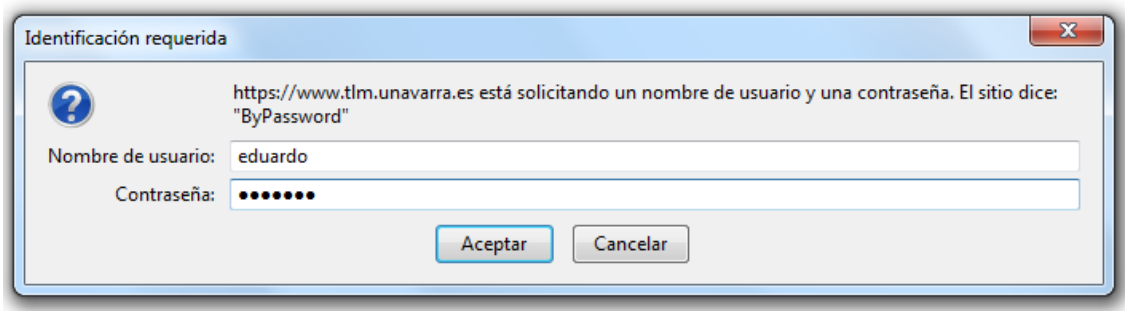
Índice hora 3

Objetivos

- Describir el esquema de autenticación básica de usuarios HTTP
- Presentar la forma de “emular” un seguimiento de usuarios mediante el empleo de cookies
- Describir esquemas de mejora: cachés
- Identificar la interacción de un servidor web con aplicaciones en el lado del servidor: CGI
- Presentar lo servidores de pasarela web: proxy
- Cuestionar la seguridad de la web: HTTPs

2.4 Autenticación básica de usuarios HTTP

- La RFC 2617 define un procedimiento básico para la identificación de usuarios web:
  - Paso 1: el navegador solicita un recurso de forma habitual
  - Paso 2: el servidor responde indicando que el recurso está protegido
  - Paso 3: el navegador solicita al usuario que introduzca un identificador y una contraseña
  - Paso 4: el navegador repite la solicitud al servidor, incluyendo la identificación del usuario
  - Paso 5: el servidor verifica la identificación y responde con el recurso solicitado



Autenticación básica de usuarios HTTP

- Cabecera WWW-authenticate:
  - Tipo de autenticación: basic (login/password).
  - Realm: dominio (grupo) de usuarios bajo el cual se concede acceso al recurso.

Paso 2
HTTP/1.1 401 Unauthorized Server : Netscape−FastTrack/2.0 c Date : Sun, 17 Nov 1999 16:42:46 GMT WWW−authenticate:basic realm=”Default UserDB” Content−type : text/html Content−length : 223 <HTML><HEAD><TITLE>Unauthorized </TITLE></HEAD>

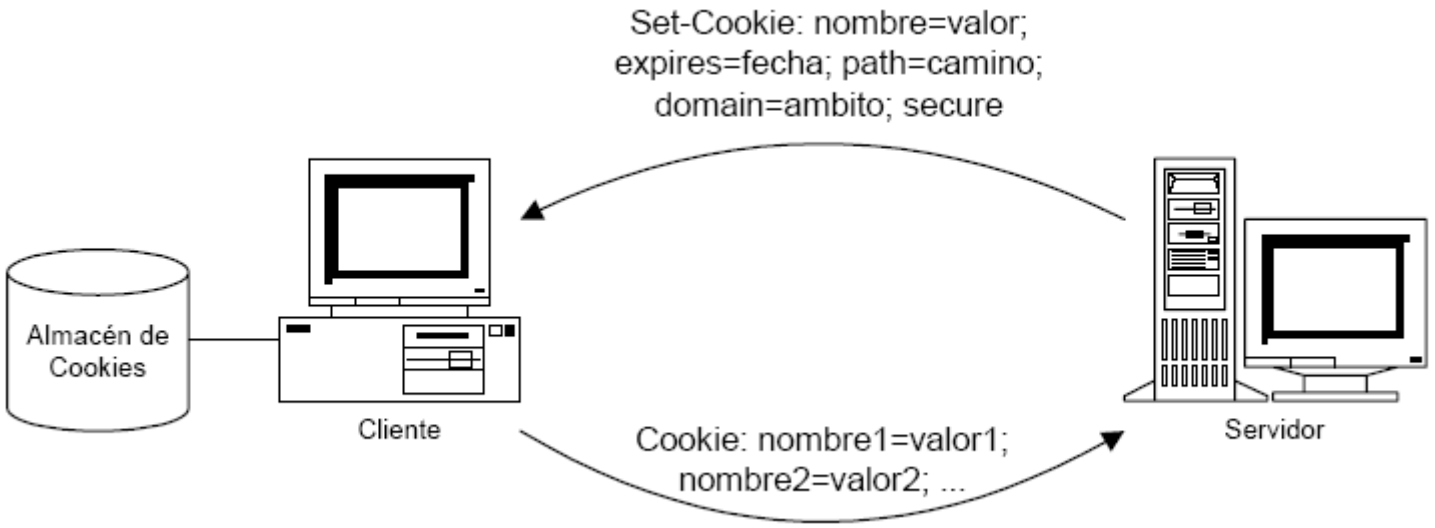
Autenticación básica de usuarios HTTP

- Cabecera Authorization:
  - Tipo de autenticación: basic.
  - Cadena usuario: contraseña codificada en Base64.

Paso 4
GET /auth/index.html HTTP/1.1 Authorization : basic KDENfKdifwekFD23nf== Connection : Keep−Alive User−Agent : Mozilla/4.06 Host : www.unavarra.edu Accept : image/gif , image/x−xbitmap , image/jpeg , . . .

2.5 Seguimiento de estado

- Una cookie es un dato (de formato conocido por la aplicación que lo genera o usa) que un servidor puede enviar al cliente para que éste la almacene y la reenvíe en posteriores accesos al mismo servidor.
- Las cookies se emplean para recordar preferencias de un cliente y ”simular” sesiones, dado que el protocolo HTTP carece de ellas.
- RFC 2109 define cabeceras HTTP para cookies.



Seguimiento de estado

- El servidor HTTP inicializa una cookie insertando la cabecera de respuesta Set-Cookie:
  - nombre=valor : asocia a una propiedad un valor específico. Espacio, coma y punto y coma se traducen al código URL.
  - expires=fecha : fecha de caducidad de la cookie. Formato: weekday, DD-Month-YY HH:MM:SS GMT. Por defecto, una cookie expira al cerrar el navegador.
  - domain=ámbito : identifica los servidores a los que el cliente debe enviar el cookie. Por defecto, sólo la envía al servidor que genera la cookie.
  - path=camino : especifica los recursos a los que se envía la cookie. Por defecto, el CGI que genera la cookie.
  - secure : sólo se envía cuando se emplea el protocolo HTTPS
- Un servidor puede enviar varias Set-Cookie en una respuesta.
- Una cookie se puede eliminar indicando una fecha pasada.

Seguimiento de estado

- Devolución de una cookie por parte del cliente:
  - Al conectarse a un servidor web, el navegador comprueba si dispone de alguna cookie con domain y path que verifiquen el URL consultado.
  - Con las cookies que verifiquen el URL, el navegador inserta una cabecera de petición del formato:

    Cookie: nombre1=valor1; nombre2=valor2; ...
- Cookies y privacidad:
  - Las cookies permiten a los sitios aprender muchas cosas sobre usted
  - Las motores de búsqueda utilizan redireccionamiento & las cookies para aprender aún más
  - Las compañías de publicidad obtienen información a través de los sitios web

2.6 Caché

- Un navegado guarda los últimos objetos descargados en una caché (normalmente en disco duro) para usarlos si son necesarios en un futuro cercano, evitando la descarga correspondiente
- Para evitar la transferencia de recursos web, HTTP permite al cliente comprobar si un objeto almacenado en caché ha sido modificado
- La técnica se conoce como GET condicional: mediante la cabecera If-Modified-Since, un cliente solicita un recurso sólo en el caso de que no se haya modificado

Petición de un recurso que no está en la caché
GET /imagen.gif HTTP/1.0 User−agent : Mozilla/4.0

Caché

- El servidor envía un mensaje de respuesta con la cabecera Last-Modified, que se asocia al objeto en la caché del cliente.

Respuesta del servidor con el recurso solicitado
HTTP/1.0 200 OK Date : Wed, 12 Aug 1998 15:39:29 Server : Apache/1.3.0 (Unix) Last−Modified : Mon, 22 Jun 1998 09:23:24 Content−Type : image/gif [CRLF] (data)

Caché

- Una nueva solicitud del mismo objeto incluye la cabecera If-Modified-Since con la fecha de última modificación del objeto en caché.

Petición de un recurso que está en caché
GET /imagen.gif HTTP/1.0 User−agent : Mozilla/4.0 If−modified−since : Mon, 22 Jun 1998 09:23:24

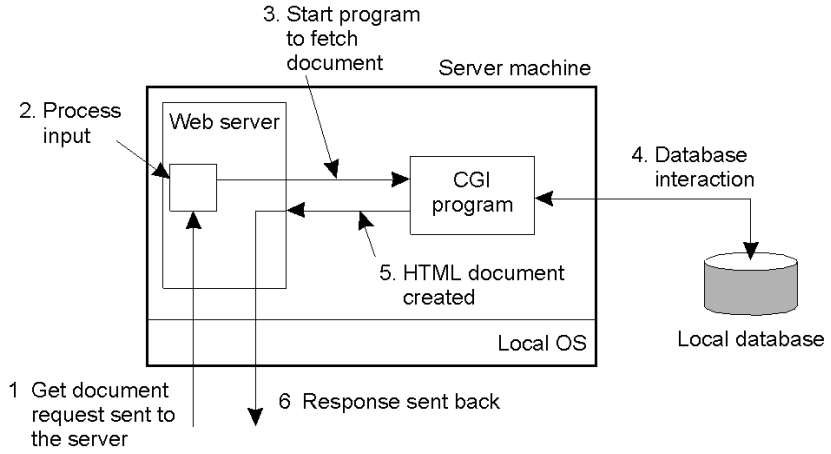
Respuesta sin cuerpo en caso de que no se haya modificado
HTTP/1.0 304 Not Modified Date : Wed, 19 Aug 1998 15:39:29 Server : Apache/1.3.0 (Unix)

2.7 CGIs

- Common Gateway Interface (CGI): El interfaz CGI define un mecanismo de comunicación estándar entre un servidor web y una aplicación externa.
  - 1. El usuario del cliente web recibe una página HTML con un formulario o hiperenlaces cuyo URL contiene datos.
  - 2. El usuario pulsa en un botón SUBMIT de un formulario, o selecciona un hiperenlace con datos.
  - 3. El programa navegador codifica los datos antes de enviarlos al servidor.
  - 4. El navegador emplea el método GET o el método POST para enviar los datos al servidor, de acuerdo con lo especificado en la definición HTML del formulario.
  - 5. El servidor reconoce que están solicitando la ejecución de un programa CGI, e invoca a dicho programa un nuevo proceso.

CGIs

- 6. El programa CGI recibe los datos enviados desde el cliente a través de variables de entorno, parámetros de la línea de comando o la entrada estándar.
- 7. El programa CGI procesa la solicitud y genera la respuesta en forma de página web a través de la salida estándar. Parte de los campos de la cabecera HTTP de respuesta los proporciona el propio programa CGI.
- 8. El servidor web envía al cliente web los resultados que emite el programa gateway. Normalmente completa la cabecera de respuesta.
- 9. El cliente web visualiza la página web que contiene los resultados.



CGIs – variables de entorno

Un programa CGI puede recibir información del servidor a través de variables de entorno:

- Con información relativa al propio servidor:
  - HTTP HOST: nombre del servidor
  - SERVER PORT: puerto usado por el servidor
  - DOCUMENT ROOT: directorio con las páginas HTML
- Con información sobre la conexión y la petición HTTP:
  - REMOTE ADDR: dirección IP del cliente
  - HTTP COOKIE: cookies enviadas por el cliente
  - REQUEST METHOD: método HTTP de la petición
  - QUERY STRING: cadena de caracteres, en formato URL, que aparece en la petición GET. Corresponde a los caracteres del URL posteriores al símbolo ?
  - CONTENT TYPE: tipo MIME del cuerpo de la petición
  - CONTENT LENGTH: tamaño del cuerpo de la petición

CGIs – GET/POST

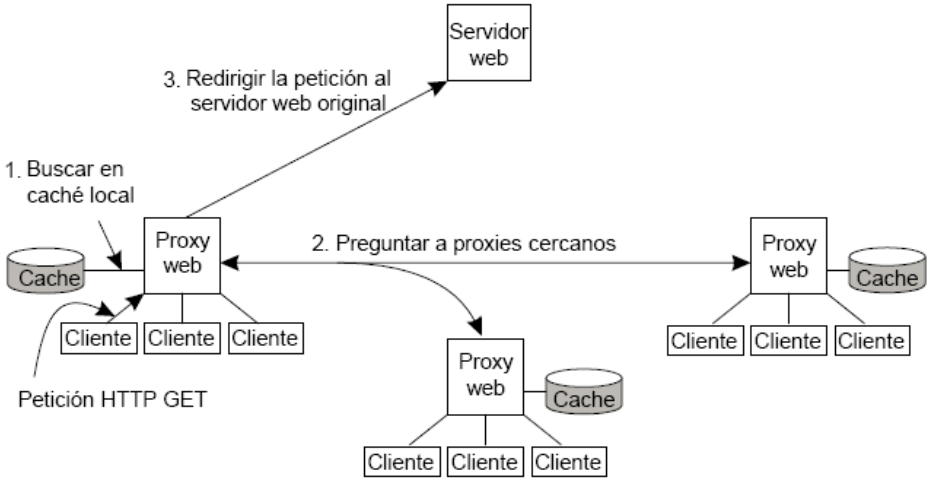
- El método GET:
  - Los datos son enviados en el campo del URL de la línea de solicitud
  - Ventaja: el usuario puede conservar URLs que representan consultas a CGIs con determinados parámetros.
  - Inconveniente: la cadena de caracteres que define la codificación URL puede ser larga y sobrepasar el tamaño máximo tolerado por el servidor.

```
www.algunsitio.com/busqueda?nombre=arcesio&apellido=net
```

- El método POST:
  - Los datos son enviados al servidor a través del cuerpo del mensaje
  - Ventaja: el método no se ve restringido por el tamaño de los datos producidos por un formulario.
  - Inconveniente: el usuario no puede conservar las consultas realizadas con este método.

2.8 Proxy web

- Un servidor proxy web es una entidad de red a la que los clientes delegan la realización de solicitudes HTTP.
- Ventajas del uso de proxys:
  - Reducción del tiempo de respuesta.
  - Reducción del tráfico web a través de la red de acceso a Internet, y por tanto reducción del coste de conexión.
  - Seguridad: control de recursos accedidos y revisión de los mismos (por ejemplo, comprobación antivirus)



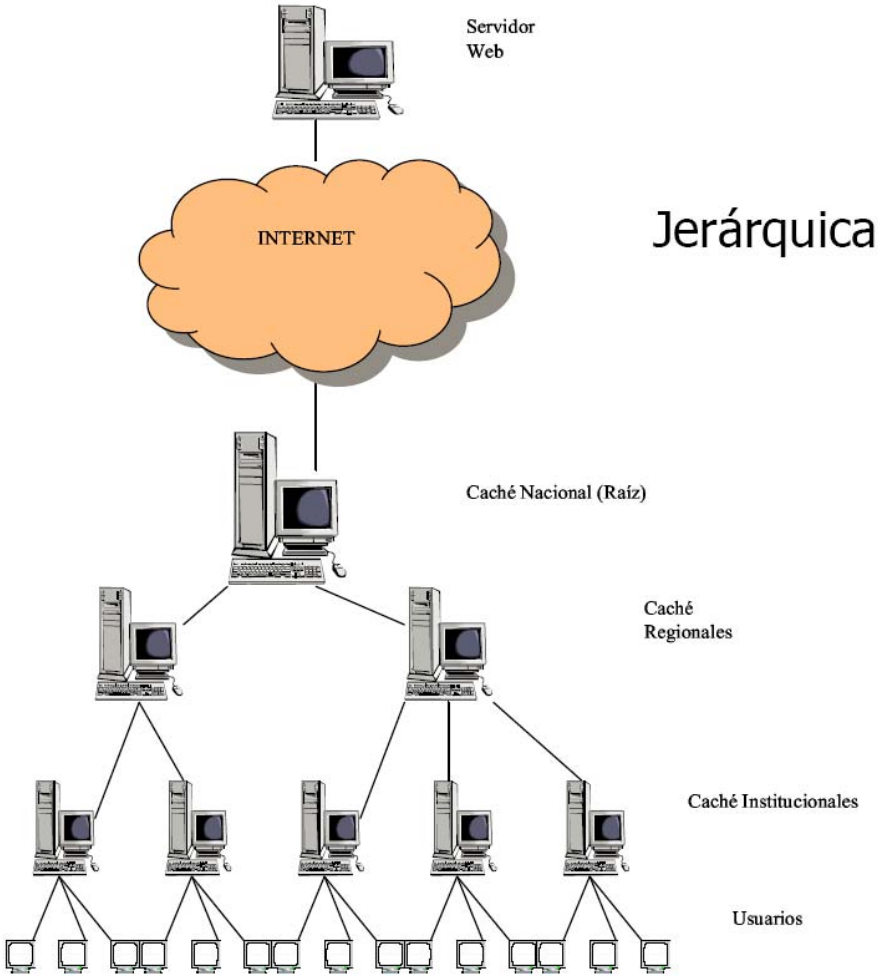
Proxy web

Secuencia de uso de un proxy:

- 1. El navegador establece una conexión con el proxy y le envía la solicitud HTTP. La cabecera Host indica el servidor original.
- 2. El proxy comprueba si tiene una copia del objeto almacenada localmente. En tal caso, le envía el objeto al navegador en un mensaje HTTP de respuesta.
- 3. Si el proxy no tiene el objeto, abre una conexión con el servidor original y le envía una petición HTTP del objeto requerido por el navegador. El servidor original envía la respuesta al servidor proxy.
- 4. El proxy almacena una copia en su caché y redirige otra copia en una respuesta HTTP al navegador, por la conexión establecida en el primer paso.

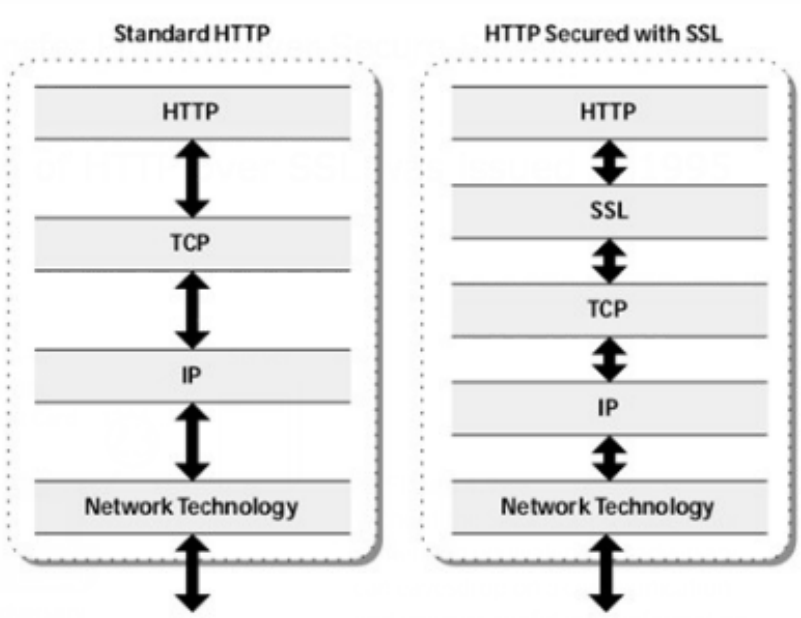
Proxy web, jerarquía

- Se pueden establecer servidores proxy cooperativos que aumentan la probabilidad de disponer de una copia cercana
- **Múltiples cachés** web
  - caché cooperativa
  - topología jerárquica
  - protocolo *squid*
- Árbol de distribución de contenidos



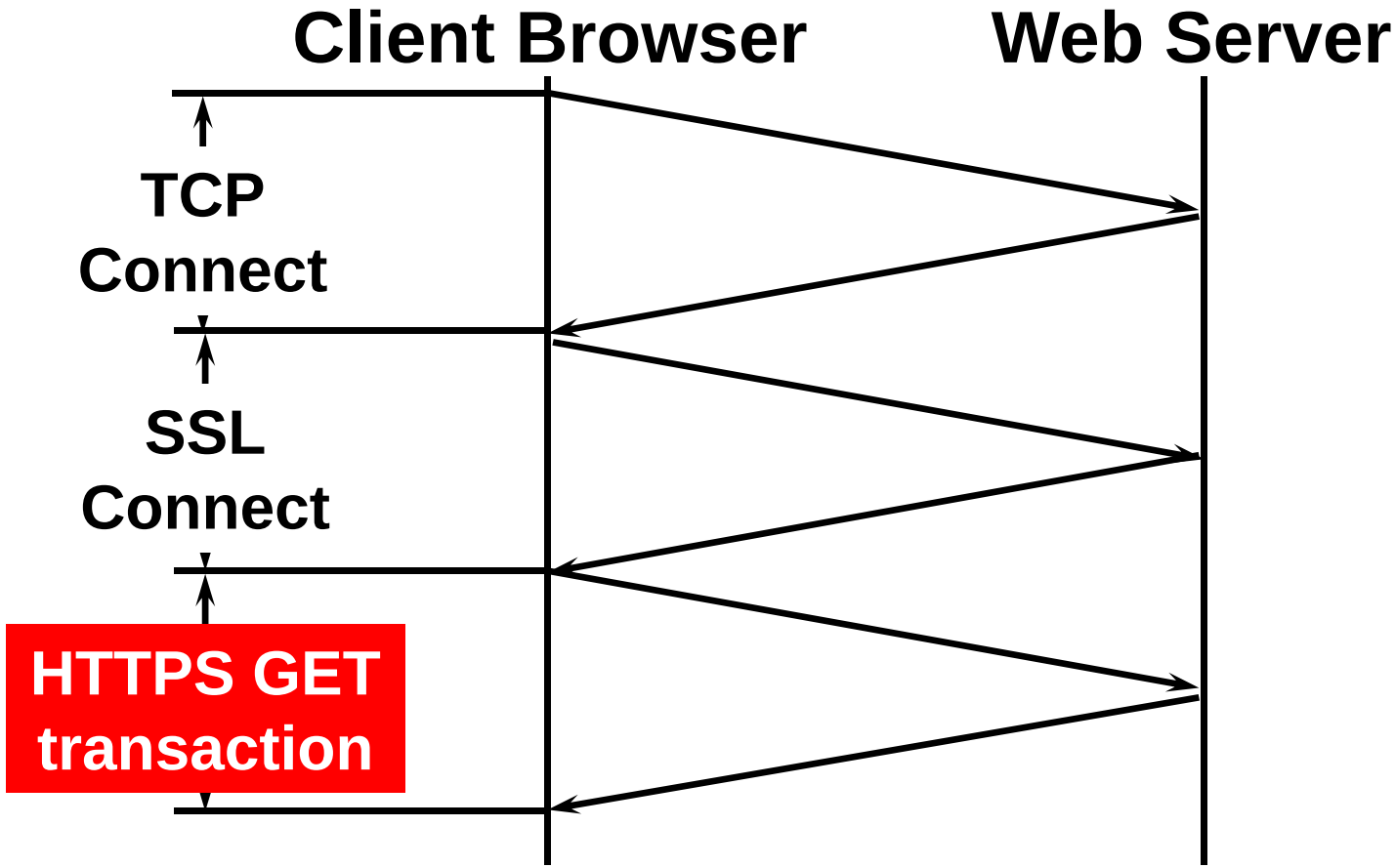
2.9 HTTP seguro

- HTTP no provee seguridad
- HTTP puede correr sobre Secure Socket Layer (SSL), que se referencia como HTTPS y provee mecanismos de:
  - Confidencialidad: cifra los datos
  - Autenticación: verificar la identidad de servidor o/y cliente
  - Integridad: verificar la no modificación de los datos



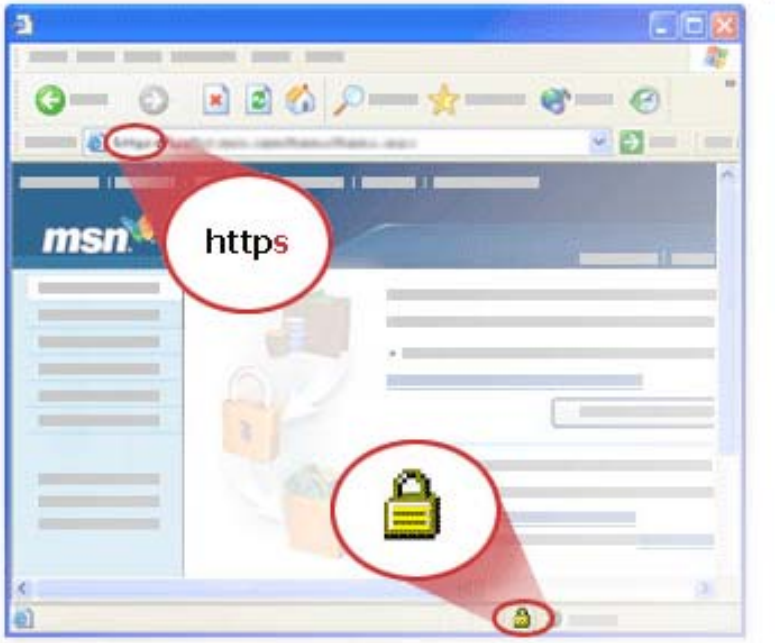
HTTP seguro

- Coste extra
  - Computacional: encriptar/desencriptar todos los datos
  - En el establecimiento



HTTP seguro

- Utilizar el puerto 443 TCP
- Necesidad de utilizar HTTPS para acceso a datos privados, login con contraseña, etc.



Resumen

- HTTP es un protocolo simple gracias en gran parte a ser un protocolo sin estado
  - Sin embargo, las web necesita seguir el estado de las sesiones de sus clientes para ofrecer muchas funcionalidades
  - Las cookies son la solución
- Las cachés y proxys permiten reducir el tráfico por la red
- Además, un proxy permite mejorar la experiencia del usuario a nivel de velocidad y seguridad
- Un servidor web puede ejecutar aplicaciones externas vía un interfaz CGI
- HTTP es un protocolo no seguro.
  - Es necesario encapsularlo sobre SSL para pasar a ser HTTPS y ofrecer un protocolo mínimamente seguro

Referencias

- RFC1945 (HTTP/1.0) y RFC2616 (HTTP/1.1)
- [Forouzan]
  - Capítulo 22, “World Wide Web and HTTP”, sección 22.3
- [Kurose]
  - Capítulo 2, sección 2.2
- David Gourley, Brian Totty, Marjorie Sayer, Anshu Aggarwal, Sailu Reddy. HTTP: The Definitive Guide. ISBN 978-1565925090. O'Reilly Media. September 2002.