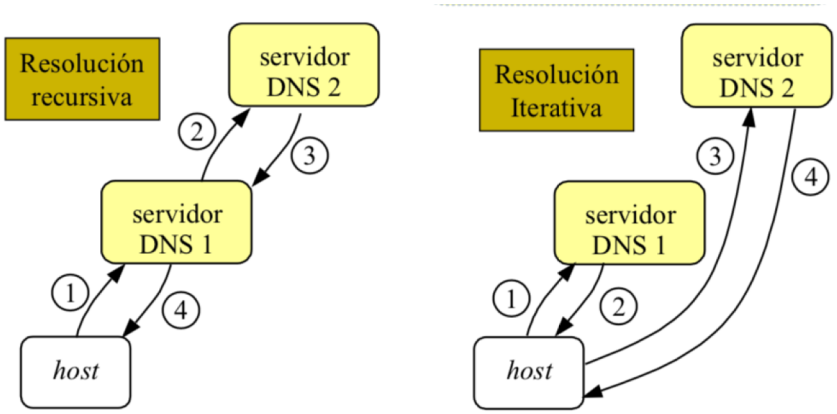
Funciones del cliente:

- Interrogar al servidor DNS
- Interpretar las respuestas (RR registro de recurso o errores)
- Devolver información al programa que realiza la petición

Los clientes pueden formular **dos tipos de preguntas**:

Recursiva: Obligar al servidor DNS a que responda aunque tenga que consultar a otros servidores. (+ frecuente).

Iterativa: el servidor responde solo si tiene la información, si no, remite la dirección de otro servidor capaz de resolver la pregunta. De esta forma el cliente tiene mayor control sobre el proceso de búsqueda. (- frecuente).



Gestión de la Base de Datos DNS

Cada zona debe tener al menos 1 servidor de autoridad (primario tiene copia master de DB y secundario obtiene la DB por transferencia.

Cuando un cliente solicita a través de un resolver una resolución de nombre puede ocurrir:

- Respuesta **CON** autoridad: el servidor tiene autoridad sobre la zona en la que se encuentra el nombre solicitado y devuelve la dirección IP.
- Respuesta **SIN** autoridad: el servidor no tiene autoridad en la zona que se encuentra el nombre pero si la tiene en caché.
- No conoce la respuesta: el servidor preguntara a otros servidores de forma recursiva o iterativa.

El **formato de los registros** es:

[Nombre_dominio] [TTL] [Clase] Tipo Dato_Registro(Valor)

TTL -> Tiempo de vida para almacenarse

Clase -> actualmente solo se utiliza IN, para información de internet.

Formato de los mensajes DNS:

Cabecera.
Consulta (o consultas).
(En la respuesta) RR de respuesta.
(En la respuesta) RR que identifican servidores con autorización.
(En la respuesta) RR con información adicional.

3. La Navegación Web

La **WWW (World Wide Web)** es la aplicación más importante en internet. Es un sistema de distribución de información basado en hipertexto o hipermedios.

Con un navegador web (cliente web) podemos acceder a la www y "navegar" por ella a través de los enlaces. Estos son sofisticadas aplicaciones que ser encargar de recoger y mostrar la información de los servidores web.

Servidor Web

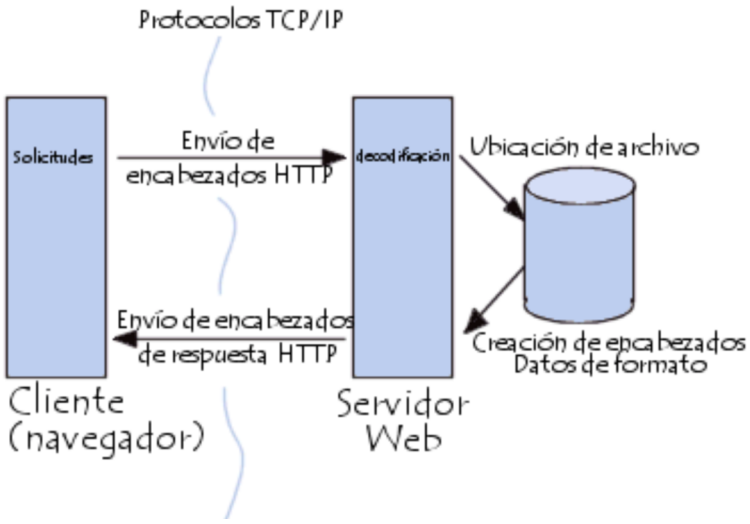
Es un ordenador que provee los datos solicitados por parte de los navegadores de otras computadoras. Los servidores almacenan información que entregan a través del protocolo http en formato html a petición de los clientes. Un servidor web espera conexiones en el **puerto 80 TCP.** (Servidor http Apache + usado del mundo)

Pasos que sigue:

1. Acepta una conexión TCP de un cliente (navegador).
2. Obtiene el nombre del archivo solicitado por el cliente
3. Obtiene el archivo del disco
4. Devuelve el archivo al cliente
5. Libera la conexión TCP

Tiempos de respuesta ideales de las paginas web:

- 0,1 segundo -> tiempo de respuesta ideal.
- 1 segundo -> Tiempo de respuesta más alto que es aceptable. Tiempos superiores a 1s interrumpen la experiencia del usuario.
- 10 segundos -> tiempo de respuesta inaceptable.



4. Protocolo HTTP

El protocolo de transferencia de hipertexto es un sencillo **protocolo Cliente/Servidor** que se basa en sencillas operaciones de solicitud/respuesta entre los clientes web y los servidores http. Un proceso servidor escucha en el puerto TCP 80. Basado en ASCII, por lo que puede transmitir cualquier tipo de documento.

Existen tres **funciones básicas**:

- **GET**: recoger un objeto
- **POST**: enviar información al servidor
- **HEAD:** solicitar características de un objeto
- **PUT:** actualizar información de un objeto del servidor.
- **DELETE:** eliminar documento especificado del servidor.
- **TRACE:** solicita al servidor mensaje de respuesta con fines de comprobación.
- **OPTIONS:** devuelve los métodos http que soporta el servidor para una url.
- **CONNECT:** se utiliza para saber si tiene acceso a un host.

Cada operación HTTP implica una conexión con el servidor, que es liberada al término de la misma. Es decir, en una operación se puede recoger un único objeto.

El servidor trata cada operación como una operación totalmente independiente del resto.

Solo existen **dos tipos de mensajes**, uno para realizar **peticiones** y otro para devolver la **respuesta**.

Mensaje de solicitud		Mensaje de respuesta
Comando HTTP + parámetros		Resultado de la solicitud
Cabeceras del requerimiento		Cabeceras de la respuesta
(línea en blanco)		(línea en blanco)
Información opcional		Información opcional

Etapas de conexión HTTP:

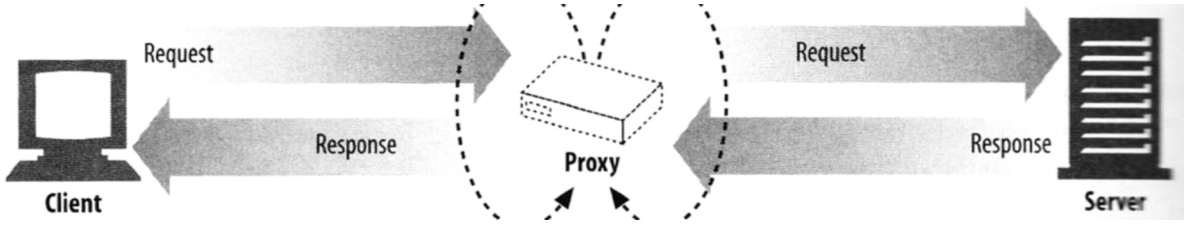
1. Un usuario **accede** a una **URL**.
2. El cliente web **decodifica** la dirección **URL**. Identifica **protocolo de acceso**, dirección **DNS** o **IP** del servidor, posible **puerto opcional** y el **objeto requerido** del servidor.
3. **Abre conexión TCP/IP** con el servidor llamando al **puerto TCP (80).**
4. Se realiza la **petición**. Para ello se envía el comando necesario, la dirección del objeto, la versión del protocolo http empleada (casi siempre 1.0).
5. El **servidor** devuelve la **respuesta** al cliente. Código estado MIME + información.
6. Se **libera la conexión**.

Este proceso se repite en cada acceso al servidor HTTP.

Códigos de estado del servidor HTTP

- 1xx: informativos.
- 2xx: operaciones realizadas correctamente.
- 3xx: informan de operaciones complementarias que se deben realizar para finalizar la operación.
- 4xx: errores del cliente.
- 5xx: errores del servidor.

Servidor Proxy -> servidor intermedio entre el cliente y el servidor que hace el papel de cliente y servidor.



Ventajas Servidor Proxy

- **Control:** solo el intermediario hace el trabajo real por lo que se puede limitar los derechos de los usuarios y dar permisos solo al proxy.
- **Ahorro:** solo uno de los usuarios (el proxy) debe estar equipado para hacer el trabjo real.
- **Velocidad:** el proxy puede hacer caché. Si varios usuarios piden lo mismo, guarda la respuesta de una petición para darla directamente cuando otro la pida.
- **Filtrado:** el proxy puede negarse a responder algunas peticiones si considera que están prohibidas.

Servidor Caché -> Servidor que almacena en memoria páginas que se han consultado mas recientemente.

- Ventajas Servidor Caché:
- Reduce tiempo de respuesta
- Optimiza el rendimiento de los servidores
- Reduce la transferencia de datos redundantes
- Reduce el culo de botella en los servidores web.

✓ Etapas del proceso de caché.
(ejemplo válido para una solicitud con comando GET)

1. Recepción : el servidor Caché lee el mensaje que le llega.

2. Análisis: el servidor Caché analiza el mensaje, extrae la URL y las cabeceras.

3. Búsqueda: El servidor caché verifica si tiene una copia disponible en local del documento solicitado.

4. Validación: El servidor verifica si la copia eS lo suficientemente actual, de no ser así solicita al servidor una nueva versión.

5. Respuesta: El servidor caché genera la respuesta y la envía al cliente web.

Http es un protocolo que no mantiene estado, es decir que trata todas las conexiones de forma anónima e independiente.

Las cookies son pequeños ficheros de texto que se intercambian los clientes y servidores http para **solucionar** una de las principales deficiencias del protocolo, la **falta de información de estado entre dos transacciones.**

Ejemplo de una cookie

EGSOFT_ID
191.46.211.13-655193640.29148285
www.rollingstone.com/
0
2867435528
30124157

---

4. Correo Electrónico

Es un servicio en red que permite a los usuarios enviar y recibir mensajes y archivos mediante sistemas de comunicación electrónicos.

Una dirección de correo electrónico (email) es un conjunto de palabras que identifican a una persona que puede enviar y recibir correo.

El correo se entrega usando una arquitectura cliente servidor. Un mensaje de correo se crea usando un **programa de correo cliente**, y se necesita un destinatario, asunto y el propio mensaje. El programa envía el mensaje a un servidor y el servidor lo redirige al servidor de correo destinatario y allí se le suministra al cliente de correo del destinatario.

Subsistemas que forman la arquitectura del sistema de correo:

- **Agentes de Transferencia (MTA):** mueven los mensajes de origen a destino.
- **Agentes de Usuario (MDA):** permiten leer y enviar correo. Programas locales que proporcionan una interfaz grafica.

Agentes de **transferencia** se clasifican en:

- Agentes de **Distribución**: utilizando para ello el protocolo **SMTP**.
- Agentes de **Entrega Final:** que permiten al usuario gestionar su correo a través de una maquina remota utilizando los protocolos **POP3** e **IMAP4**.

Agente de Distribución (SMTP)

SMTP -> es un sencillo protocolo cliente/servidor. El objetivo principal es transmitir correo entre servidores de correo mediante una conexión punto a punto. Establecida una comunicación TCP entre la computadora que remite el correo, que opera como cliente y el **puerto 25** del servidor de correo de destino, el cliente permanece a la espera de recibir un mensaje del servidor que pueden ser:

- **Está preparado** para recibir correo: el cliente anuncia de quien viene el mensaje y a quien va. Si esta ese destinatario en el servidor, se envía el mensaje. Una vez intercambiado el correo se libera la conexión.
- **No está preparado:** el cliente libera la conexión y lo intenta después.

Problemas en SMTP:

- **No requiere autenticaión,** por lo que permite a cualquiera en internet enviar correo a cualquiera. Esto hace posible el correo **basura/spam.**

Agente de Transferencia de Entrega Final de Usuario

El correo entrante se puede obtener básicamente a través de los siguientes protocolos:

POP3 -> el objetivo es obtener el correo del **buzón remoto** y almacenarlo en la máquina local para su lectura posterior. Inicia cuando el usuario arranca el lector de correo y establece una conexión TCP con el servidor (puerto 110). Una vez establecida la conexión el protocolo pasa por tres estados:

- **Autorización**
- **Transacciones**
- **Actualización**

Además POP3 bloquea las bandejas de entrada durante el acceso lo que hace imposible que dos usuarios accedan a la misma bandeja simultáneamente.

IMAP -> no copia el correo en la maquina personal del usuario ya que puede tener varias máquinas para consultar el correo. Observa si sus correos han sido leídos con anterioridad. Supone que todo el correo electrónico permanecerá en el servidor de manera indefinida. Se comunica con el servidor IMAP por TCP puerto 220.

Diferencias importantes:

POP3	IMAP4
Los clientes se conectan **brevemente al servidor**, solamente el tiempo que les tome descargar el nuevo mensaje.	Los clientes permanecen conectados el tiempo que su interfaz esté activa y descargan los mensajes bajo demanda.
Supone que el cliente conectado es el **único dueño** de una cuenta de correo.	Permite **accesos simultáneos a varios clientes** y proporciona **mecanismos** para que se **detecten** los **cambios** realizados en un **buzón** o en los **mensajes.**
	Los clientes de IMAP pueden **crear, renombrar** o **eliminar** correo del servidor y mover mensajes entre cuentas de correo. Permite **buscar** correos lo que **evita** que los clientes **descarguen todos los mensajes** de su buzón de correo.

Características	POP3	IMAP4
RFC	RFC 1939	RFC 2060
Puerto TCP utilizado	110	143
Dónde se almacena el correo electrónico	PC del usuario	Servidor
Tiempo de conexión requerido	Poco	Mucho
Uso de recursos del servidor	Mínimo	Amplio
Quién mantiene los buzones	Usuario	ISP
Bueno para usuarios móviles	No	Si
Descargas parciales de mensajes	No	Si
Sencillo de implementar Soporte amplio	Si	No

Agentes de Usuario

Es una aplicación informática que funciona como cliente en un protocolo de red. Esta diseñado para recoger y enviar correo. Existen diferentes tipos:

- **Aplicación de Cliente** de correo electrónico (MUA).
- **Interfaz Web:** es como porque permite ver y almacenar mensajes desde cualquier sitio.

POP3 - port 110
IMAP - port 143
SMTP - port 25
HTTP - port 80
Secure SMTP (SSMTP) - port 465
Secure IMAP (IMAP4-SSL) - port 585
IMAP4 over SSL (IMAPS) - port 993
Secure POP3 (SSL-POP) - port 995

5. Seguridad y Protocolos Seguros

Primitivas de Seguridad

- **Confidencialidad:** Solo accede a la información quien debe hacerlo.
- **Responsabilidad:**
  - **Autenticación:** los agentes son quien dicen ser.
  - **No Repudio:** no se puede negar el autor de una determinada acción.
  - **Control de accesos:** garantía de identidad para el acceso.
- **Integridad:** detección de alteraciones de la información.
- **Disponibilidad:** mantener disponibles los servicios.

Mecanismos de Seguridad -> Detección, Prevención y Recuperación.

Criptología: ciencia que trata los problemas teóricos relacionados con la seguridad en el intercambio en clave de mensajes entre un emisor/receptor.

- **Criptografía:** ocupada del cifrado de mensajes en clave y diseño de criptosistemas. Mecanismos más utilizado para proporcionar seguridad en las redes.
- **Criptoanálisis:** descifrar los mensajes en clave.

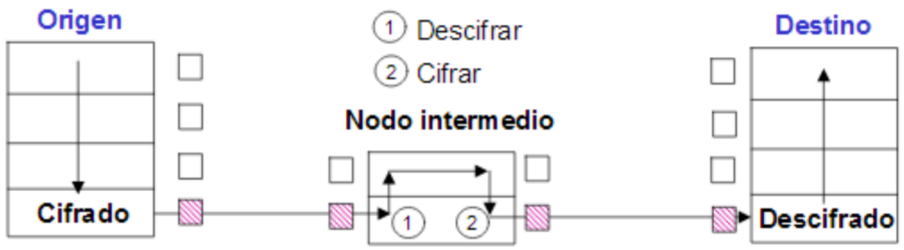
Criptosistema:

- **Simétricos (Clave privada o secreta)**
- **Asimétricos (Clave Pública)**

Métodos Básicos de Cifrado.

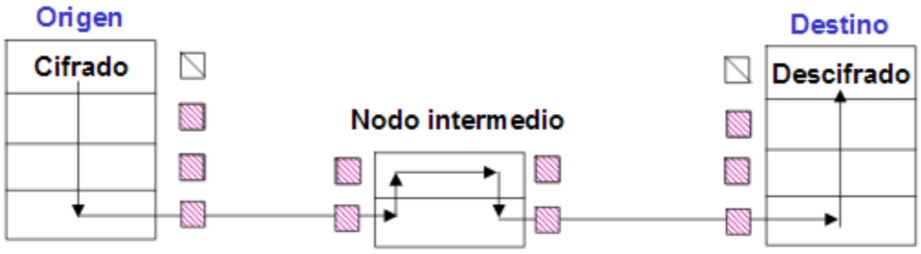
Cifrado de enlace:

- De capa 2 de OSI
- Cifra todo el mensaje, incluidas las cabeceras de niveles superiores
- Requiere nodos intermedios con capacidades de cifrado/descifrado
- La información está protegida entre cada par de nodos consecutivos (distintas claves para cada par)
- Es necesario descifrarla, aunque sea parcialmente, para procesos
- de encaminamiento, control de errores...



Cifrado extremo a extremo:

- De capa 7 de OSI
- Sólo se cifran los datos, las cabeceras se añaden y se transmiten sin cifrar
- El cifrado de datos se mantiene desde origen hasta destino



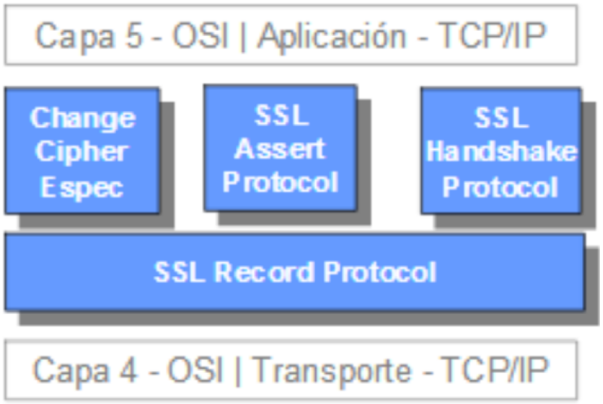
Protocolos Seguros – Nivel de Aplicación

SSH (Secure Shell) → Es un protocolo de nivel de aplicación para crear conexiones seguras entre dos sistemas de redes no seguras. Alternativa a telnet, ftp…

1. Se crea una capa de transporte segura para que el cliente sepa que esta conectado con el servidor correcto y se cifra la comunicación entre el cliente y el servidor por medio de un código simétrico.
2. Con la conexión segura al servidor, el cliente se autentifica ante el servidor.
3. Con el cliente autentificado al servidor, se puede usar diferentes servicios con seguridad, como una sesión Shell, tuneles TCP/IP…

Protocolos Seguros – Nivel de Transporte

SSL (Secure Socket Layer) → su objetivo es la realización de conexiones seguras a los servidores independientemente del SO de los extremos del canal. Compuesto por dos capas SSL Record Protocol y la segunda formada por 3 pro. Problemas -> solo trabaja sobre TCP, no repudio e ineficiencia debido al handshake inicial.



SSL Funcionamiento

- El cliente al hacer la conexión **informa** sobre los **sistemas criptográficos** que tiene **disponibles** y el **servidor responde** con los sistemas que soporta, identificador de la conexión y una clave certificada.
- El cliente deberá **elegir un sistema criptográfico**. Verifica la clave publica del servidor y genera una clave cifrada con la clave del servidor. Este es uno de los puntos mas importantes de SSL porque si alguien pudiese descifrar la información, solo conseguiría romper esa conexión y necesitaría una clave diferente para la conexión posterior.
- Una vez Finalizado este proceso, los protocolos toman el control del nivel de aplicación.

Protocolos Seguros – Nivel de Transporte

IPSec (IP Security) → Suministra seguridad a nivel de red proporcionando seguridad para ip y los protocolos de capas superiores.

Modos de Funcionamiento

- **Modo de Transporte** (IP Seguro)
- **Modo Túnel** (IP Seguro dentro de IP estándar)

VPNs (Virtual Private Networks)

¿Qé es una VPN?

Proporciona el medio para usar una infraestructura de red publica como un canal apropiado para comunicaciones privadas de datos (proporciona confianza). Constituye un túnel cifrado y encapsulado a través de internet. Utiliza protocolos de tunneling y conexiones temporales.

- **VPNs de Intranet** → Conectividad interna en una empresa
- **VPNs de Acceso Remoto** → Amplian la red interna a las oficinas remotas y trabajadores itinerantes.
- **VPNs de Extranets** → Amplian la red de las empresas e incluyen proveedores, empresas asociadas y clientes.

Para **implementar** una **VPN:**

- Diseñar una topología de red y firewalls
- Escoger un protocolo para los tuneles
- Diseñar una PKI (Public Key Infraestructure)

VENTAJAS	DESVENTAJAS
- Ahorro en Costes. **- No se compromete la seguridad de la red empresarial.** **- El cliente tiene acceso a todos los recursos ofrecidos en la LAN.** **- Acceso desde cualquier punto del mundo.**	**-No se garantiza la disponibilidad** (No internet -> No VPN) **- Fiabilidad menor que una línea dedicada.** **- Mayor complejidad en la configuración del cliente.** **- Se considera segura pero no hay que olvidar que la inf. Sigue viajando por internet.**

¿Qué es Tunneling?

Transmisión de paquetes de datos de un protocolo encapsulado en otro. Permiten crear redes privadas virtuales o VPNs.

- **PPTP**: Orientado a **usuario**, permite establecer un túnel de forma transparente al proveedor de internet.
- **L2TP:** orientado al **proveedor**, permite establecer un túnel de forma transparente al usuario (generalmente se usa junto con IPSec)

Cortafuegos (Firewall)

Combinación de **técnicas**, **políticas de seguridad** y **tecnologías** destinadas a proporcionar **seguridad** en la red, **controlando el tráfico** que circula entre dos o más redes. El nivel de protección que nos ofrece depende de las necesidades concretas.

Deben **combinarse** con otras medidas de seguridad → protocolos seguros.

Componentes:

- **Filtros:** permiten bloquear selectivamente determinados paquetes. Normalmente son routers o computadoras con capacidad de filtrado.
- **Nodos Bastión:** computadoras altamente seguras que sirven como punto de contacto entre la red local e internet.

Funciones

- **Controlar** permitiendo o denegando los accesos desde la red al exterior y desde el exterior a la red.
- **Filtrar** los paquetes que circulan
- **Monitorizar el tráfico.**
- **Almacenar** los paquetes para analizarlos en caso de problemas.

1. **Filtrado de Paquetes:** se controla selectivamente el trafico de la red definiendo una serie de reglas. **Muy efectivos** como primera barrera si se combina con otras medidas de seguridad.

2. **Servicios Delegados (proxy service):** son aplicaciones especializadas que funcionan en cortafuegos y hacen de intermediario entre los servidores y los clientes.

Mecanismos de Seguridad

Cifrado de Clave Pública → se basa en el uso de **dos claves diferentes.** Una clave puede descifrar lo que la otra ha cifrado. Una de las claves es llamada clave **privada** y es usada por el propietario para **cifrar** los mensajes y la otra clave, clave **pública**, es usada para **descifrar** el mensaje.

Cifrado de Clave Privada → utiliza **una clave** para el cifrado y descifrado del mensaje. Esta clave se debe intercambiar entre los equipos por medio de un **canal seguro**. Ambos extremos deben **tener la misma clave.**

La clave **privada** el propietario tiene que mantenerla en **secreto** mientras que la **pública** es **difundida**. Ambas claves se **generan siempre a la vez** y están **ligadas** entre ellas.

Firma Digital → Proceso que hace posible garantizar la autenticidad del remitente y verificar la integridad del mensaje recibido.

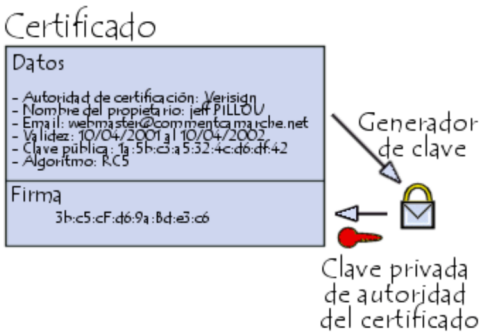
Certificado Digital

Son pequeños archivos que divididos en dos partes:

- La parte que contiene la **información**
- La parte que contiene la **firma** (por **entidad de certificación** o **local**)

La información que contiene el certificado (estandarizada por la norma X.509):

- Versión de X.509
- Número de serie
- Algoritmo de cifrado
- Nombre de la entidad de certificación
- Fecha entrada en vigencia
- Fecha validez
- Clave publica
- Firma del emisor del certificado



Un certificado permite asociar una clave pública con una entidad para garantizar su validez.

6.Aplicaciones Multimedia

El termino multimedia hace referencia al uso combinado de diferentes medios de comunicación: texto, imagen, sonido, video, animación…

Tipos de Aplicaciones

- **Flujo de Audio/Video (streaming) almacenado ->** Youtube
- **Flujo de Audio/Video en vivo ->** emisoras de radio
- **Audio y Video Interactivo ->** Skype

Características fundamentales

- Elevado ancho de banda
- Tolerantes relativamente a la perdida de datos
- Exigen Delay acotado
- Exigen Jitter acotado
- Se pueden beneficiar del multicast

Delay → medida de tiempo en la que un paquete tarda en viajar desde un origne hasta un destino.

Jitter → Señal de ruido no deseada en una señal.