

2012

Protocolo HTTP



http://

Álvaro Primo Guijarro

Teoría de SRI

12/01/2012

Índice

World Wide Web	6
Funcionamiento de la Web	6
Estándares Web	6
¿Qué son los Estándares Web?	7
¿Para qué sirven?	8
Acceso Universal	8
Una Web con significado.....	9
Confianza en la Web.....	9
¿Cómo funcionan?	10
Ejemplos.....	11
HTML	11
XML.....	11
Características generales de un servicio Web.....	12
¿Qué son los Servicios Web?.....	12
¿Para qué sirven?	12
- Componentes y funcionamiento. ¿Cómo funcionan?	12
Nombres y direcciones (URIs y URLs).....	14
¿Qué es un URL?	15
¿Cuál es la diferencia entre URL y URI?	15
Páginas web.....	15
Características	15
Sitios Web.....	16
<i>Diferencia entre sitio web y página web</i>	17
Aplicación Web.....	17
Ejemplos de aplicaciones web.....	17
Protocolo HTTP.....	18
Funcionamiento básico.	18
Los mensajes HTTP	19
Un mensaje de petición.....	19
Un mensaje de respuesta.....	20

Protocolo HTTP

Métodos de petición	21
Las cabeceras del protocolo http	23
Connection (<i>conexión</i>)	23
Content-Language (<i>idioma del contenido</i>)	23
Content-Length (<i>longitud del contenido</i>)	24
Content-Location (<i>localización del contenido</i>)	24
Content-Type (<i>tipo de contenido</i>)	24
Date (<i>fecha</i>)	24
Expect (<i>espera</i>)	24
Expires (<i>expiración</i>)	24
From (<i>"desde"</i>)	24
If-Match (<i>"si cuadra"</i>)	25
If-Modified-Since (<i>"si se ha modificado desde"</i>)	25
If-None-Match (<i>"si no cuadra"</i>)	25
IP (<i>remote adress</i>)	25
Host (<i>servidor</i>)	25
Last-Modified (<i>última modificación</i>)	25
Location (<i>localización</i>)	25
Referer (<i>remitente</i>)	25
Request (<i>solicitud</i>)	26
Status Code (<i>código de estado</i>)	26
User-Agent (<i>agente de usuario</i>)	26
Errores	26
Los códigos de respuesta	27
Almacenamiento en cache	29
Tipos de cachés web	29
Control de los cachés web	29
- Redirecciones	30
Comprensión	30
Ventajas	30
Desventajas	31
Cookies	31
¿Por qué se han creado las cookies?	31
Autenticación	32

Protocolo HTTP

AUTENTICACIÓN BÁSICA	32
AUTENTICACIÓN MEDIANTE RESÚMENES O DIGEST	33
AUTENTICACIÓN DE WINDOWS INTEGRADA.....	34
AUTENTICACIÓN HTTPS.....	34
Conexiones persistentes	35
Conexión persistente del HTTP	35
Ventajas.....	35
Configuración de un servidor Web	36
- Instalación, configuración y uso.....	36
Instalar el Servidor Apache.	36
Como configurar el Servidor Apache	36
USO.....	37
Autenticación y control de acceso.	38
Registro y monitorización del servicio Web.....	39
Tipos MIME	39
Subtipos de Multiparte	40
Webdav	42
Visión general acerca del protocolo Webdav	42
Navegadores web.....	43
Funcionamiento de un navegador web	44
Parámetros de apariencia y uso.....	44
Barra de Título.....	44
Barra de Menús.....	45
Barra de Herramientas.....	45
Barra de Direcciones	46
Actual Página Web con Vínculos.....	46
Barra de Estado	47
Barra de Vínculos	47
Seguridad del protocolo HTTP.....	47
Protocolo HTTPS.....	47
Conexiones seguras: SSL , TLS.	49
Gestión de certificados y acceso seguro con HTTPS	50
Almacenamiento virtual de sitios web “HOSTS VIRTUALES”	51
Alojamiento virtual basado en IPS	52

Protocolo HTTP

Alojamiento virtual basado en nombres.....	52
Alojamiento virtual basado en puertos.....	53
Basado en puerto	53
Alojamientos híbridos	53
Servidor privado virtual hosting	54
Administrados de alojamiento.....	54
Unmetred hosting.....	54
El software de virtualización	54
Nube de servidor	54

World Wide Web

En informática, la **World Wide Web (WWW)** o **Red informática mundial** es un sistema de distribución de información basado en hipertexto o hipermedios enlazados y accesibles a través de Internet. Con un navegador web, un usuario visualiza sitios web compuestos de páginas web que pueden contener texto, imágenes, videos u otros contenidos multimedia, y navega a través de ellas usando hiperenlaces.



La Web fue creada alrededor de 1989 por el inglés Tim Berners-Lee y el belga Robert Cailliau mientras trabajaban en el CERN en Ginebra, Suiza, y publicado en 1992. Desde entonces, Berners-Lee ha jugado un papel activo guiando el desarrollo de estándares Web (como los lenguajes de marcado con los que se crean las páginas web), y en los últimos años ha abogado por su visión de una Web semántica.

Funcionamiento de la Web

El primer paso consiste en traducir la parte nombre del servidor de la URL en una dirección IP usando la base de datos distribuida de Internet conocida como DNS. Esta dirección IP es necesaria para contactar con el servidor web y poder enviarle paquetes de datos.

El siguiente paso es enviar una petición HTTP al servidor Web solicitando el recurso. En el caso de una página web típica, primero se solicita el texto HTML y luego es inmediatamente analizado por el navegador, el cual, después, hace peticiones adicionales para los gráficos y otros ficheros que formen parte de la página. Las estadísticas de popularidad de un sitio web normalmente están basadas en el número de páginas vistas o las peticiones de servidor asociadas, o peticiones de fichero, que tienen lugar.

Al recibir los ficheros solicitados desde el servidor web, el navegador renderiza la página tal y como se describe en el código HTML, el CSS y otros lenguajes web. Al final se incorporan las imágenes y otros recursos para producir la página que ve el usuario en su pantalla.

Estándares Web

Destacamos los siguientes estándares:

- el *Identificador de Recurso Uniforme (URI)*, que es un sistema universal para referenciar recursos en la Web, como páginas web,

Protocolo HTTP

- el *Protocolo de Transferencia de Hipertexto* (HTTP), que especifica cómo se comunican el navegador y el servidor entre ellos,
- el *Lenguaje de Marcado de Hipertexto* (HTML), usado para definir la estructura y contenido de documentos de hipertexto,
- el *Lenguaje de Marcado Extensible* (XML), usado para describir la estructura de los documentos de texto.

Berners Lee dirige desde 2007 el World Wide Web Consortium (W3C), el cual desarrolla y mantiene esos y otros estándares que permiten a los ordenadores de la Web almacenar y comunicar efectivamente diferentes formas de información.

¿Qué son los Estándares Web?

Un estándar es un conjunto de reglas normalizadas que describen los requisitos que deben ser cumplidos por un producto, proceso o servicio, con el objetivo de establecer un mecanismo base para permitir que distintos elementos hardware o software que lo utilicen, sean compatibles entre sí.



El W3C, organización independiente y neutral, desarrolla estándares relacionados con la Web también conocidos como Recomendaciones, que sirven como referencia para construir una Web accesible, interoperable y eficiente, en la que se puedan desarrollar aplicaciones cada vez más robustas.

En la creación de las Recomendaciones del W3C participan sus Miembros (más de 400 organizaciones, distribuidas a lo largo de todo el mundo y de diversos ámbitos: grandes empresas de hardware o software, centros investigadores, universidades, administraciones públicas, etc.), el Equipo del W3C, expertos invitados, y cualquier usuario de la Web que quiera mostrar su opinión. Todos ellos trabajan conjuntamente a través de un proceso basado en el consenso, la neutralidad y la transparencia de la información.

El resultado: más de 110 tecnologías desde 1996.

Protocolo HTTP

1996	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007
PNG	HTML 3.2	XML 1.0	CSS 1	XHTML 1.0	MathML 2.0	XML Signature	DOM 2 HTML	CC/0*	SMIL 2.0 (and ext.)	XSL-FO 1.0 (and)	XPath 2.0
PCSLabek	HTML 4.0	MathML 1.0	Namespaces	ATAG 1.0	Canonical XML	PyP 1.0	SVG 1.1	DOM 3 Validation	RSSH	WSA Core	XQuery 1.0
PCS Rating	PCS Rules	HTML 4.0	WebCGM	XML 1.0	XHTML Mod	XML Canonicalization	SVG Mobile	Infra (and)	SOAP MTOM	WSA SOAP Binding	XQuery Data Model
CSS 1		CSS 2	RDF (Old Version)	DOM 2 Core	Schema Primer	XHTML 1.0	XPTR Element	Namespaces 1.1	XOP*	Namespaces (and)	Query Formal Semantics
		PCSDSig	WCAG 1.0	DOM 2 Events	Schema Struct.	XPath Filter	XPTR Framework	XML 1.0 (3rd)	Char Model	Namespaces 1.1 (and)	XQuery Functions and Operators
		SMIL 1.0	Style Sheets PI	DOM 2 Style	Schema Types	Decrypt Transform	XPTR Syntax	XML 1.1	XKMS	XML 1.0 (4th)	XQuery Serialization
		DOM 1	MathML 1.0	DOM 2 Traversal	Rdfp	XML Encryption	SOAP Adjuncts	OWL Guide	XKMS Bindings	XML 1.1 (and)	XQueryX
			XPath 1.0	DOM 2 Views	XHTML 1.1	UAA 1.0	SOAP Framework	OWL Overview	QJ Framework	XHTML-Print	XSLT 2.0
			XSLT 1.0	XHTML Basic	XLink 1.0		SOAP Primer	OWL Reference	xml:id	XInclude (and)	WebCGM 2.0
			HTML 4.01		XML Base		SOAP Tests	OWL Semantics	SMIL 2.1	XSL 1.1	ITS
					SMIL 2.0		XForms 1.0	OWL Tests			SSSR
					SMIL Animation		XML Events	OWL Use Cases			SOAP Adjuncts (and)
					SVG 1.0		MathML 2.0	RDF Concepts			SOAP Framework (and)
					XSL 1.0		PNG (and)	RDF Primer			SOAP Primer (and)
					Infra			RDF Schema			SOAP Tests (and)
					WebCGM			RDF Semantics			VoiceXML 2.1
								RDF Test Cases			WSDL 2.0 Adjuncts
								RDF/XML			WSDL 2.0 Core
								Speech Recognition			WSDL 2.0 Primer
								VoiceXML 2.0			SAWSL
								DOM 3 Core			WSA Metadata
								DOM 3.1 & S			WSP 1.3 Attachment
								Speech Synthesis			HSP 1.3 Framework
								Schema Primer (and)			GRDDL
								Schema Struct. (and)			GRDDL Tests
								Schema Types (and)			
								WebArch			
								XInclude			

Date: 7-7-Sep-2007

Algunos de los estándares Web más conocidos y ampliamente utilizados son: HTML (HyperText Markup Language), para definir la estructura de los documentos; XML (eXtensible Markup Language), que sirve de base para un gran número de tecnologías; y CSS (Cascading Style Sheets), que permite asignar estilos para la representación de los documentos.

¿Para qué sirven?

La finalidad de los estándares es la creación de una Web universal, accesible, fácil de usar y en la que todo el mundo pueda confiar. Con estas tecnologías abiertas y de uso libre se pretende evitar la fragmentación de la Web y mejorar las infraestructuras para que se pueda evolucionar hacia una Web con la información mejor organizada.

Acceso Universal

El W3C se guía por los principios de accesibilidad, internacionalización, e independencia de dispositivo, entre otros. Esto facilita que el acceso a la Web sea posible desde cualquier lugar, en cualquier momento y utilizando cualquier dispositivo. No importa si se utiliza hardware, software, o una infraestructura de red específica. Además de las posibles restricciones técnicas, se tiene en cuenta la existencia de múltiples idiomas, las diversas localizaciones geográficas, y las diferencias culturales o tradiciones, así como las posibles limitaciones físicas, psíquicas o sensoriales de los usuarios.

La concienciación de que no todas las personas acceden a la Web de la misma forma, permite centrarse en determinados colectivos que tienen necesidades concretas, como pueden ser las personas de edad avanzada en el caso de limitaciones psíquicas, físicas o sensoriales.

El avance de las tecnologías inalámbricas, así como la gran variedad de dispositivos con acceso a la Web presentes en sectores como el de la telefonía móvil, en el de automoción (navegadores en los salpicaderos de automóviles), en los electrodomésticos (refrigeradores con pantallas táctiles) o en los televisores, fomenta la ubicuidad de la Web. Esto pone de manifiesto la necesidad de utilizar tecnologías y lenguajes unificados, libres y gratuitos, cuyo uso no esté limitado por patentes comerciales.

Una Web con significado

Tradicionalmente, se podría considerar la Web como un conjunto de documentos conexos entre sí a través de términos léxicos y sintácticos. Estos documentos están expresados en lenguaje natural y contienen contenido destinado a personas. Esto limita a las máquinas a la hora de procesar la información de forma eficiente, ya que no pueden evitar las ambigüedades del lenguaje natural. Gracias al nuevo enfoque del W3C, la Web evoluciona hacia lo que se denomina Web Semántica, una ampliación de la Web tradicional, que ofrece mecanismos para añadir significado a los recursos (documentos, imágenes, vídeos, etc.), de forma que cualquier máquina pueda interpretar los datos existentes en la Web de una forma similar a como lo hacen los humanos. La información no sólo está especificada como una serie de información textual o gráfica inconexa entre sí, sino que la Web se puede considerar como una gran base de datos organizada y estructurada teniendo en cuenta la naturaleza semántica de los elementos que la componen.

La Web Semántica fomenta una mejora en el rendimiento y eficiencia de la Web, lo que se transmite en una experiencia más satisfactoria para el usuario, el que obtendrá mayor precisión en sus búsquedas y operaciones, y podrá tener acceso a mayores cantidades de información específica y útil. De la misma forma, esta estructuración y tratamiento de los datos más preciso evitará las tareas frustrantes y difíciles, como es la búsqueda, obtención y mezcla de información desde distintas fuentes.

Confianza en la Web

La Web es un medio colaborativo, donde los usuarios interactúan creando contenidos (en wikis, blogs o foros), realizan transacciones (compras online, operaciones bancarias), o crean redes sociales (de amistad o laborales), donde se relacionan entre sí.

Estas actividades requieren que los usuarios confíen entre sí y han promovido el desarrollo de ciertas tecnologías para asegurar esta confianza: firmas digitales de documentos que fomentan la responsabilidad de las personas que se declaran autores de estos; encriptación de los datos para la confidencialidad; y mecanismos de

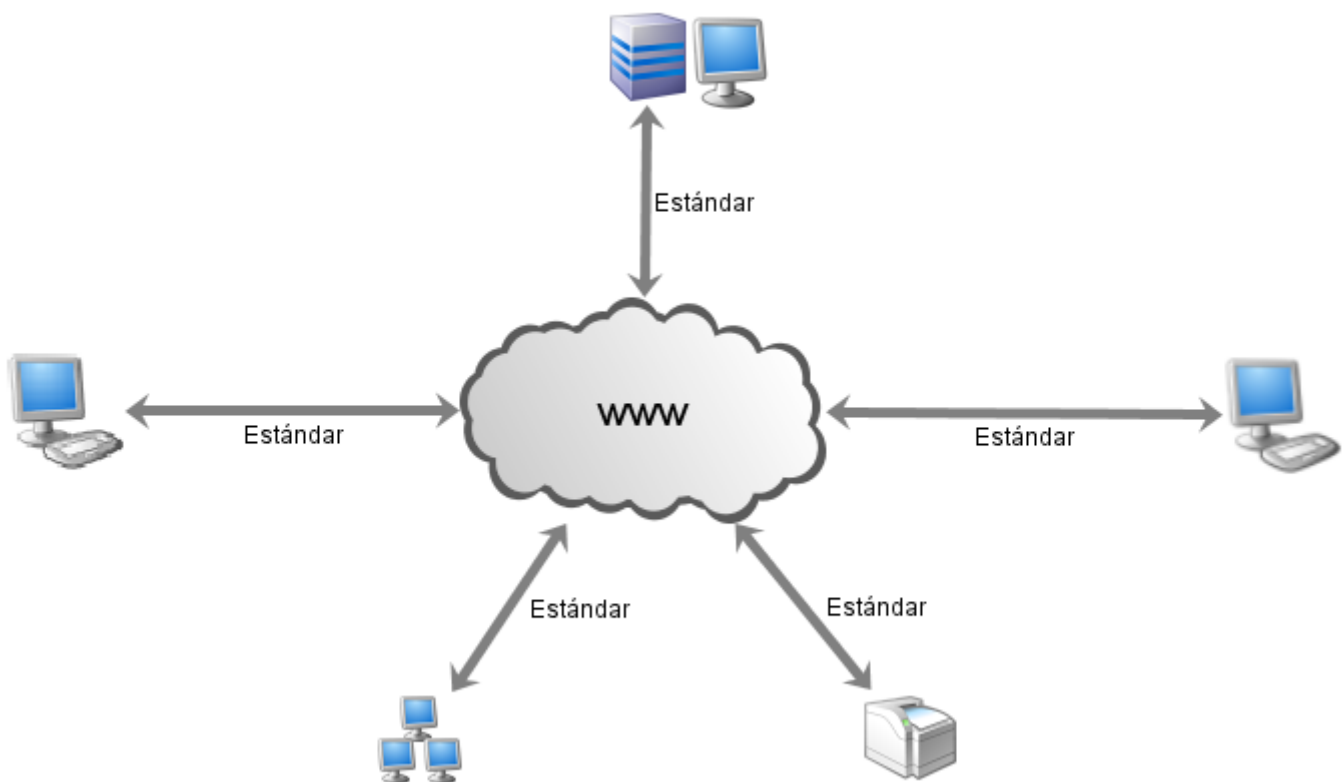
establecimiento y declaración de las políticas de privacidad de los datos de los sitios Web.

¿Cómo funcionan?

La creación de un estándar Web requiere un proceso controlado, que consta de varias etapas que aseguran la calidad de la especificación. Este proceso permite la intervención de todos los usuarios de las tecnologías, con el objetivo de que puedan aportar su conocimiento y opiniones para la mejora de los documentos.

Tras este proceso, elaborado por especialistas en la materia, se obtienen unos estándares de calidad, y al estar disponible para todo el mundo, las especificaciones se depuran exhaustivamente antes de ser consideradas como Recomendación.

Estos estándares, están sujetos a la Política de Patentes del W3C, lo que permite que sean utilizados libremente por toda la comunidad Web. Al utilizar las mismas tecnologías, las máquinas se entienden entre sí y cualquier usuario puede interactuar con el resto.



Para ayudar a los desarrolladores que deseen utilizar sus Recomendaciones, el W3C ofrece una serie de herramientas que permiten verificar si se hace una correcta aplicación de las especificaciones. Manuales de directivas o buenas prácticas de

tecnologías concretas, y los validadores sintácticos de los lenguajes, son ejemplos de estas ayudas.

Ejemplos

Durante la evolución de la Web, ha quedado patente la necesidad de disponer de estándares y existe un gran número de estos que han sentado las bases para el desarrollo de la Web y han fomentado el éxito de esta. Algunos ejemplos son: el lenguaje de etiquetado para hacer páginas Web, HTML; y XML, un lenguaje para crear estructuras de documentos.

HTML

El beneficio de la utilización de estándares se puede observar con la creación y evolución del lenguaje HTML, para la Web. En 1994, el W3C comenzó el proceso de estandarización del HTML para representar el contenido en la Web. La expansión y el número de documentos en la Web se ha visto incrementado en los últimos años de forma espectacular, hasta convertirse en una herramienta de uso cotidiano como hoy la conocemos. Esto se ha debido a la facilidad de creación de documentos y a que todos comparten el mismo lenguaje para la representación de la información. Al usar el mismo formato para el desarrollo se consigue que cualquier agente de usuario que interprete dicho lenguaje represente el documento de la Web de la misma forma. Al ser independiente de cualquier plataforma (y de cualquier fabricante) permite que cualquiera lo pueda usar, independientemente del sistema operativo, navegador, etc. Si hubiesen existido fabricantes que pretendieran imponer lenguajes alternativos al HTML, tendrían un público restringido a la cantidad de clientes que usasen su tecnología.

XML

El lenguaje XML, ampliamente utilizado para estructurar la información de documentos. El XML fue estandarizado por el W3C, que autorizó su uso libremente, convirtiéndolo en una pieza clave en la interoperabilidad de la mayoría de los sistemas de información. Muchos otros lenguajes y tecnologías están basados en XML, lo que no se habría podido conseguir si este lenguaje tuviese alguna patente comercial que restringiese su uso.

Características generales de un servicio Web.

¿Qué son los Servicios Web?

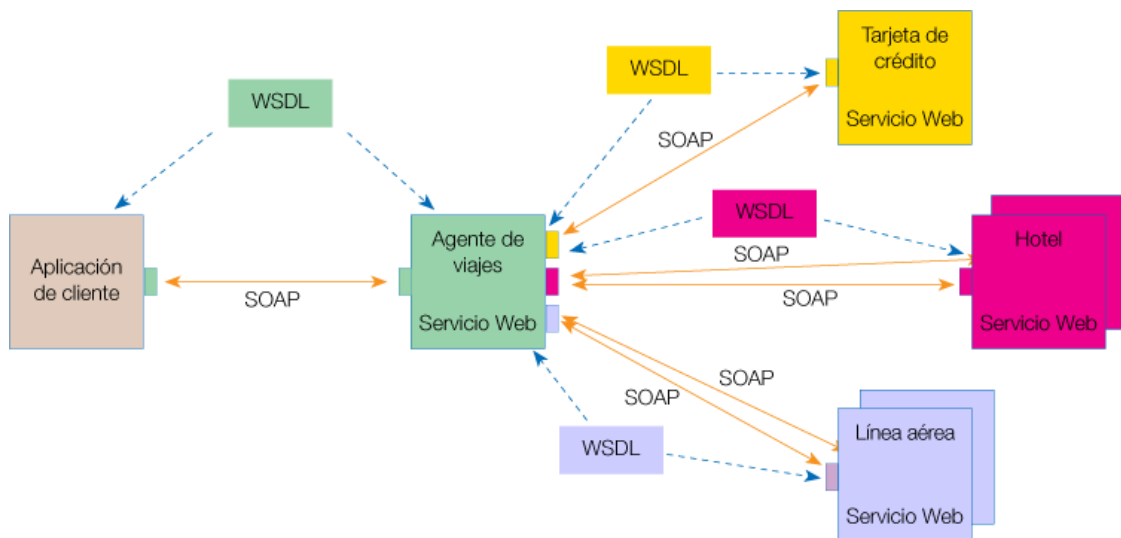
Existen múltiples definiciones sobre lo que son los Servicios Web, lo que muestra su complejidad a la hora de dar una adecuada definición que englobe todo lo que son e impliquen. Una posible sería hablar de ellos como un conjunto de aplicaciones o de tecnologías con capacidad para interoperar en la Web. Estas aplicaciones o tecnologías intercambian datos entre sí con el objetivo de ofrecer unos servicios. Los proveedores ofrecen sus servicios como procedimientos remotos y los usuarios solicitan un servicio llamando a estos procedimientos a través de la Web.

¿Para qué sirven?

Estos servicios proporcionan mecanismos de comunicación estándares entre diferentes aplicaciones, que interactúan entre sí para presentar información dinámica al usuario. Para proporcionar interoperabilidad y extensibilidad entre estas aplicaciones, y que al mismo tiempo sea posible su combinación para realizar operaciones complejas, es necesaria una arquitectura de referencia estándar.

- Componentes y funcionamiento. ¿Cómo funcionan

El siguiente gráfico muestra cómo interactúa un conjunto de Servicios Web:



Según el ejemplo del gráfico, un usuario (que juega el papel de cliente dentro de los Servicios Web), a través de una aplicación, solicita información sobre un viaje que desea realizar haciendo una petición a una agencia de viajes que ofrece sus **servicios** a través de Internet. La agencia de viajes ofrecerá a su cliente (usuario) la información requerida. Para proporcionar al cliente la información que necesita, esta agencia de viajes solicita a su vez información a otros recursos (otros Servicios Web) en relación con el hotel y la compañía aérea. La agencia de viajes obtendrá información de estos recursos, lo que la convierte a su vez en

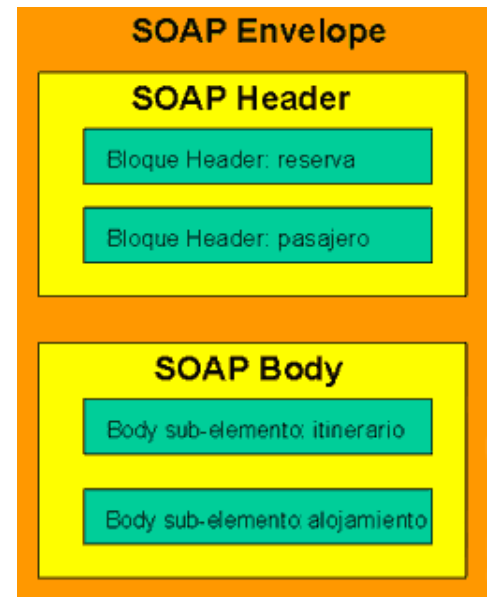
cliente de esos otros Servicios Web que le van a proporcionar la información solicitada sobre el hotel y la línea aérea. Por último, el usuario realizará el pago del viaje a través de la agencia de viajes que servirá de intermediario entre el usuario y el servicio Web que gestionará el pago.

En todo este **proceso intervienen una serie de tecnologías que hacen posible esta circulación de información**. Por un lado, estaría SOAP (Protocolo Simple de Acceso a Objetos). Se trata de un protocolo basado en XML, que permite la interacción entre varios dispositivos y que tiene la capacidad de transmitir información compleja. Los datos pueden ser transmitidos a través de HTTP, SMTP, etc. SOAP especifica el formato de los mensajes. El mensaje SOAP está compuesto por un **envelope** (sobre), cuya estructura está formada por los siguientes elementos: **header** (cabecera) y **body** (cuerpo).

Para optimizar el rendimiento de las aplicaciones basadas en Servicios Web, se han desarrollado tecnologías complementarias a SOAP, que agilizan el envío de los mensajes (MTOM) y los recursos que se transmiten en esos mensajes (SOAP-RRSHB).

Por otro lado, WSDL (Lenguaje de Descripción de Servicios Web), permite que un servicio y un cliente establezcan un acuerdo en lo que se refiere a los detalles de transporte de mensajes y su contenido, a través de un documento procesable por dispositivos. WSDL representa una especie de contrato entre el proveedor y el que solicita. WSDL especifica la sintaxis y los mecanismos de intercambio de mensajes.

Durante la evolución de las necesidades de las aplicaciones basadas en Servicios Web de las grandes organizaciones, se han desarrollado mecanismos que permiten enriquecer las descripciones de las operaciones que realizan sus servicios mediante anotaciones semánticas y con directivas que definen el comportamiento. Esto permitiría encontrar los Servicios Web que mejor se adapten a los objetivos deseados. Además, ante la complejidad de los procesos de las grandes aplicaciones empresariales, existe una tecnología que permite una definición de estos procesos mediante la composición de varios Servicios Web individuales, lo que se conoce como coreografía.



Nombres y direcciones (URIs y URLs)

NOMBRES, DIRECCIONES

Algunos ejemplos de nombres e identificadores son las URL, los nombres de dominio de Internet, los nombres de archivos... etc.

Podemos distinguir entre nombres puros (patrones de bits sin interpretar) y no puros (contienen información sobre el objeto al que nombran (p. ej: la ubicación del objeto)). En el otro extremo de un nombre puro se sitúa la dirección de un objeto, la cual es eficaz para acceder a éste, pero está el problema de que un objeto puede cambiar de localización.



Se dice que un nombre está resuelto cuando está traducido a datos relacionados con el recurso en cuestión. La asociación entre un nombre y un objeto se llama enlace. Los nombres suelen enlazarse a los atributos de los objetos y no a su implementación. Un atributo es una propiedad de un objeto.

Identificadores de Recurso Unificados (URI): Un ejemplo de URI son los URL, que son direcciones únicamente de recursos web, a los que se puede acceder con facilidad (nombre DNS más un camino hacia el recurso). Pero si un recurso se mueve o se borra, el URL no apuntará a nada (se dice comúnmente que está roto) o apuntará a otro objeto (si ha sido referenciado igual que el anterior).

Otro tipo de URI son los Nombres Uniformes de Recurso (URN), que tratan de resolver los anteriores problemas. Un servicio de búsqueda URN relaciona los URN con su URL correspondiente, la cual puede variar en el tiempo (sin que varíe el URN). Si un administrador cambia la URL, debe registrar la nueva en el servicio de búsqueda.

URIs relativas

Las URIs relativas son URIs parciales, utilizadas para referirse a un documento desde otro en la misma computadora. De esta forma, podemos definir una URI relativa como la ruta que se debe seguir desde la ubicación del documento actual (ruta de directorios) a la ubicación del recurso referido, además del nombre de archivo.

Supongamos que el documento actual, localizado en "http://servidor.es/documentos/index.asp", necesita apuntar a un documento ubicado en "http://servidor.es/documentos/nuevos/mejores/dos.asp". La URI relativa para referirse a ese recurso desde el documento actual será: "nuevos/mejores/dos.asp"

El directorio especial ".." provee una forma de ir hacia atrás al directorio "padre". De modo que para apuntar desde

"http://nuevoservidor.mil/documentos/nuevos/mejores/rec.htm" a

"http://nuevoservidor.mil/documentos/antiguos/mejores/junio.htm", la URI relativa será: "../antiguos/mejores/junio.htm"

¿Qué es un URL?

Los URLs (Uniform Resource Locator) son identificadores que permiten acceder a recursos (páginas) web. En la misma forma en que los humanos utilizamos direcciones para identificar y encontrar ubicaciones, los URLs le sirven al navegador (y otros sistemas) para encontrar una página o recurso Web en el vasto mundo del Internet.

¿Cuál es la diferencia entre URL y URI?

Aunque se acostumbra llamar URLs a todas las direcciones Web, URI es un identificador más completo y por eso es recomendado su uso en lugar de la expresión URL.

Un URI (Uniform Resource Identifier) se diferencia de un URL en que permite incluir en la dirección una subdirección, determinada por el "fragmento".

Páginas web

Una **página web** es el nombre de un documento o información electrónica adaptada para la *World Wide Web* y que puede ser accedida mediante un navegador para mostrarse en un monitor de computadora o dispositivo móvil. Esta información se encuentra generalmente en formato HTML o XHTML, y puede proporcionar navegación a otras páginas web mediante enlaces de hipertexto. Las páginas web frecuentemente incluyen otros recursos como hojas de estilo en cascada, guiones (*scripts*) e imágenes digitales, entre otros.

Las páginas web pueden estar almacenadas en un equipo local o un servidor web remoto. El servidor web puede restringir el acceso únicamente para redes privadas, p. ej., en una intranet corporativa, o puede publicar las páginas en la World Wide Web. El acceso a las páginas web es realizado mediante su transferencia desde servidores utilizando el protocolo de transferencia de hipertexto (HTTP).

Características

Una página web está compuesta principalmente por información (sólo texto y/o módulos multimedia) así como por hiperenlaces; además puede contener o asociar

Protocolo HTTP

datos de estilo para especificar cómo debe visualizarse, y también aplicaciones embebidas para así hacerla interactiva.

Las páginas web son escritas en un lenguaje de marcado que provee la capacidad de manejar e insertar hiperenlaces, generalmente HTML.

El contenido de la página puede ser predeterminado («página web estática») o generado al momento de visualizarla o solicitarla a un servidor web («página web dinámica»). Las páginas dinámicas que se generan al momento de la visualización, se especifican a través de algún lenguaje interpretado, generalmente JavaScript, y la aplicación encargada de visualizar el contenido es la que realmente debe generarlo.

Las páginas dinámicas que se generan, al ser solicitadas, son creadas por una aplicación en el servidor web que alberga las mismas.

Respecto a la estructura de las páginas web, algunos organismos, en especial el W3C, suelen establecer directivas con la intención de normalizar el diseño, y para así facilitar y simplificar la visualización e interpretación del contenido.

Una página web es en esencia una tarjeta de presentación digital, ya sea para empresas, organizaciones, o personas, así como una tarjeta de presentación de ideas y de informaciones. Así mismo, la nueva tendencia orienta a que las páginas web no sean sólo atractivas para los internautas, sino también optimizadas (preparadas) para los buscadores a través del código fuente. Forzar esta doble función puede, sin embargo, crear conflictos respecto de la calidad del contenido.



Sitios Web.

En inglés **website** o **web site**, un **sitio web** es un sitio (localización) en la World Wide Web que contine documentos (**páginas web**) organizados gerárquicamente. Cada documento (página web) contiene texto y o gráficos que aparecen como información digital en la pantalla de un ordenador. Un sitio puede contener una combinación de gráficos, texto, audio, vídeo, y otros materiales dinámicos o estáticos.

Cada sitio web tiene una **página de inicio (en inglés Home Page)**, que es el primer documento que ve el usuario cuando entra en el sitio web poniendo el nombre del dominio de ese sitio web en un navegador. El sitio normalmente tiene otros

documentos (páginas web) adicionales. Cada sitio pertenece y es gestionado y por un individuo, una compañía o una organización.

Como medio, los sitios web son similares a las películas, a la televisión o a las revistas, en que también crean y manipulan imágenes digitales y texto, pero un sitio web es también un medio de comunicación. La diferencia principal entre un sitio web y los medios tradicionales es que un sitio web está en una red de ordenadores (Internet) y está codificado de manera que permite que los usuarios interactúen con él. Una vez en un sitio web, puedes realizar compras, búsquedas, enviar mensajes, y otras actividades interactivas.

Diferencia entre sitio web y página web

A veces se utiliza erróneamente el término **página web** para referirse a **sitio web**. **Una página web es** parte de un sitio web y es un único archivo con un nombre de archivo asignado, mientras que un sitio web es un conjunto de archivos llamados páginas web. Si lo comparáramos con un libro, un sitio web sería el libro entero y una página web de ese sitio web sería un capítulo de ese libro. El título del libro sería el nombre del dominio del sitio web. Un capítulo, al igual que una página web, tiene un nombre que lo define. Decimos que sería un capítulo y no una página del libro porque a menudo es necesario desplazarse hacia abajo en la pantalla para ver todo el contenido de una página web, al igual que en un libro te *desplazas* a través de varias páginas para ver todo el contenido de un capítulo. El índice de los capítulos del libro sería el equivalente al **mapa del sitio web (sitemap en inglés)**.

Aplicación Web

Una aplicación web es cualquier aplicación que es accedida vía web por una red como internet o una intranet.

En general, el término también se utiliza para designar aquellos programas informáticos que son ejecutados en el entorno del navegador (por ejemplo, un applet de Java) o codificado con algún lenguaje soportado por el navegador (como JavaScript, combinado con HTML); confiándose en el navegador web para que reproduzca (renderice) la aplicación.

Una de las ventajas de las aplicaciones web cargadas desde internet (u otra red) es la facilidad de mantener y actualizar dichas aplicaciones sin la necesidad de distribuir e instalar un software en, potencialmente, miles de clientes. También la posibilidad de ser ejecutadas en múltiples plataformas.

Ejemplos de aplicaciones web

Las aplicaciones web son utilizadas para implementar webmail, ventas online, subastas

online, wikis, foros de discusión, weblogs, MMORPGs, redes sociales, juegos, etc.

Protocolo HTTP

El Protocolo de Transferencia de HiperTexto (Hypertext Transfer Protocol) es un sencillo protocolo cliente-servidor que articula los intercambios de información entre los clientes Web y los servidores HTTP. La especificación completa del protocolo HTTP 1/0 está recogida en el RFC 1945. Fue propuesto por Tim Berners-Lee, atendiendo a las necesidades de un sistema global de distribución de información como el World Wide Web.

Desde el punto de vista de las comunicaciones, está soportado sobre los servicios de conexión TCP/IP, y funciona de la misma forma que el resto de los servicios comunes de los entornos UNIX: un proceso servidor escucha en un puerto de comunicaciones TCP (por defecto, el 80), y espera las solicitudes de conexión de los clientes Web. Una vez que se establece la conexión, el protocolo TCP se encarga de mantener la comunicación y garantizar un intercambio de datos libre de errores.

HTTP se basa en sencillas operaciones de solicitud/respuesta. Un cliente establece una conexión con un servidor y envía un mensaje con los datos de la solicitud. El servidor responde con un mensaje similar, que contiene el estado de la operación y su posible resultado. Todas las operaciones pueden adjuntar un objeto o recurso sobre el que actúan; cada objeto Web (documento HTML, fichero multimedia o aplicación CGI) es conocido por su URL.

Funcionamiento básico.

Cada vez que un cliente realiza una petición a un servidor, se ejecutan los siguientes pasos:

- Un usuario accede a una URL, seleccionando un enlace de un documento HTML o introduciéndola directamente en el campo Location del cliente Web.
- El cliente Web decodifica la URL, separando sus diferentes partes. Así identifica el protocolo de acceso, la dirección DNS o IP del servidor, el posible puerto opcional (el valor por defecto es 80) y el objeto requerido del servidor.
- Se abre una conexión TCP/IP con el servidor, llamando al puerto TCP correspondiente.
Se realiza la petición. Para ello, se envía el comando necesario (GET, POST, HEAD,...), la dirección del objeto requerido (el contenido de la URL que sigue a la dirección del servidor), la versión del protocolo HTTP empleada (casi siempre HTTP/1.0) y un conjunto variable de información, que incluye datos sobre las capacidades del browser, datos opcionales para el servidor,...

Protocolo HTTP

- El servidor devuelve la respuesta al cliente. Consiste en un código de estado y el tipo de dato MIME de la información de retorno, seguido de la propia información.
- Se cierra la conexión TCP.

Los mensajes HTTP

- En una comunicación HTTP sólo existen dos tipos de mensajes, los de petición (request) y los de respuesta (reply).

Un mensaje de petición

Capturado usando ethereal conectándose a www.google.es mediante mozilla.

Significado de algunas de las líneas:

- Los mensajes de petición están codificados en ASCII, comienzan siempre por la cadena GET, la cadena POST o la cadena HEAD y acaban siempre en una línea en blanco (retorno de carro y nueva línea).
- El número de líneas es variable, pero como mínimo siempre existe la primera (línea de petición) donde se indica el tipo de petición (GET), la página HTML reclamada (/directorio/pagina.html, aunque en el ejemplo se trata de la página index.html que es la página por defecto) y la versión del protocolo HTTP utilizado (HTTP/1.1).
- El resto de las líneas de cabecera.
 - La primera línea, que comienza en el ejemplo por Host: especifica el host en que reside el objeto (necesario para los proxies).
 - La línea Connection: indica el tipo de conexión (keep-alive significa conexión persistente y close conexión no persistente).
 - La línea User-Agent: indica el navegador Web utilizado. Esto es importante para el servidor porque dependiendo del navegador se pueden enviar páginas HTML diferentes (con idéntica URL).
 - La línea Accept-Language: indica que el usuario prefiere recibir una versión en inglés del objeto.
- <Cuerpo de entidad> es un campo de los mensajes de petición que está vacío cuando se utiliza el método GET, pero que sí se utiliza con el método POST. Este método se emplea, por ejemplo, para rellenar formularios (donde el cliente debe enviar información al servidor). Finalmente, el método HEAD es similar al método GET, pero el servidor en su respuesta no va a incluir el objeto pedido. Este método es normalmente utilizado por los desarrolladores de aplicaciones.
- En la versión HTTP/1.1, además de GET, POST y HEAD, existen otros dos métodos: PUT que permite a un usuario cargar un objeto en el servidor Web y DELETE que permite borrarlo.

Protocolo HTTP

Un

```
HTTP/1.1 200 OK

Cache-control: private

Content-Type: text/html

Content-Encoding: gzip

Server: GWS/2.1

Content-length: 1484
```

mensaje de respuesta

Significado de algunas de las líneas:

- Los mensajes de respuesta siempre tienen 3 secciones: la línea inicial de estados, las líneas de cabecera y el cuerpo de la entidad.
- La línea inicial de estados tiene 3 campos: la versión del protocolo, el código de estado y el estado (estas dos cosas significan lo mismo). En el ejemplo, 200 OK significa que el servidor ha encontrado el objeto y que lo ha servido (en el ejemplo no se muestra tal cual, se trata de un objeto comprimido). En la siguiente tabla se presentan algunos de los códigos de control más comunes:

Código	Significado
200 OK	Petición exitosa
301 Moved Permanently	El objeto demandado ha sido movido a la URL especificada en Location:
400 Bad Request	Petición no entendida por el servidor
404 Not Found	Objeto no encontrado en el servidor
505 HTTP Version Not Supported	Obvio

- Server: indica el software del servidor Web.
- Date: indica la fecha y hora en la que se creó y envió la respuesta HTTP. Este campo es importante porque ayuda a los proxies a mantener sus cachés.
- Content-length: indica el tamaño en bytes del objeto enviado.

Métodos de petición

Tabla: Métodos HTTP

Método	Significado
GET	Devuelve el recurso identificado en la URL pedida.
HEAD	Funciona como el GET, pero sin que el servidor devuelva el cuerpo del mensaje. Es decir, sólo se devuelve la información de cabecera.
POST	Indica al servidor que se prepare para recibir información del cliente. Suele usarse para enviar información desde formularios.
PUT	Envía el recurso identificado en la URL desde el cliente hacia el servidor.
OPTIONS	Pide información sobre las características de comunicación proporcionadas por el servidor. Le permite al cliente negociar los parámetros de comunicación.
TRACE	Inicia un ciclo de mensajes de petición. Se usa para depuración y permite al cliente ver lo que el servidor recibe en el otro lado.
DELETE	Solicita al servidor que borre el recurso identificado con el URL.
CONNECT	Este método se reserva para uso con proxys. Permitirá que un proxy pueda dinámicamente convertirse en un túnel. Por ejemplo para comunicaciones con SSL.

Protocolo HTTP

HTTP define 8 métodos (algunas veces referido como "verbos") que indica la acción que desea que se efectúe sobre el recurso identificado. Lo que este recurso representa, si los datos pre-existentes o datos que se generan de forma dinámica, depende de la aplicación del servidor. A menudo, el recurso corresponde a un archivo o la salida de un ejecutable que residen en el servidor.

HEAD

Pide una respuesta idéntica a la que correspondería a una petición GET, pero sin el cuerpo de la respuesta. Esto es útil para la recuperación de meta-información escrita en los encabezados de respuesta, sin tener que transportar todo el contenido.

GET

Pide una representación del recurso especificado. Por seguridad **no debería** ser usado por aplicaciones que causen efectos ya que transmite información a través de la URI agregando parámetros a la URL.

Ejemplo:

GET /images/logo.png HTTP/1.1 obtiene un recurso llamado logo.png

Ejemplo con parámetros:

/index.php?page=main&lang=es

POST

Somete los datos a que sean procesados para el recurso identificado. Los datos se incluirán en el cuerpo de la petición. Esto puede resultar en la creación de un nuevo recurso o de las actualizaciones de los recursos existentes o ambas cosas.

PUT

Sube, carga o realiza un upload de un recurso especificado (archivo), es el camino más eficiente para subir archivos a un servidor, esto es porque en POST utiliza un **mensaje multiparte** y el mensaje es decodificado por el servidor. En contraste, el método PUT te permite escribir un archivo en una conexión socket establecida con el servidor.

La desventaja del método PUT es que los servidores de hosting compartido no lo tienen habilitado.

Ejemplo:

PUT /path/filename.html HTTP/1.1

DELETE

Borra el recurso especificado.

TRACE

Este método solicita al servidor que envíe de vuelta en un mensaje de respuesta, en la sección del cuerpo de entidad, toda la data que reciba del mensaje de solicitud. Se utiliza con fines de comprobación y diagnóstico.

OPTIONS

Devuelve los métodos HTTP que el servidor soporta para un URL específico. Esto puede ser utilizado para comprobar la funcionalidad de un servidor web mediante petición en lugar de un recurso específico.

CONNECT

Este método se reserva para uso con proxies. Permitirá que un proxy pueda dinámicamente convertirse en un túnel. Por ejemplo para comunicaciones con SSL.

Las cabeceras del protocolo http

Connection (*conexión*)

Permite especificar diferentes opciones para la conexión. Por ejemplo:

Connection: close

indica que la conexión debe cerrarse una vez transmitido el mensaje completo



Content-Language (*idioma del contenido*)

Esta cabecera indica el idioma de los destinatarios del recurso. Si no existe, se entiende que el recurso está orientado a todos los usuarios, independientemente del idioma. Esta cabecera permite listar varios idiomas. Por ejemplo, una herramienta on-line de traducción inglés-francés, podría incluir en sus páginas la cabecera:

Content-Language: es, fr

Es importante recalcar que esta cabecera no indica necesariamente el idioma del documento, sino del público al que objetivamente se orienta. Un texto sencillo de inglés para estudiantes hispanoparlantes incluiría la cabecera:

Content-Language: es

aunque el contenido pueda estar en inglés (y, por tanto, las metaetiquetas HTML indiquen que se trata de un documento en inglés).

Content-Length (*longitud del contenido*)

Indica la longitud del cuerpo del recurso, expresada en número de octetos.

Content-Location (*localización del contenido*)

Dirección complementaria que ofrece el servidor en su respuesta. Esta nueva dirección (una URI absoluta o relativa) no corrige la dirección original del recurso solicitado por el cliente, sino que ofrece una ruta a un recurso que complementa al solicitado originalmente.

Content-Type (*tipo de contenido*)

Indica, como su nombre indica, el tipo de contenido del recurso. Así, la cabecera

Content-Type: text/html; charset=ISO-8859-1

indica que el recurso es de tipo texto, concretamente código HTML, y codificado según la especificación ISO-8859-1.

Date (*fecha*)

Indica la fecha de creación del recurso. Tiene la forma:

Date: Tue, 12 Jul 2005 09:32:25 GMT

Expect (*espera*)

Mediante esta cabecera, el cliente indica qué tipo de respuesta espera del servidor. Si el servidor no está preparado para responder como el cliente espera, debe indicarlo mediante el envío de un código de estatus 417 (*Expectation Failed*).

Expires (*expiración*)

Indica la fecha a partir de la cual el recurso debe considerarse obsoleto. Un ejemplo:

Date: Tue, 12 Jul 2005 09:32:25 GMT

From (*"desde"*)

Dirección de correo electrónico del usuario (humano) autor de la solicitud.

- Más información sobre la [cabecera From](#)

If-Match ("*si cuadra*")

Se usa junto con la cabecera de método para hacerlo condicional. Esto permite actualizaciones eficientes de la caché. Si el cliente guarda en su caché alguna entidad (algún elemento distinguible) del recurso solicitado puede verificar gracias a esta cabecera si esta entidad sigue estando en vigor, es decir, si la copia guardada en la caché sigue siendo válida.

If-Modified-Since ("*si se ha modificado desde*")

Igual que la cabecera If-Match, If-Modified-Since se usa con la cabecera que indica el método para expresar una condición. Si el recurso no ha variado desde la fecha indicada por el cliente, el servidor no debe enviarlo. Enviaré, en cambio, un código de estatus 304, confirmando al cliente (navegador, por ejemplo, o robot de un buscador) que la copia que tiene en caché sigue siendo una copia fiel del recurso guardado en el servidor.

If-None-Match ("*si no cuadra*")

Igual que las cabecera If-Match e If-Modified-Since, se usa junto con la cabecera de método para someterlo a una condición. Funciona de forma inversa a if-Match. El servidor no debe ejecutar la solicitud (expresada mediante la cabecera de método) si la entidad expresada por la condición de If-None-Match se cumple.

IP (*remote adress*)

No es estrictamente una cabecera del protocolo HTTP, sino del protocolo TCP/IP. Expresa la identificación numérica de una máquina.

Host (*servidor*)

Nombre del servidor.

Last-Modified (*última modificación*)

Mediante esta cabecera el servidor informa de la fecha y hora en que el recurso fue modificado por última vez.

- Más información sobre la [cabecera Last-Modified](#)

Location (*localización*)

Mediante este campo el servidor indica la dirección (la URL) de un recurso cuando no se encuentra en la dirección en que se ha solicitado. De esta forma, el servidor invita al navegador (o al software del cliente en general) a que se redirija a la nueva localización.

Referer (*remitente*)

Documento desde el cual se ha realizado la solicitud actual. Si desde la URL www.cibernetia.com/index.php clicamos el enlace que lleva a

www.cibernetia.com/headers_manual/index.php, la primera URL figurará como *referer* en la solicitud de la segunda URL.

Request (*solicitud*)

Indica el fichero (el documento) solicitada y el método y versión del protocolo que se van a emplear para realizar la conexión.

- Más información sobre el [método, versión del protocolo y fichero solicitado](#)

Status Code (*código de estado*)

Mediante el código de estado el servidor informa al navegador sobre cómo ha resuelto la solicitud de un documento. Esta cabecera nos indicará, por ejemplo, si se ha servido el documento con éxito o se ha producido algún problema, como un error interno del servidor, o alguna incidencia, como una redirección hacia otra URL diferente.

- Puedes consultar la [lista de códigos de status](#)

User-Agent (*agente de usuario*)

El *user-agent* identifica el software de la máquina cliente (es decir, se refiere al software instalado en el ordenador que solicita una página web). La identificación se realiza, normalmente, mediante una combinación de sistema operativo y navegador.

Un par de ejemplos:

- *Mozilla/4.0 (compatible; MSIE 6.0; Windows 98)*
Esta cabecera indica que el cliente está navegando con la versión 6.0 de Internet Explorer corriendo en un Windows 98.
- *Googlebot/2.1 (+http://www.google.com/bot.html)*
En este caso es un robot el que está solicitando la página, concretamente *Googlebot*, la araña de Google.

Errores

Las 5 clases definidas son las siguientes:

1xx. Informacional. Se recibe la petición y se continua con el proceso. Los códigos en este rango indican respuestas provisionales. Los servidores web no deben enviar mensajes 1xx al cliente HTTP excepto bajo condiciones experimentales.

2xx. Éxito. Esta clase de códigos indican que la petición del cliente fue recibida, entendida, aceptada y procesada exitosamente.

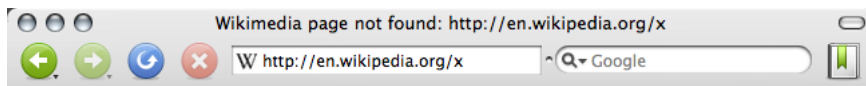
3xx. Redireccionamiento. Para estos códigos el cliente debe realizar acciones adicionales para completar la petición. La acción requerida debe ser portada por el *user agent* sin la interacción del usuario si y solo si el método usado en la segunda petición es de tipo *GET* o *HEAD*. El *user agent* no debería redireccionar

Protocolo HTTP

automáticamente más de 5 veces, sino se considera un bucle infinito.

4xx. Error en el Cliente. Estos códigos son arrojados cuando el cliente parece tener un error. Estos tipos de errores son los más comunes que se pueden encontrar.

5xx. Errores de Servidor. El servidor falla cuando aparentemente se esta ante una petición válida. El Servidor responde con este tipo de errores cuando es incapaz de realizar la petición



404 error: File not found

The [URL](#) you requested was not found.

Did you mean to type <http://en.wikipedia.org/wiki/x>? You will be automatically redirected there in five seconds.

Maybe you would like to look at:

- [The main page](#)
- [The list of Wikimedia downloads](#)

A project of the [Wikimedia foundation](#).

Los códigos de respuesta

Son los códigos que se ven cuando el navegador no puede mostrar la página solicitada. El código de respuesta está formado por tres dígitos: el primero indica el estado y los dos siguientes explican la naturaleza exacta del error.

Código	Mensaje	Descripción
10x	Mensaje de información	Estos códigos no se utilizan en la versión 1.0 del protocolo
20x	Éxito	Estos códigos indican la correcta ejecución de la transacción
200	OK	La solicitud se llevó a cabo de manera correcta
201	CREATED	Sigue a un comando POST e indica el éxito, la parte restante del cuerpo indica la dirección URL donde se ubicará el documento creado recientemente.
202	ACCEPTED	La solicitud ha sido aceptada, pero el procedimiento que sigue no se ha llevado a cabo
203	PARTIAL INFORMATION	Cuando se recibe este código en respuesta a un comando de GET indica que la respuesta no está completa.
204	NO RESPONSE	El servidor ha recibido la solicitud, pero no hay información de respuesta
205	RESET CONTENT	El servidor le indica al navegador que borre el contenido en

Protocolo HTTP

		los campos de un formulario
206	PARTIAL CONTENT	Es una respuesta a una solicitud que consiste en el encabezado <i>range</i> . El servidor debe indicar el encabezado <i>content-Range</i>
30x	Redirección	Estos códigos indican que el recurso ya no se encuentra en la ubicación especificada
301	MOVED	Los datos solicitados han sido transferidos a una nueva dirección
302	FOUND	Los datos solicitados se encuentran en una nueva dirección URL, pero, no obstante, pueden haber sido trasladados
303	METHOD	Significa que el cliente debe intentarlo con una nueva dirección; es preferible que intente con otro método en vez de GET
304	NOT MODIFIED	Si el cliente llevó a cabo un comando GET condicional (con la solicitud relativa a si el documento ha sido modificado desde la última vez) y el documento no ha sido modificado, este código se envía como respuesta.
40x	Error debido al cliente	Estos códigos indican que la solicitud es incorrecta
400	BAD REQUEST	La sintaxis de la solicitud se encuentra formulada de manera errónea o es imposible de responder
401	UNAUTHORIZED	Los parámetros del mensaje aportan las especificaciones de formularios de autorización que se admiten. El cliente debe reformular la solicitud con los datos de autorización correctos
402	PAYMENT REQUIRED	El cliente debe reformular la solicitud con los datos de pago correctos
403	FORBIDDEN	El acceso al recurso simplemente se deniega
404	NOT FOUND	Un clásico. El servidor no halló nada en la dirección especificada. Se ha abandonado sin dejar una dirección para redireccionar... :)
50x	Error debido al servidor	Estos códigos indican que existe un error interno en el servidor
500	INTERNAL ERROR	El servidor encontró una condición inesperada que le impide seguir con la solicitud (una de esas cosas que les suceden a los servidores...)
501	NOT IMPLEMENTED	El servidor no admite el servicio solicitado (no puede saberlo todo...)
502	BAD GATEWAY	El servidor que actúa como una puerta de enlace o proxy ha recibido una respuesta no válida del servidor al que intenta acceder
503	SERVICE UNAVAILABLE	El servidor no puede responder en ese momento debido a que se encuentra congestionado (todas las líneas de comunicación se encuentran congestionadas, inténtelo de nuevo más adelante)
504	GATEWAY TIMEOUT	La respuesta del servidor ha llevado demasiado tiempo en relación al tiempo de espera que la puerta de enlace podía admitir (excedió el tiempo asignado...)

Almacenamiento en cache.

Se llama **caché web** a la caché que almacena documentos web (es decir, páginas, imágenes, etcétera) para reducir el ancho de banda consumido, la carga de los servidores y el retardo en la descarga. Un caché web almacena copias de los documentos que pasan por él, de forma que subsiguientes peticiones pueden ser respondidas por el propio caché, si se cumplen ciertas condiciones.

Tipos de cachés web

Las cachés web pueden utilizarse de diversas formas. Las cachés de agente de usuario (*User-Agent*), como las presentes en los navegadores web, son **cachés privados**, que funcionan solo para un único usuario. También existen paquetes específicos que se instalan como proxy local y actúan como caché además de realizar otras tareas, como por ejemplo Proxomitron.

Los intermediarios en la comunicación cliente-servidor también pueden implementar **cachés compartidos** (también llamadas *proxy-cachés directos*) que sirvan páginas a varios usuarios. Los proxy-cachés suelen ser usados por los proveedores de servicios de Internet (ISP), universidades y empresas para ahorrar ancho de banda. La intermediación de estos proxy-cachés difieren de la de los privados en que los clientes no necesitan ser explícitamente configurados para usarlos. Algunos paquetes que pueden ser usados como proxy-cachés son Squid, Microsoft ISA Server y Blue Coat.

Las **cachés pasarela** (llamadas también *proxy-cachés inversos* o aceleradores web) funcionan a cargo del propio servidor original, de forma que los clientes no distinguen unos de otros. Puede hacerse funcionar conjuntamente varias cachés pasarela para implementar una Content Delivery Network (CDN), como es el caso de Akamai. Paquetes como Varnish Cache pueden usarse para este propósito.

Los intermediarios que funcionan como caché realizan con frecuencia otras tareas, tales como la autenticación de usuarios y el filtrado de contenidos. Varios cachés pueden ser coordinados entre sí con la ayuda de protocolos específicos tales como ICP o HTCP.

Control de los cachés web

El protocolo HTTP define tres mecanismos básicos para controlar las cachés:

- **Frescura**, que permite que una respuesta sea usada sin comprobar de nuevo el servidor origen, y puede ser controlada tanto por el servidor como el cliente. Por ejemplo, la cabecera de respuesta `Expires` facilita una fecha en la que el documento caduca, y la directiva `Cache-Control: max-age` informa al caché del número de segundos durante los que la respuesta será válida.
- **Validación**, que puede usarse para comprobar si una respuesta cacheada sigue siendo buena tras caducar. Por ejemplo, si la respuesta tiene una cabecera `Last-Modified`, un caché puede hacer una petición condicional usando la cabecera `If-Modified-Since` para saber si la página cambió.

- **Invalidación**, que normalmente es un efecto secundario de otra petición que pasa por la caché. Por ejemplo, si la URL asociada con una respuesta cacheada es solicitada posteriormente mediante una petición POST, PUT o DELETE, la respuesta cacheada quedará invalidada.

- Redirecciones

¿Cuándo se necesita una redirección web?

Existen diferentes casos de real necesidad para los cuales se debe de usar la redirección: por ejemplo en caso de cambio en la Url de nuestro portal, variación del nombre de un fichero, o cambio de carpeta en la arborescencia de nuestro sitio web.

Su funcionamiento:

- Se necesita que el encabezamiento enviado por la página consultada corresponda a su estatus. Por ejemplo, si una página ha cambiado de lugar en nuestro portal, es de vital importancia que la antigua Url haga un redireccionamiento hacia la nueva, utilizando un encabezamiento HTTP que precise que esta página ha cambiado de manera definitiva de dirección (código 301) – Esto permitirá al robot el no volver a indexar nunca la antigua Url, poniendo al día su base de datos aplicando la nueva Url a la página en cuestión.

Si no aplicamos la redirección desde la antigua Url, el robot y los visitantes obtendrán un error 404, lo cual no será una buena señal, ya que de este modo el encontrar la nueva dirección se convertiría en una misión complicada.

Compresión

El protocolo de transferencia de documentos de hipertexto (HTTP), utilizado en la web, provee la poderosa pero poco conocida habilidad de trabajar con información comprimida utilizando algoritmos de compresión estándares en la industria.

Se trata entonces de comprimir la información enviada por el servidor del sitio web, dejando al navegador del visitante el trabajo de descomprimirlo. Esto se realiza automáticamente, sin que el visitante lo perciba ni deba intervenir.

Ventajas

Al comprimir información, esta se envía mucho más rápido desde el servidor hasta el navegador del visitante, produciendo así una mejor experiencia en la visita del sitio y recortando la cantidad de ancho de banda --y sus costos-- utilizado por el sitio. En

general se puede conseguir una compresión de entre 5:1 y 10:1 (y de hasta 50:1), logrando así una reducción del tamaño de las páginas de, en promedio, 65% a 85%. Esto resulta generalmente en una transferencia de entre 3 a 6 veces más rápido, Google, Amazon, Yahoo, AT&T y una larga lista de gigantes utilizan esta tecnología. Por ejemplo la pagina principal de Google tiene apenas 1.412 bytes, que sin compresión hubiera tenido 3.873 bytes, logrando así un ahorro del 63.5%.

Desventajas

Como la compresión se realiza dinámicamente, esta requiere algo de procesamiento. Sin embargo, en nuestra experiencia esto no tiene un impacto significativo en la performance del servidor.

Cookies

Una cookie es **información** enviada desde un servidor de páginas web y **almacenada** en el **disco duro** del **visitante** a través del navegador. Esta información será reenviada de nuevo al servidor en cada petición, de forma que el servidor puede identificar o recuperar información sobre el usuario que está accediendo.

¿Por qué se han creado las cookies?

Las cookies fueron implementadas por primera vez por Netscape Communications para la creación del típico cesto de comprar en una tienda online. El problema hasta entonces era que el protocolo HTTP carecía de la posibilidad de mantener información por sí mismo. Los métodos usados antes eran:

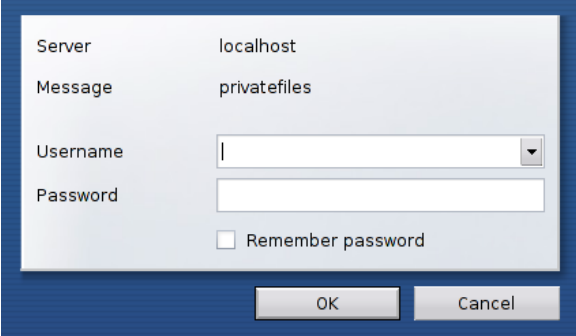
- **Identificación por IP:** un método muy poco fiable, pues bajo una misma IP podían estar accediendo distintos usuarios (por ejemplo desde un ciber), además que la dirección IP de un usuario puede cambiar.
- **Por URL:** Consiste en añadir la información en la URL, después del interrogante ?. Esta es una técnica más precisa en lo que se refiere a identificación, pero tiene problemas de seguridad.

Gracias a las cookies, un servidor web puede identificar un conjunto pc-navegador-usuario y mostrar la información adecuada a ese conjunto, por ejemplo un carrito de compra que haya creado.

Autenticación

La autenticación es el proceso de identificar si un cliente es elegible para tener acceso a un recurso. El protocolo HTTP soporta la autenticación como un medio de negociar el acceso a un recurso seguro.

La solicitud inicial de un cliente es normalmente una solicitud anónima, que no contiene ninguna información de autenticación. Las aplicaciones de servidor HTTP pueden denegar la solicitud anónima indicando que se requiere la autenticación. La aplicación de servidor envía encabezados de la autenticación de WWW para indicar los esquemas de autenticación soportados.



Existen varios tipos de autenticación:

- **Autenticación básica:** soportado por todos los servidores web y navegadores, así como terminales móviles.
- **Autenticación mediante resúmenes ó digest:** soportada por todos los servidores y en algunos navegadores.
- **Autenticación de Windows integrada:** evolución de la antigua autenticación por desafío respuesta de Windows. Solamente en plataforma Windows para navegador Internet Explorer.
- **Autenticación https:** es una combinación del protocolo HTTP y protocolos criptográficos

AUTENTICACIÓN BÁSICA

Cuando el usuario accede a un recurso del servidor web protegido mediante autenticación básica, tiene lugar el siguiente proceso:

1. El navegador presenta al usuario la ventana de autenticación, para que introduzca su nombre y contraseña.
2. El navegador intenta establecer una conexión con el servidor utilizando esta información.
3. Si el servidor rechaza la información de autenticación, el navegador le presenta nuevamente la ventana al usuario hasta que éste introduce por fin una contraseña válida o cierra la ventana.
4. Cuando el servidor web verifica con éxito los datos de autenticación, se establece la conexión de acceso al recurso protegido.

Ventana de autenticación que presenta el navegador cuando se pretende acceder a un recurso protegido.

El gran inconveniente de este método es que la contraseña viaja en claro desde el navegador del usuario hasta el servidor, por lo que cualquier atacante armado con un sniffer podrá interceptarla inmediatamente. Su mayor ventaja, en cambio, es que forma parte del protocolo HTTP 1.0 y posteriores versiones, estando por tanto universalmente aceptado en la práctica totalidad de navegadores. Cuando se salta a otra página o recurso igualmente protegidos por este método, en lugar de presentar al usuario una nueva ventana de autenticación, lo cual resultaría muy engorroso, el navegador recuerda la información tecleada por el usuario la primera vez y se la envía al servidor automáticamente. Nótese que en las sucesivas autenticaciones, esta información continúa viajando en claro, aunque el usuario no sea consciente de ello.

AUTENTICACIÓN MEDIANTE RESÚMENES O DIGEST

Dado que el método anterior envía las contraseñas en claro, no resulta muy adecuado cuando las exigencias de seguridad son elevadas. Para paliar este inconveniente, además de cifrar el canal con SSL, otra alternativa consiste en enviar un resumen criptográfico de la contraseña (un hash) en vez de la propia contraseña, de la siguiente forma:

1. El servidor envía al navegador cierta información que será utilizada en el proceso de autenticación.
2. El navegador añade esta información a su nombre de usuario y contraseña, junto con otra información adicional, y crea un resumen del conjunto. Esta información adicional persigue el cometido de impedir ataques de reactuación, en los que un atacante intercepta y copia el resumen, volviéndolo a utilizar para autenticarse él mismo ante el servidor.
3. Se envía en claro tanto el resumen como la información adicional al servidor a través de la red.
4. El servidor añade esta información adicional a una copia en claro de la contraseña del cliente y crea el resumen del conjunto.
5. El servidor compara el resumen que ha creado con el que le ha llegado del navegador.
6. Si ambos números coinciden, se le concede acceso al usuario.

La autenticación mediante resúmenes ha sido incorporada al estándar HTTP 1.1, pero desgraciadamente la mayoría de navegadores no la soportan. Se puede encontrar una descripción sobre su funcionamiento y consideraciones sobre su seguridad en un borrador de Internet.

AUTENTICACIÓN DE WINDOWS INTEGRADA

Este tipo de autenticación, exclusivo de NT, ha pasado a llamarse Autenticación Integrada de Windows y constituye una variante de la autenticación mediante resúmenes criptográficos. Se trata igualmente una forma segura de autenticación en la medida en que no se envían ni la contraseña ni el nombre de usuario a través de la Red. En su lugar, el navegador tiene que demostrarle al servidor que conoce la clave por medio de un corto intercambio de datos, pero sin revelar nunca la clave. No obstante, debido a los detalles de implantación, resulta incompatible con la autenticación por resúmenes, por lo que su uso se circunscribe a servidores NT.

Funciona de la siguiente manera:

1. A diferencia de la autenticación básica o mediante resúmenes, no se le presenta al usuario una ventana para que introduzca su nombre y contraseña, sino que se utiliza la información de la sesión abierta por el ordenador del cliente, es decir, se utiliza el nombre de usuario, contraseña y dominio que se utilizó al entrar al ordenador desde el que se está navegando.
2. Si este intercambio inicial falla, entonces se le presenta al usuario la ventana de identificación, donde introduce su nombre, contraseña y además el dominio.
3. Si los datos proporcionados no son correctos, se le presenta esta ventana repetidamente hasta que el usuario introduce la información correcta o la cierra.

Aunque este método presenta la ventaja de no transmitir las contraseñas a través de la Red, superando el inconveniente de la autenticación básica, posee dos importantes limitaciones para su uso en Internet:

- Sólo está soportado por Microsoft Internet Explorer, versión 2.0 o posterior y servidores NT.
- No funciona para conexiones con proxy.

Estas limitaciones hacen que la autenticación integrada de Windows sea más adecuada para intranets, en las que se puede exigir a los usuarios que el navegador que utilicen sea Internet Explorer y en las que tanto los servidores como los clientes se encuentran detrás del mismo proxy. Es muy importante que las cuentas de los usuarios que se autentifiquen de esta forma posean el derecho de Acceder a este equipo desde la red.

AUTENTICACIÓN HTTPS

El uso del formato HTTPS para enviar mensajes garantiza la autenticación de los usuarios que necesitan acceso a los recursos de Message Queue Server por medio de un servidor Web estableciendo una conexión de nivel de sockets seguro (SSL) para conseguir una comunicación segura entre un remitente y un destinatario. El emisor es siempre considerado como cliente SSL y el destinatario como servidor SSL independientemente de si el equipo está ejecutando Message Queue Server o software de cliente. Tenga en cuenta que la autenticación para establecer una sesión de SSL no es la misma que la autenticación de mensajes, que confirma que un mensaje no se ha manipulado y se puede utilizar para comprobar la identidad del

Protocolo HTTP

remitente. Para obtener información acerca de la autenticación de mensajes, consulte Administrar la autenticación de mensajes.

En la autenticación HTTPS se utilizan dos tipos de certificados:

- **Certificados de servidor.** Este certificado contiene información sobre el servidor que permite a un cliente identificar el servidor antes de compartir información confidencial.
- **Certificados de cliente.** Este certificado contiene información personal sobre el usuario e identifica el servidor al cliente de SSL (el remitente).

Conexiones persistentes

Conexión persistente del HTTP

Las conexiones persistentes del HTTP, también llamadas HTTP guardar-vivo, o reutilización de la conexión del HTTP, son la idea de usar la misma conexión del TCP para enviar y para recibir múltiplo Peticiones del HTTP/responses, en comparación con abrir una nueva conexión para cada solo par de la petición/de la respuesta.

Ventajas

- menos CPU y uso de la memoria (porque pocas conexiones están abiertas simultáneamente)
- permite Can#ería del HTTP de peticiones y de respuestas
- reducido congestión de red (menos Conexiones del TCP)
- reducido estado latente en las peticiones subsecuentes (no apretón de manos)
- los errores se pueden divulgar sin la pena de cerrar la conexión del TCP

Según RFC 2616 (página 47), un cliente single-user no debe mantener más de 2 conexiones con ningún servidor o poder. A poder debe utilizar hasta las conexiones $2*N$ a otro servidor o poder, donde está el número N de usuarios simultáneamente activos. Estas pautas se piensan para mejorar tiempos de reacción del HTTP y para evitar la congestión.

Conexión HTTP persistente, también llamado mantenimiento de conexiones HTTP, o volver a usar la conexión HTTP, es la idea de usar la misma conexión TCP para enviar y recibir múltiples peticiones HTTP / respuestas, en lugar de abrir una nueva conexión para todos y cada uno de petición / respuesta par.

En HTTP 1.0, no hay especificación oficial para saber cómo funciona keepalive. Era, en esencia, añadido a un protocolo existente. Si el navegador es compatible con mantenimiento de conexión, se añade una cabecera adicional a la solicitud:

Entonces, cuando el servidor recibe la solicitud y genera una respuesta, sino que también agrega un encabezado a la respuesta:

Después de esto, la conexión no se cae, sino que se mantiene abierta. Cuando el cliente envía una nueva solicitud, que utiliza la misma conexión. Esto continuará hasta que el cliente o el servidor decide que la conversación ha terminado, y uno de ellos cae la conexión.

En HTTP 1.1 se consideran todas las conexiones persistentes menos que se declare lo contrario. Las conexiones HTTP persistentes no utilizan separar los mensajes de keepalive, que sólo permiten múltiples solicitudes para el uso de una única conexión. Sin embargo, el tiempo de espera de conexión por defecto de Apache httpd 2.0 es tan poco como 15 segundos y para Apache 2.2 a 5 segundos. La ventaja de un tiempo corto es la capacidad de ofrecer múltiples componentes de una página web de forma rápida sin atar varios procesos de servidor o discusiones durante mucho tiempo.

Configuración de un servidor Web

- Instalación, configuración y uso.

Instalar el Servidor Apache.

La instalación es sencilla, descarga de Apache.org. la última versión para Windows, puedes utilizar el siguiente vínculo. [Descargar Apache](#)
Crea dos carpetas en la unidad C, la primera de nombre *Apache* y la segunda *servidor_web*. Descomprime el archivo descargado y ejecútalo, sigue los pasos de la instalación y de los datos que te piden solo escoge el destino de la instalación, que será la carpeta que creaste en C:\Apache, los otros datos déjalos de la forma predeterminada para configurarlos más tarde.
El programa al instalarse crea un icono en el área de notificación que te permitirá: iniciar, detener y reiniciar Apache; tienes que tener en cuenta que cualquier cambio que hagas en el archivo de configuración no tendrá efecto hasta que reinicies el servidor.



Como configurar el Servidor Apache

Toda la configuración para el funcionamiento de Apache se guarda en un archivo de texto nombrado: **httpd.conf** que se encuentra en la ruta C:\Apache\conf, lo podemos editar en cualquier editor de texto como el Bloc de notas pero un programa recomendado es *Notepad++*, software libre que es inmejorable.

Puedes descargar Notepad++ desde aquí.

Tienes dos opciones a continuación:

- 1- Primera opción, la más sencilla, descarga en el siguiente link una copia del archivo [httpd.conf](#), descomprímelo, cópialo o muévelo a la carpeta C:\Apache\conf y sustituye el archivo original, ya tendrás listo para funcionar el servidor.
- 2- La otra opción, más avanzada pero no difícil, abre el archivo httpd.conf y edita manualmente las líneas que se indican:

Protocolo HTTP

Todas las líneas que comienzan con el símbolo # son comentarios, explican en cada sección las distintas opciones pero se encuentran en inglés.

La línea 52 **Listen** indica el puerto y dirección IP por el que el servidor va a recibir las peticiones, puedes usarla de las siguientes maneras:

- 1- El servidor va recibir peticiones solo de la misma PC: **Listen localhost:80**
- 2- Recibirá peticiones de otras máquinas en una red local: **Listen 80**

En la línea 149 **DocumentRoot** es necesario especificar la ruta de la carpeta local que contendrá las páginas y archivos a servir, en tu caso será la carpeta que creaste en *C:/servidor_web*, quedaría de la siguiente forma:

DocumentRoot "C:/servidor_web"

La línea 177 **<Directory>** establece los permisos necesarios al directorio anterior, quedaría: **<Directory "C:/servidor_web">**

Esta es la configuración con los parámetros esenciales para comenzar a utilizar Apache. Guarda los cambios realizados y reinicia el servidor dando clic en el icono del área de notificación.

USO

Con la instalación de Apache es posible disponer en nuestra PC de un pequeño servidor que nos posibilitará entre otras tareas:

- 1- Probar y ver las páginas web como verdaderamente van a mostrarse desde internet antes de subirlas a un host o servidor en la red. Útil e indispensable si tienes o vas a crear tu sitio por modesto que este sea.
- 2- Crear mediante el modulo Virtual Host múltiples sitios web en nuestra PC, que podemos descargar con wget y acceder a ellos igual que en la red pero esta vez de forma local.
- 3- Poder ver localmente páginas web hechas en lenguaje php.
- 4- Servir nuestras páginas o sitio web directamente a internet, a los que puede acceder y conectarse cualquier persona desde el exterior, en este caso lógicamente el funcionamiento del servidor estará limitado al tiempo que tengamos funcionando la PC y a las posibilidades de nuestra conexión. Puede constituir una experiencia muy alentadora para cualquier aficionado, esta posibilidad da la ventaja de que no es necesario depender de ninguna compañía ni servidor remoto para subir a la red el contenido que queremos mostrar. Es como montar una pequeña estación de radio y empezar a transmitir, (una similitud) pero en este caso el alcance es global.
- 5- Puede actuar como intermediario entre nuestra PC e internet lo que nos da varias ventajas en el ámbito de la seguridad.
- 6- A través de él podemos servir internet a varias PC conectadas en una red local.
- 7- Es posible activar un módulo que permite guardar en cache todas las páginas cargadas lo que mejorará el rendimiento de nuestra navegación.

Protocolo HTTP

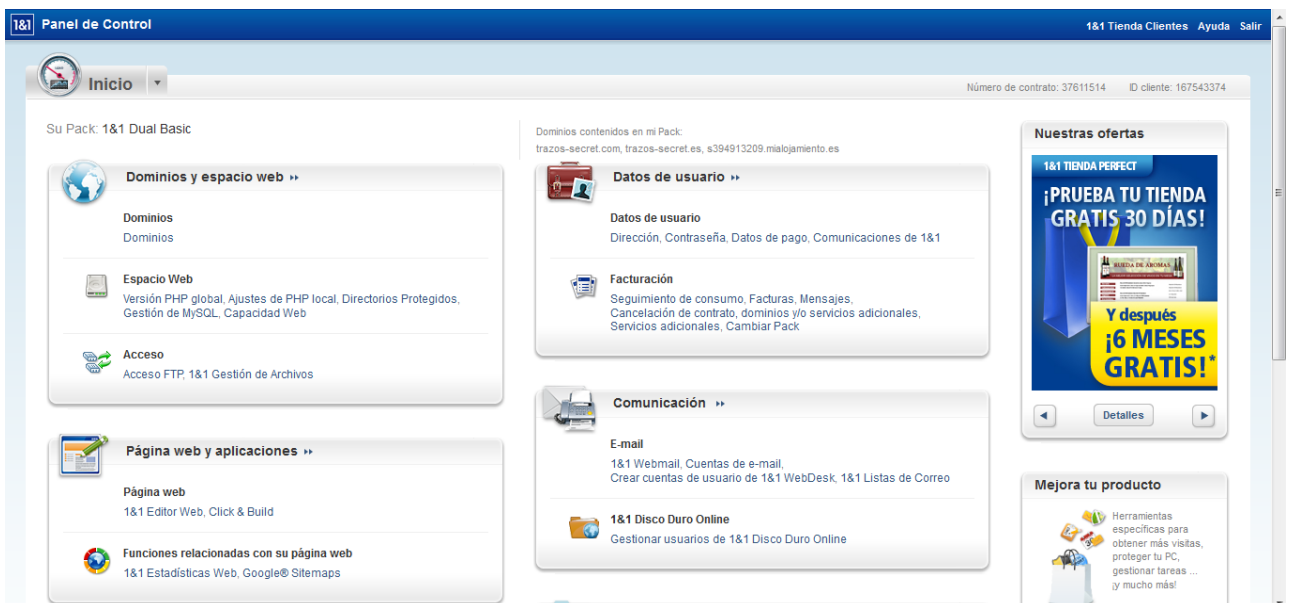
Autenticación y control de acceso.

Esquema de autenticación	Descripción
Anónimo	Una solicitud anónima no contiene ninguna información de autenticación. Esto equivale a conceder acceso al recurso a todo el mundo.
Básica	La autenticación básica envía una cadena codificada por Base64 que contiene un nombre de usuario y contraseña para el cliente. Base64 no es una forma de cifrado y debe considerarse igual que enviar el nombre de usuario y contraseña en texto no cifrado. Si un recurso necesita ser protegido, considere fervientemente utilizar un esquema de autenticación distinto de la autenticación básica.
Implícita	<p>La autenticación implícita es un esquema de desafío-respuesta destinado a reemplazar a la autenticación básica. El servidor envía una cadena de datos aleatorios llamada <i>valor de seguridad (nonce)</i> al cliente a modo de desafío. El cliente responde con un hash que incluye el nombre de usuario, contraseña y valor de seguridad, entre otra información adicional. La complejidad que introduce este intercambio y el hash de datos hacen que sea más difícil robar y reutilizar las credenciales del usuario con este esquema de autenticación.</p> <p>La autenticación implícita requiere el uso de cuentas de dominio de Windows. El <i>dominio kerberos</i> implícito es el nombre de dominio de Windows. Por consiguiente, no puede utilizar un servidor que se ejecute en un sistema operativo que no admita los dominios de Windows, como Windows XP Home Edition, con autenticación implícita. A la inversa, cuando el cliente se ejecuta en un sistema operativo que no admite los dominios de Windows, una cuenta de dominio se debe especificar explícitamente durante la autenticación.</p>
NTLM	La autenticación de NT LAN Manager (NTLM) es un esquema de desafío-respuesta que es una variación más segura de la autenticación implícita. NTLM utiliza las credenciales de Windows para transformar los datos del desafío en lugar del nombre de usuario descodificado y contraseña. La autenticación NTLM requiere varios intercambios entre el cliente y servidor. El servidor y cualquier proxy que intervenga deben admitir las conexiones persistentes para completar correctamente la autenticación.
Negotiate	Negociar la autenticación automáticamente selecciona entre el protocolo Kerberos y la autenticación NTLM, dependiendo de la disponibilidad. Se utiliza el protocolo Kerberos si está disponible; de lo contrario, se prueba NTLM. La autenticación Kerberos mejora significativamente en NTLM. La autenticación Kerberos es más rápida que NTLM y además permite el uso de autenticación mutua y delegación de credenciales a los equipos remotos.

Registro y monitorización del servicio Web

Los archivos de [registros](#) o archivos log como se conocen comúnmente, son archivos en donde se van almacenando un registro de todos los eventos que ocurren en un sistema durante un periodo de tiempo en particular. Estos archivos son usados tanto por el sistema operativo como por las aplicaciones o demonios (procesos) para registrar datos o información sobre un evento en particular. En un sistema Linux podemos encontrar estos archivos de registro o logs en la carpeta `/var/log`. En esta carpeta encontraremos casi todos los archivos de registros de un sistema, pero cabe destacar que muchas aplicaciones crean estos archivos en sus propias carpetas fuera de `/var/log`.

Ahora bien, ¿En qué nos sirve los logs para [monitorear](#) nuestro sistema? pues muy sencillo, los principales archivos logs que están en la carpeta `/var/log` van almacenando información de casi todos los eventos que ocurren en tu PC prácticamente desde que la enciendes y en ellos podremos ver por ejemplo que pasa internamente en Linux cuando conectas una Memoria USB, un [Modem](#) USB o cuando estás conectado a [internet](#) puedes ver los intentos de entrada bloqueados por tu firewall. En otras circunstancias podremos ser capaces de observar algún mensaje de error que se pueda producir cuando estás conectando algún hardware nuevo o si tienes un servicio web instalado podrás ver quiénes están conectados a tu equipo.



Tipos MIME

Multipurpose Internet Mail Extensions o **MIME** (en español "extensiones multipropósito de correo de internet") son una serie de convenciones o especificaciones dirigidas al intercambio a través de Internet de todo tipo de archivos (texto, audio, vídeo, etc.) de forma transparente para el usuario. Una parte importante del MIME está dedicada a mejorar las posibilidades de transferencia de texto en distintos idiomas y alfabetos. En sentido general las extensiones de MIME van encaminadas a soportar: G

- Texto en conjuntos de caracteres distintos de US-ASCII;

- adjuntos que no son de tipo texto;
- cuerpos de mensajes con múltiples partes (multi-part);
- información de encabezados con conjuntos de caracteres distintos de ASCII.

Prácticamente todos los mensajes de correo electrónico escritos por personas en Internet y una proporción considerable de estos mensajes generados automáticamente son transmitidos en formato MIME a través de SMTP. Los mensajes de correo electrónico en Internet están tan cercanamente asociados con el SMTP y MIME que usualmente se les llama mensaje **SMTP/MIME**.

En 1991 la IETF (Grupo de Trabajo en Ingeniería de Internet, Internet Engineering Task Force en inglés) comenzó a desarrollar esta norma y desde 1994 todas las extensiones MIME están especificadas de forma detallada en diversos documentos oficiales disponibles en Internet.

MIME está especificado en seis *Request for Comments* o RFC (en español "solicitud de comentarios"): RFC 2045, RFC 2046, RFC 2047, RFC 4288, RFC 4289 y RFC 2077.

Los tipos de contenido definidos por el estándar MIME tienen gran importancia también fuera del contexto de los mensajes electrónicos. Ejemplo de esto son algunos protocolos de red tales como HTTP de la Web. HTTP requiere que los datos sean transmitidos en un contexto de mensajes tipo e-mail aunque los datos pueden no ser un e-mail propiamente dicho.

En la actualidad ningún programa de correo electrónico o navegador de Internet puede considerarse completo si no acepta MIME en sus diferentes facetas (texto y formatos de archivo).

Subtipos de Multiparte

El estándar MIME define varios subtipos para mensajes multiparte, estos especifican la naturaleza de la parte del mensaje y su relación con otras partes. El subtipo es especificado en el encabezado "Content-type" para todo el mensaje. Por ejemplo, un mensaje MIME multiparte que usa el subtipo *digest* tendrá un "Content-Type": "multipart/digest".

La RFC inicialmente define 4 subtipos: mixed, digest, alternate y parallel. Una aplicación que cumpla mínimamente el estándar debe soportar al menos mixed y digest; el resto de los subtipos son opcionales. Otras RFCs definen subtipos adicionales como: signed y form-data.

Lo que sigue es una lista de los subtipos más comúnmente usados:

Mixed

Multipart/mixed es usado para enviar mensajes o archivos con diferentes encabezados "Content-Type" ya sea en línea o como adjuntos. Si se envían imágenes u otros archivos fácilmente legibles, la mayoría de los clientes de correo electrónico las mostrarán como parte del mensaje (a menos que se especifique de manera diferente el encabezado "Content-disposition"). De otra manera serán ofrecidos como adjuntos. El content-type implícito para cada parte es "text/plain".

Message

Una parte `message/rfc822` contiene un mensaje de correo electrónico, incluyendo cualquier encabezado. *Rfc822* es un nombre erróneo, dado que el mensaje puede ser un mensaje MIME completo. Es usado también para resúmenes el reenviar mensajes.

Definido en la RFC 2046.

] Digest

Multipart/digest es una forma simple de enviar múltiples mensajes de texto. El content-type implícito para cada parte es "message/rfc822".

Definido en RFC 2046, Sección 5.1.5.

Alternative

El subtipo multipart/alternative indica que cada parte es una versión "alternativa" del mismo contenido (o similar), cada una en formatos diferentes denotados por su encabezado "Content-Type". Los formatos son ordenados atendiendo a cuán fieles son al original, con el menos fiel al inicio. Los sistemas pueden escoger la "mejor" representación que ellos son capaces de procesar; en general esta será la última parte que el sistema entiende, a menos que otros factores puedan afectar este comportamiento.

Related

El subtipo multipart/related es usado para indicar que las partes del mensaje no deben ser consideradas individualmente sino como agregados de un todo. El mensaje consiste de una parte raíz (implícitamente la primera) que hace referencia a otras partes, las que a su vez pueden hacer referencia a otras partes. Las partes son comúnmente referenciadas por el encabezado: "Content-ID". La sintaxis de la referencia no está especificada sino que está dictada por la codificación o el protocolo usado en la parte que contiene la referencia.

Un uso común de este subtipo es para enviar páginas web completas con imágenes en un único mensaje. La parte raíz contendría el documento HTML, que usaría etiquetas HTML para imágenes, para referirse a las imágenes almacenadas en partes subsiguientes.

Definido en RFC 2387

Report

Multipart/report es un tipo de mensaje que contiene datos formateados para que un servidor de correo lo interprete. Está entre un text/plain (o algún otro tipo de contenido fácilmente legible) y un message/delivery-status.

Definido en RFC 3462

Signed

El subtipo multipart/signed es usado para adjuntar una firma digital al mensaje. Esta tiene dos partes, una parte *cuerpo* y una parte *firma*. La parte del cuerpo completa, incluyendo los encabezados MIME, es usada para crear la parte de la firma. Existen muchos tipos de firmas, como application/pgp-signature y application/x-pkcs7-signature.

Protocolo HTTP

Definido en RFC 1847, Sección 2.1

Encrypted

Un mensaje multipart/encrypted tiene dos partes. La primera contiene información de control que es necesaria para descifrar la segunda parte, de tipo: application/octet-stream.

Definido en RFC 1847, Sección 2.2

Form Data

Como su nombre lo indica, multipart/form-data es usada para expresar valores enviados a través de un formulario. Originalmente definido como parte de HTML 4.0, es mayormente utilizado para enviar archivos vía HTTP.

Definido en RFC 2388

Mixed-Replace (Experimental)

El tipo de contenido multipart/x-mixed-replace fue desarrollado como parte de una tecnología para emular server push y streaming sobre HTTP.

Todas las partes de un mensaje mixed-replace poseen el mismo significado semántico. Sin embargo, cada parte invalida - "reemplaza" - a la parte previa tan pronto como es recibida completamente. Los clientes deben procesar la parte individual al momento de su llegada y no deben esperar a que termine el mensaje completo.

Desarrollado originalmente por Netscape, aún es soportado por Mozilla, Firefox, Safari (pero no en Safari para iPhone) y Opera, pero tradicionalmente ignorada por Microsoft.

Webdav

El objetivo de WebDAV es hacer de la World Wide Web un medio legible y editable, en línea con la visión original de Tim Berners-Lee. Este protocolo proporciona funcionalidades para crear, cambiar y mover documentos en un servidor remoto (típicamente un servidor web). Esto se utiliza sobre todo para permitir la edición de los documentos que sirve un servidor web, pero puede también aplicarse a sistemas de almacenamiento generales basados en web, que pueden ser accedidos desde cualquier lugar. La mayoría de los sistemas operativos modernos proporcionan soporte para WebDAV, haciendo que los ficheros de un servidor WebDAV aparezcan como almacenados en un directorio local.

Visión general acerca del protocolo Webdav

WebDAV añade los siguientes métodos a HTTP:

- PROPFIND - Usado para recuperar propiedades, almacenadas como XML, desde un recurso. También está sobrecargado para permitir recuperar la estructura de colección (alias jerarquía de directorios) de un sistema remoto.
- PROPPATCH - Usado para cambiar y borrar múltiples propiedades de un recurso en una simple operación atómica (atomic commit).
- MKCOL - Usado para crear colecciones (alias directorio)

Protocolo HTTP

- COPY - Usado para copiar un recurso desde un URI a otro.
- MOVE - Usado para mover un recurso desde un URI a otro.
- LOCK - Usado para bloquear (lock) un recurso. WebDAV soporta tanto bloqueos compartidos como exclusivos.
- UNLOCK - Para desbloquear un recurso.

Recurso es el nombre HTTP para una referencia que está apuntada por un Identificador de Recursos Uniforme o URI (Uniform Resource Identifier).

El grupo de trabajo WebDAV esta todavía trabajando en unas cuantas extensiones a WebDAV, incluyendo: control de redirecciones, enlaces, límites de espacio en disco y mejoras en la especificación base para que alcance el nivel de madurez del resto de estándares de Internet.

Navegadores web

Aplicación que opera a través de Internet, interpretando la información de archivos y sitios web para que podamos ser capaces de leerla, (ya se encuentre ésta alojada en un servidor dentro de la World Wide Web o en un servidor local).



El navegador interpreta el código, HTML generalmente, en el que está escrita la página web y lo presenta en pantalla permitiendo al usuario interactuar con su contenido y navegar hacia otros lugares de la red mediante enlaces o hipervínculos.



La funcionalidad básica de un navegador web es permitir la visualización de documentos de texto, posiblemente con recursos multimedia incrustados. Los documentos pueden estar ubicados en la computadora en donde está el usuario, pero también pueden estar en cualquier otro dispositivo que esté conectado a la computadora del usuario o a través de Internet, y que tenga los recursos necesarios para la transmisión de los documentos (un software servidor web).

Tales documentos, comúnmente denominados *páginas web*, poseen *hipervínculos* que enlazan una porción de texto o una imagen a otro documento, normalmente relacionado con el texto o la imagen.

El seguimiento de enlaces de una página a otra, ubicada en cualquier computadora conectada a la Internet, se llama *navegación*, de donde se origina el nombre *navegador* (aplicado tanto para el programa como para la persona que lo utiliza, a la cual también se le llama *cibernauta*). Por otro lado, *hojeador* es una traducción literal del original en inglés, *browser*, aunque su uso es minoritario.

Protocolo HTTP

Funcionamiento de un navegador web

La comunicación entre el servidor web y el navegador se realiza mediante el protocolo HTTP, aunque la mayoría de los navegadores soportan otros protocolos como FTP, Gopher, y HTTPS (una versión cifrada de HTTP basada en Secure Socket Layer o Capa de Conexión Segura (SSL)).

La función principal del navegador es descargar documentos HTML y mostrarlos en pantalla. En la actualidad, no solamente descargan este tipo de documentos sino que muestran con el documento sus imágenes, sonidos e incluso vídeos *streaming* en diferentes formatos y protocolos. Además, permiten almacenar la información en el disco o crear marcadores (*bookmarks*) de las páginas más visitadas.

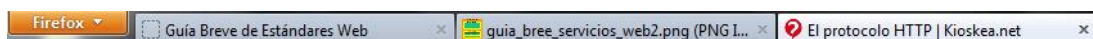
Algunos de los navegadores web más populares se incluyen en lo que se denomina una Suite. Estas Suite disponen de varios programas integrados para leer noticias de Usenet y correo electrónico mediante los protocolos NNTP, IMAP y POP.

Los primeros navegadores web sólo soportaban una versión muy simple de HTML. El rápido desarrollo de los navegadores web propietarios condujo al desarrollo de dialectos no estándares de HTML y a problemas de interoperabilidad en la web. Los más modernos (como Google Chrome, Amaya, Mozilla, Netscape, Opera e Internet Explorer 9.0) soportan los estándares HTML y XHTML (comenzando con HTML 4.01, los cuales deberían visualizarse de la misma manera en todos ellos).

Los estándares web son un conjunto de recomendaciones dadas por el World Wide Web consortium W3C) y otras organizaciones internacionales acerca de como crear e interpretar documentos basados en la web. Su objetivo es crear una web que trabaje mejor para todos, con sitios accesibles a mas personas y que funcionen en cualquier dispositivo de acceso a Internet.

Parámetros de apariencia y uso

Barra de Título

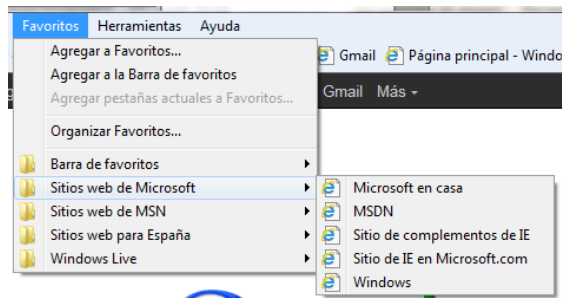


La barra de título muestra el título de la página y el nombre del navegador a la izquierda. A la derecha se hallan los botones estándar: Minimizar, Maximizar y Cerrar.

Protocolo HTTP

Barra de Menús

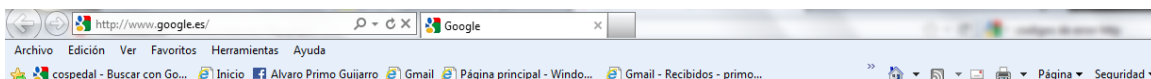
La Barra de Menús contiene las listas de comandos en cascada.



Barra de Herramientas

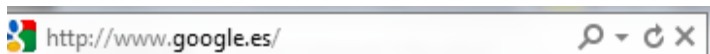
La barra de herramientas tiene botones para los comandos utilizados con más frecuencia. Cuando el ratón pasa por encima de un botón, este se verá en colores y parecerá en relieve. Algunos botones no se verán, si el tamaño de la ventana es pequeño.

Hasta que sepa la función de cada botón en la barra de herramientas, probablemente usted va a querer mostrar las etiquetas con texto.



Protocolo HTTP

Barra de Direcciones



La Barra de Direcciones muestra la URL (Universal Resource Location), también llamada dirección, ([address](#)), para las páginas web que se ven en la ventana del navegador. La barra de Vínculos generalmente se ve a la derecha de la barra de Direcciones.

Puede escribir una URL en la Barra de Direcciones y apretar la tecla ENTRAR para desplegar la página cuya ubicación ha escrito.



El botón Ir es agregado a la derecha de la Barra de Direcciones en IE5. Si prefiere más usar el ratón que el teclado, puede hacer un clic en el botón Ir, en lugar de apretar la tecla ENTRAR para abrir la página en la dirección que figura en la Barra de Direcciones.



Actual Página Web con Vínculos



La página web existente, se muestra en la parte de abajo de la ventana del navegador. El navegador ofrecerá barras de despliegue, si la página es demasiado ancha o muy alta para que quepa en la ventana.


Un **vínculo** hacia otra página web, imagen, o archivo, debería verse como extraordinaria. Un vínculo de texto por defecto, debería verse subrayado y el texto en color azul. Usted hace un clic en un link para apuntar a su objetivo en el navegador.

Protocolo HTTP

Barra de Estado

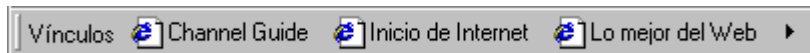


La Barra de Estado le contesta a usted. En su lado izquierdo verá mensajes sobre qué es lo que el navegador está haciendo. El mensaje más común es "Terminado"



 Listo, lo cual significa que el navegador cree que ha finalizado la carga de una página web.



Si su ratón pasa por encima de un vínculo, la dirección de ese vínculo aparecerá en la barra de estado. También hay iconos para mostrar el estado de su conexión. Habrá más detalles más adelante en la lección Paso a Paso.

Barra de Vínculos



La Barra de Vínculos, es un lugar conveniente para los atajos hacia las páginas web a las que accede con mayor frecuencia. IE ya viene con algunos sitios de Microsoft que se ven en la Barra de Vínculos. Según las diferentes versiones, se verán sitios algo distintos en la lista. Puede borrar aquellos sitios y agregar los suyos propios.

 A la derecha, puede ver los vínculos que no se muestran, desplegando la barra con un clic en  la flecha en el extremo derecho.

 Para ver vínculos que no se muestran, clic en  la doble flecha en el borde derecho de la Barra de Vínculos. Aparecerá una lista que desciende.

Seguridad del protocolo HTTP

Protocolo HTTPS

Hypertext Transfer Protocol Secure (HTTPS), es un protocolo de red basado en el protocolo HTTP, destinado a la transferencia segura de datos de



Protocolo HTTP

hipertexto, es decir, es la versión segura de HTTP. Es utilizado por cualquier tipo de servicio que requiera de envío de datos personales o contraseñas.

La idea del protocolo, es crear un **canal seguro** sobre una red insegura. Proporcionando seguridad frente ataques **eavesdropping y man in the middle**, siempre que tenga un método de cifrado adecuados y un certificado del servidor validos.

La confianza de este protocolo proviene de un servidor de autoridad de certificación que viene preinstalado en el software del navegador.

Una conexión HTTPS es validada cuando se cumple las siguientes condiciones:

- -El usuario confía en la Autoridad de certificación para websites legítimos
- -El website proporciona un certificado valido (en caso de fallo, la mayoría de los navegadores muestran un mensaje de alerta), lo que significa que esta firmado por una autoridad confiable.
- -El certificado identifica correctamente al website
- -Que el usuario confié en que la capa de cifrado del protocolo (TLS o SSL) es inquebrantable a ataques informáticos.

Para conocer si una página web utiliza el protocolo HTTPS, debemos observar si la dirección de nuestro navegador muestra la **sigla HTTPS** al comienzo en lugar de HTTP y que al final de la barra de direcciones aparezca un **candado** para indicarnos que el protocolo de comunicación es seguro

HTTPS utiliza un cifrado en **SSL/TLS** para crear un canal de cifrado más apropiado para el trafico de información sensible. De este modo consigue que esta información no pueda ser usada por un atacante que haya conseguido interceptar la transferencia de datos de la conexión, ya que lo único que obtendrá será un flujo de datos cifrados que le resultara imposible de descifrar. El puerto estándar del protocolo HTTPS es el 443

Diferencias a generales con el protocolo HTTP

A nivel de red el protocolo HTTP opera en la capa más alta del Modelo OSI, la **capa aplicación**; mientras que el protocolo HTTPS opera en una **subcapa más baja**, cifrando un mensaje HTTP previo a la transmisión y descifrando un mensaje una vez recibido.

Estrictamente hablando, HTTPS no es un protocolo separado, pero refiere el uso del HTTP ordinario sobre una Capa de conexión Segura cifrada: **Secure Sockets Layer (SSL)** o una conexión con Seguridad de la **Capa de Transporte (TLS)**.

Limitaciones

El protocolo HTTPS tiene algunas limitaciones:

- -El nivel de protección depende de la exactitud de la implementación del navegador web, el software del servidor y los algoritmos de cifrado actualmente soportados.
- -También, HTTPS es vulnerable cuando se aplica a contenido estático de publicación disponible. El sitio entero puede ser indexado usando una Araña web y la URI del

recurso cifrado puede ser adivinada conociendo solamente el tamaño de la petición/respuesta. Esto permite a un atacante tener acceso al Texto plano (contenido estatico de publicación), y al Texto cifrado la (versión cifrada del contenido estatico), permitiendo un ataque criptográfico.

- -Debido a que SSL opera bajo HTTP y no tiene conocimiento de protocolos de nivel más alto, los servidores SSL solo pueden presentar estrictamente un certificado para una combinación de puerto/IP en particular. Esto quiere decir, que en la mayoría de los casos, no es recomendable usar Hosting virtual name-based con HTTPS.

Existe una solución llamada Server Name indication (SNI) que envía el hostname al servidor antes de que la conexión sea cifrada, sin embargo muchos navegadores antiguos no soportan esta extensión. El soporte para SIN esta disponible desde Firefox 2, Opera 8 e Internet Explorer 7 sobre Windows Vista.

Conexiones seguras: SSL , TLS.

Secure Sockets Layer (SSL; en español «capa de conexión segura») y su sucesor **Transport Layer Security (TLS;** en español «seguridad de la capa de transporte») son protocolos criptográficos que proporcionan comunicaciones seguras por una red, comúnmente Internet.

El protocolo SSL intercambia registros; opcionalmente, cada registro puede ser comprimido, cifrado y empaquetado con un código de autenticación del mensaje (MAC). Cada registro tiene un campo de *content_type* que especifica el protocolo de nivel superior que se está usando.

Cuando se inicia la conexión, el nivel de registro encapsula otro protocolo, el protocolo *handshake*, que tiene el *content_type* 22.

El cliente envía y recibe varias estructuras *handshake*:

- Envía un mensaje *ClientHello* especificando una lista de conjunto de cifrados, métodos de compresión y la versión del protocolo SSL más alta permitida. Éste también envía bytes aleatorios que serán usados más tarde (llamados *Challenge de Cliente* o *Reto*). Además puede incluir el identificador de la sesión.
- Después, recibe un registro *ServerHello*, en el que el servidor elige los parámetros de conexión a partir de las opciones ofertadas con anterioridad por el cliente.
- Cuando los parámetros de la conexión son conocidos, cliente y servidor intercambian certificados (dependiendo de las claves públicas de cifrado seleccionadas). Estos certificados son actualmente X.509, pero hay también un borrador especificando el uso de certificados basados en OpenPGP.
- El servidor puede requerir un certificado al cliente, para que la conexión sea mutuamente autenticada.

Protocolo HTTP

- Cliente y servidor negocian una clave secreta (simétrica) común llamada *master secret*, posiblemente usando el resultado de un intercambio Diffie-Hellman, o simplemente cifrando una clave secreta con una clave pública que es descifrada con la clave privada de cada uno. Todos los datos de claves restantes son derivados a partir de este *master secret* (y los valores aleatorios generados en el cliente y el servidor), que son pasados a través una *función pseudoaleatoria* cuidadosamente elegida.

TLS/SSL poseen una variedad de medidas de seguridad:

- Numerando todos los registros y usando el número de secuencia en el MAC.
- Usando un resumen de mensaje mejorado con una clave (de forma que solo con dicha clave se pueda comprobar el MAC). Esto se especifica en el RFC 2104).
- Protección contra varios ataques conocidos (incluyendo ataques man-in-the-middle), como los que implican un degradado del protocolo a versiones previas (por tanto, menos seguras), o conjuntos de cifrados más débiles.
- El mensaje que finaliza el protocolo *handshake* (*Finished*) envía un *hash* de todos los datos intercambiados y vistos por ambas partes.
- La función pseudo aleatoria divide los datos de entrada en 2 mitades y las procesa con algoritmos hash diferentes (MD5 y SHA), después realiza sobre ellos una operación XOR. De esta forma se protege a sí mismo de la eventualidad de que alguno de estos algoritmos se revelen vulnerables en el futuro.

Gestión de certificados y acceso seguro con HTTPS

Hypertext Transfer Protocol Secure (ó HTTPS) es una combinación del protocolo HTTP y protocolos criptográficos. Se emplea para lograr conexiones más seguras en la WWW, generalmente para transacciones de pagos o cada vez que se intercambie información sensible (por ejemplo, claves) en internet.

De esta manera la información sensible, en el caso de ser interceptada por un ajeno, estará cifrada.

El nivel de protección que ofrece depende de la corrección de la implementación del navegador web, del software y de los algoritmos criptográficos soportados. Además HTTPS es vulnerable cuando es aplicado a contenido estático públicamente disponible.

El HTTPS fue creado por Netscape Communications en 1994 para su navegador Netscape Navigator.

Características del HTTPS

Para distinguir una comunicación o página web segura, la URL debe comenzar con "https://" (empleando el puerto 443 por defecto); en tanto la tradicional es "http://" (empleando el puerto 80 por defecto).

Originalmente HTTPS sólo utilizaba encriptación SSL, luego reemplazado por TLS.

Protocolo HTTP

HTTPS fue adoptado como estándar web por el grupo IETF tras la publicación del RFC 2818 en mayo de 2000.

HTTP opera en la capa más alta del modelo TCP/IP, la capa de Aplicación. Pero el protocolo de seguridad trabaja en una subcapa inferior, codificando el mensaje HTTP antes de ser transmitido y decodificando el mensaje antes de que llegue.

Adquiriendo Certificados

Adquirir certificados puede ser gratuito (generalmente sólo si se paga por otros servicios) o costar entre US\$13 y US\$1,500 por año.

Las organizaciones pueden también ser su propia autoridad de certificación, particularmente si son responsables de establecer acceso a navegadores de sus propios sitios (por ejemplo, sitios en una compañía intranet, o universidades mayores). Estas pueden fácilmente agregar copias de su propio certificado firmado a los certificados de confianza distribuidos con el navegador.

También existen autoridades de certificación peer-to-peer.

Usar un Control de Acceso

El sistema puede también ser usado para la Autenticación de clientes con el objetivo de limitar el acceso a un servidor web a usuarios autorizados. Para hacer esto, el administrador del sitio típicamente crea un certificado para cada usuario, un certificado que es guardado dentro de su navegador. Normalmente, este contiene el nombre y la dirección de correo del usuario autorizado y es revisado automáticamente en cada reconexión para verificar la identidad del usuario, potencialmente sin que cada vez tenga que ingresar una contraseña.

Almacenamiento virtual de sitios web “HOSTS VIRTUALES”

El término *Hosting Virtual* se refiere a hacer funcionar más de un sitio web (tales como `www.company1.com` y `www.company2.com`) en una sola máquina. Los sitios web virtuales pueden estar "basados en direcciones IP", lo que significa que cada sitio web tiene una dirección IP diferente, o "basados en nombres diferentes", lo que significa que con una sola dirección IP están funcionando sitios web con diferentes nombres (de dominio). El hecho de que estén funcionando en la misma máquina física pasa completamente desapercibido para el usuario que visita esos sitios web.

Apache fue uno de los primeros servidores web en soportar hosting virtual basado en direcciones IP. Las versiones 1.1 y posteriores de Apache soportan hosting virtual (vhost)

basado tanto en direcciones IP como basado en nombres. Ésta última variante de hosting virtual se llama algunas veces *basada en host* o *hosting virtual no basado en IP*.

Alojamiento virtual basado en IPS

También llamado IP dedicado o virtual hosting, cada máquina virtual tiene una dirección IP diferente. El servidor Web está configurado con múltiples interfaces de red física, o interfaces de red virtual en la misma interfaz física. El software del servidor web, utiliza la dirección IP del cliente se conecta con el fin de determinar a qué sitio web para mostrar al usuario. La razón principal de un sitio para que utilice una IP dedicada debe ser capaz de utilizar su propio certificado SSL en lugar de un certificado común.

El hosting virtual basado en IPs usa la dirección IP de la conexión para determinar qué host virtual es el que tiene que servir. Por lo tanto, necesitará tener diferentes direcciones IP para cada host. Si usa hosting virtual basado en nombres, el servidor atiende al nombre de host que especifica el cliente en las cabeceras de HTTP. Usando esta técnica, una sola dirección IP puede ser compartida por muchos sitios web diferentes.

Alojamiento virtual basado en nombres

El hosting virtual basado en nombres es normalmente más sencillo, porque solo necesita configurar su servidor de DNS para que localice la dirección IP correcta y entonces configurar Apache para que reconozca los diferentes nombres de host. Usando hosting virtual basado en nombres también se reduce la demanda de direcciones IP, que empieza a ser un bien escaso. Por lo tanto, debe usar hosting virtual basado en nombres a no ser que haya alguna razón especial por la cual tenga que elegir usar hosting virtual basado en direcciones IP. Algunas de estas razones pueden ser:

- Algunos clientes antiguos no son compatibles con el hosting virtual basado en nombres. Para que el hosting virtual basado en nombres funcione, el cliente debe enviar la cabecera de Host HTTP. Esto es necesario para HTTP/1.1, y está implementado como extensión en casi todos los navegadores actuales. Si necesita dar soporte a clientes obsoletos y usar hosting virtual basado en nombres, al final de este documento se describe una técnica para que pueda hacerlo.
- El hosting virtual basado en nombres no se puede usar junto con SSL por la naturaleza del protocolo SSL.
- Algunos sistemas operativos y algunos elementos de red tienen implementadas técnicas de gestión de ancho de banda que no pueden diferenciar entre hosts a no ser que no estén en diferentes direcciones IP.

Alojamiento virtual basado en puertos

Basado en puerto

El número de puerto por defecto para HTTP es 80. Sin embargo, la mayoría de servidores web se puede configurar para funcionar en casi cualquier número de puerto, siempre que el número de puerto no está en uso por cualquier otro programa en el servidor.

Por ejemplo, un servidor puede alojar el sitio web `www.example.com`. Sin embargo, si el propietario desea operar un segundo sitio, y no tiene acceso a la configuración del nombre de dominio para su nombre de dominio y / o no posee otras direcciones IP que pueden ser utilizados para servir el sitio de, en su lugar podría utilizar otro número de puerto, por ejemplo, `www.example.com:81` para el puerto 81, `www.example.com:8000` para el puerto 8000, o `www.example.com:8080` para el puerto 8080.

Sin embargo, este es un enfoque de usuario poco amigable. Los usuarios no se puede esperar razonablemente que saber los números de puerto para sus sitios web y móvil de un sitio entre los servidores puede requerir cambiar el número de puerto. No se usen los números de puerto estándar también puede ser visto como poco profesional y poco atractivo para los usuarios. Además, algunos firewalls bloquear todos los puertos, pero la más común, provocando un sitio alojado en un puerto no estándar que no aparecen disponibles para algunos usuarios.

Alojamientos híbridos

Por medio de un software simulamos dividir una computadora en cuatro o en cinco computadoras. Así, cada servidor virtual trabaja como si fuera una computadora independiente con un alojamiento dedicado. La diferencia con los servidores compartidos es que en éstos sólo abrimos carpetas en el disco duro para las diferentes páginas.

No son tan baratos como los compartidos, ni tan caros como los dedicados. Sin tantas ventajas técnicas como éstos últimos, pero sin tantos inconvenientes como los primeros. Una buena elección intermedia.

Usos

Puente de servidores privados virtuales la brecha entre los servicios de alojamiento web compartido y hosting dedicado, lo que la independencia de otros clientes del servicio de VPS en términos de software, pero a menor costo que un servidor dedicado físico. Como VPS ejecuta su propia copia de su sistema operativo, los clientes tienen superusuario nivel de acceso a esa instancia del sistema operativo, y se puede instalar casi cualquier software que se ejecuta en el sistema operativo. Cierta tipo de software no funciona bien en un entorno virtualizado, como virtualizers sí mismos, algunos proveedores de VPS imponer mayores restricciones, pero en general son laxas en comparación con los entornos de alojamiento compartido. Debido a la cantidad de clientes de virtualización generalmente se ejecuta en una

sola máquina, un VPS en general ha limitado el tiempo de procesador, memoria RAM y espacio en disco.

Servidor privado virtual hosting

Un número creciente de empresas que le ofrecen alojamiento virtual de servidores privados, o un servidor virtual de hosting dedicado, como una extensión de web hosting servicios. Hay varios desafíos a tener en cuenta cuando las licencias de software propietario en entornos de múltiples usuarios virtuales.

Administrados de alojamiento

El cliente se deja de controlar y administrar su propio servidor.

Unmetred hosting

Este tipo de servicio generalmente se ofrece sin límite en la cantidad de datos transferidos sobre una línea de ancho de banda fijo. Por lo general, no medido de alojamiento se ofrece con 10 Mbit / s, 100 Mbit / s ó 1000 Mbit / s (con algunas de hasta 10 Gbit / s). Esto significa que el cliente es teóricamente capaz de utilizar 3,33 TB ~ el 10 Mbit / s, 33 TB de ~ 100 Mbit / s y 333 TB ~ en una línea de 1.000 Mbit / s por mes (aunque en la práctica de los valores será significativamente menor).

El software de virtualización

Para algunos de los paquetes de software de uso común para proporcionar la plataforma de virtualización , vea la comparación de la plataforma de máquinas virtuales

Nube de servidor

Un VPS que es dinámico (es decir, se puede cambiar en tiempo de ejecución) se refiere a menudo como un servidor de nube. Atributos clave para esto son:

- Los recursos de hardware adicionales se pueden agregar en tiempo de ejecución (CPU, RAM)
- Servidor puede ser trasladado a otro hardware, mientras que el servidor está en ejecución (de forma automática de acuerdo a la carga en algunos casos)