Configurando SSL/TLS

Hacia la seguridad real...

Miguel Macías Enguídanos miguel.macias@upv.es



Bilbao, 27/11/2012



Índice

- Introducción
- Ejemplos actuales
- Opinión generalizada
- Situación ideal
 - confidencialidad
 - integridad
 - disponibilidad
 - autenticidad
- La web: un caso especial

Esquema Nacional de Seguridad

- ENS (Real Decreto 3/2010)
 - Preámbulo

La finalidad del Esquema Nacional de Seguridad es la creación de las condiciones necesarias de confianza en el uso de los medios electrónicos, a través de **medidas para** garantizar la seguridad de los sistemas, los datos, las comunicaciones, y los servicios electrónicos, que permita a los ciudadanos y a las Administraciones públicas, el ejercicio de derechos y el cumplimiento de deberes a través de estos medios.

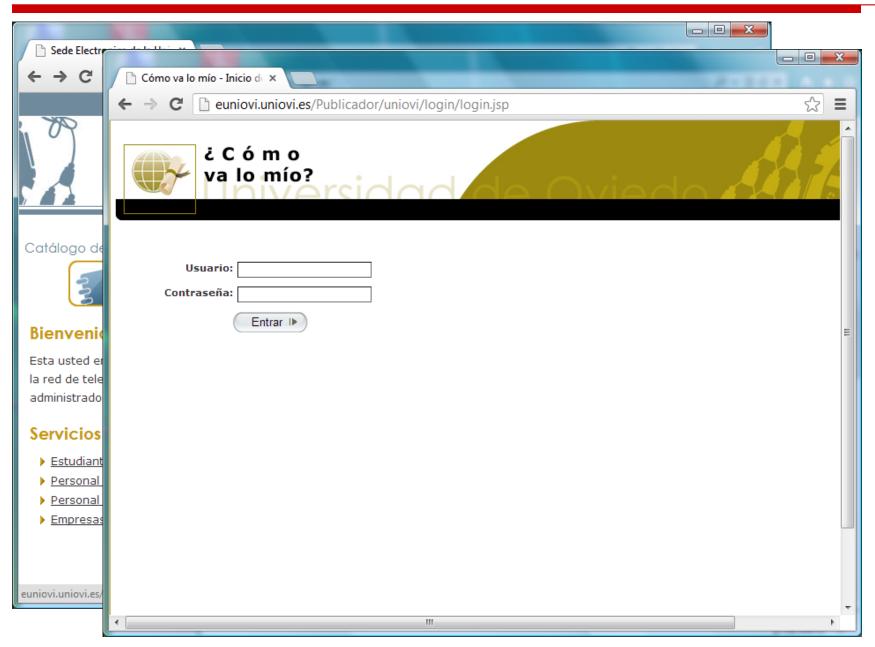
Estado actual

- De todos los cambios que implica el ENS, puede parecernos que la parte más sencilla es asegurar las comunicaciones
- SSL/TLS es sencillo de implementar en cualquier sistema y, de hecho, es algo que ya tenemos (prácticamente) todos en marcha
- Incluso la mayoría de usuarios reconocen el uso de SSL y lo "exigen" (en algunos servicios, al menos)
- ¿Podemos estar tranquilos? ¿lo hacemos bien? ¿estamos generando confianza? ¿mejor tener SSL, sea como sea, que no tener nada?

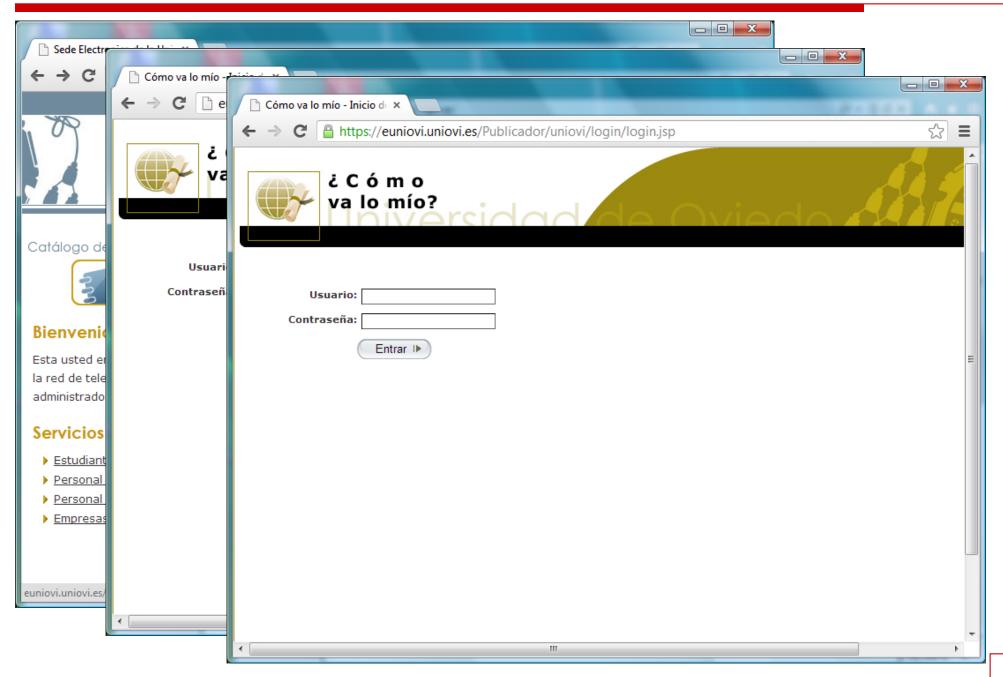
Ejemplo 1: sin seguridad (1/3)



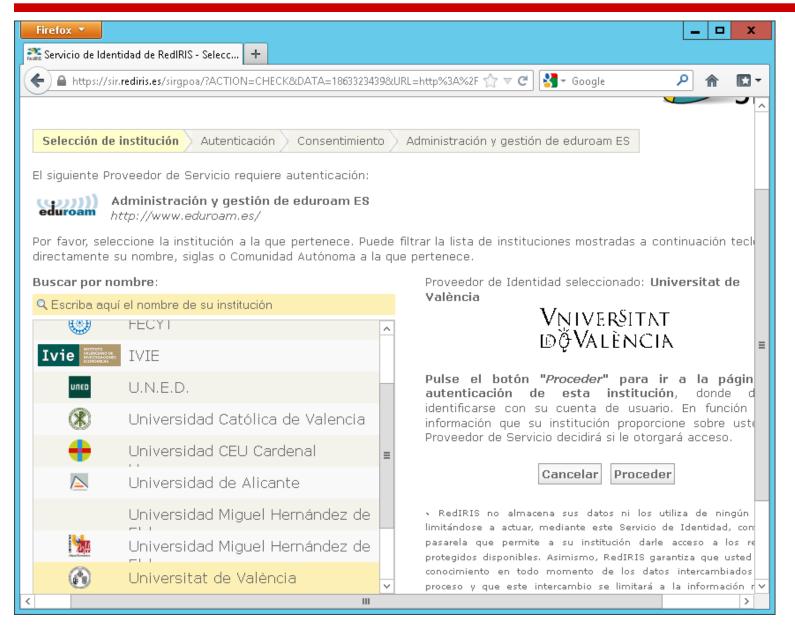
Ejemplo 1: sin seguridad (2/3)



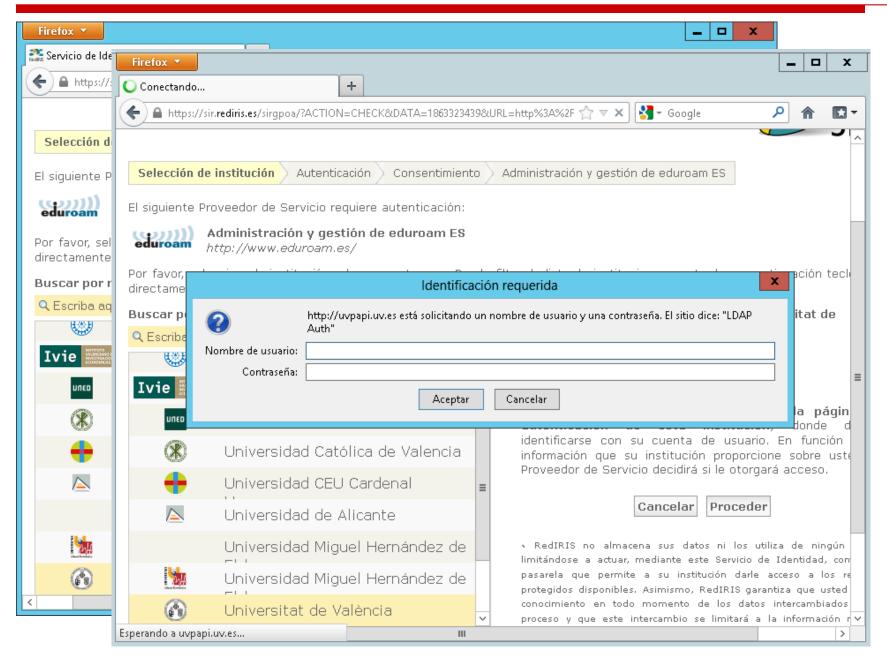
Ejemplo 1: sin seguridad (3/3)



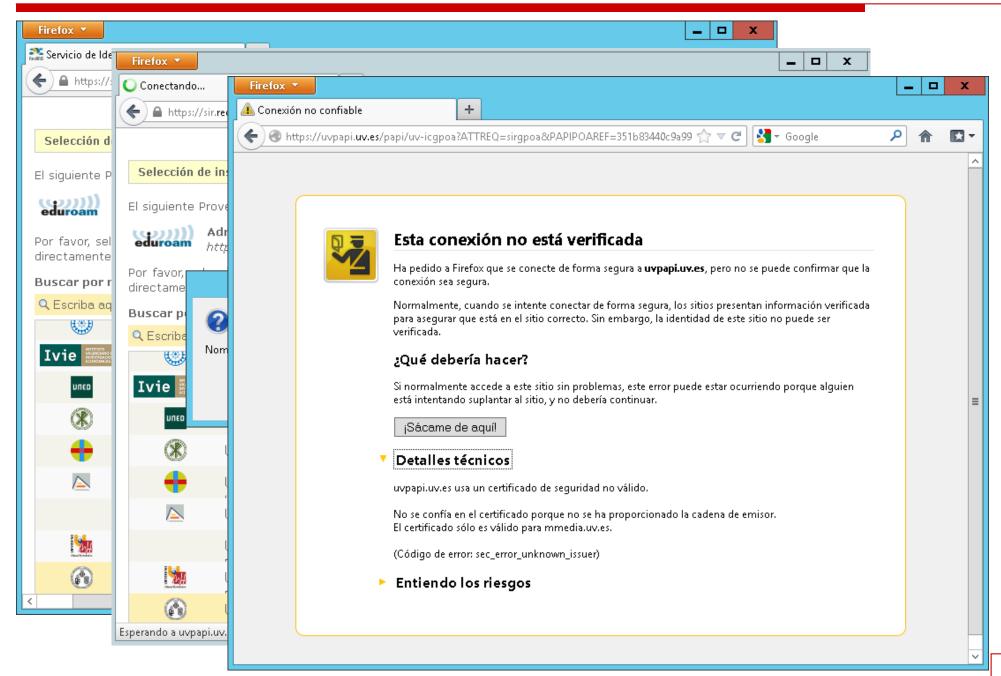
Ejemplo 2: sin seguridad (1/3)



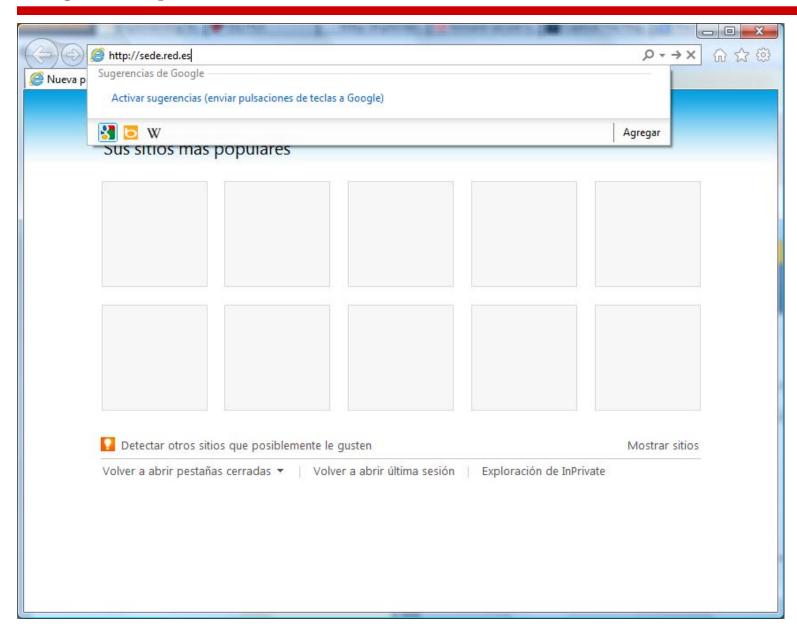
Ejemplo 2: sin seguridad (2/3)



Ejemplo 2: sin seguridad (3/3)



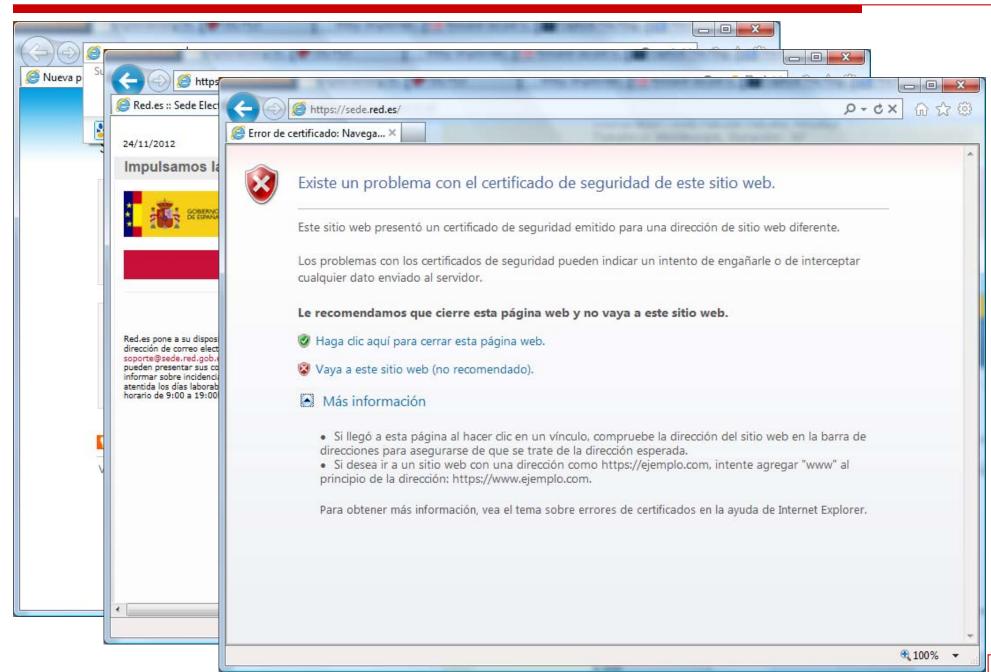
Ejemplo 3: mareando al usuario (1/3)



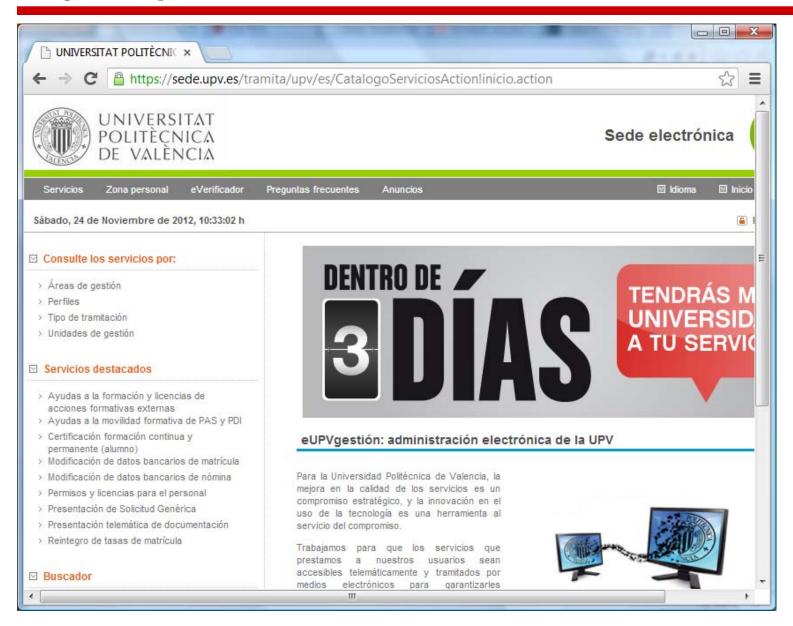
Ejemplo 3: mareando al usuario (2/3)



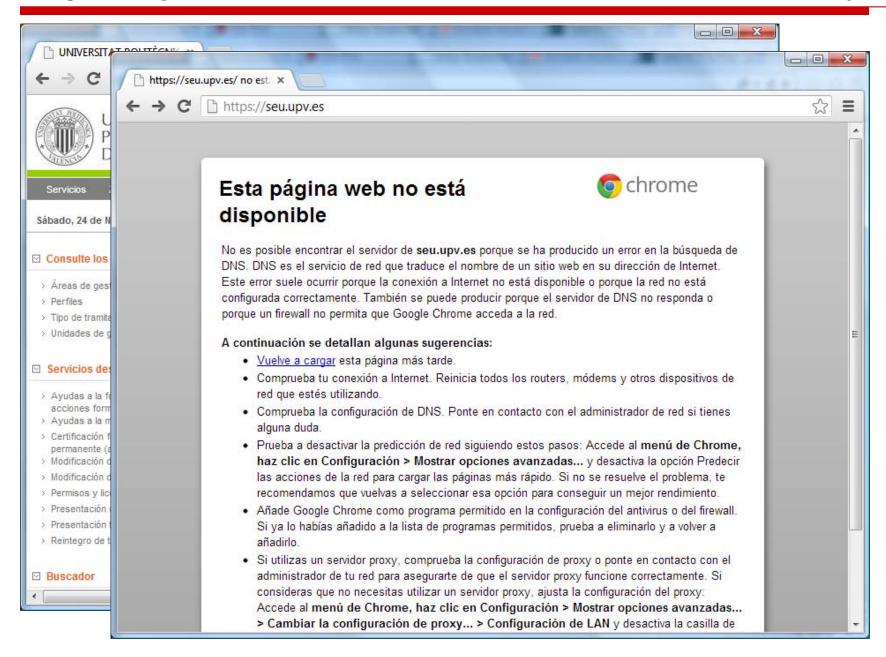
Ejemplo 3: mareando al usuario (3/3)



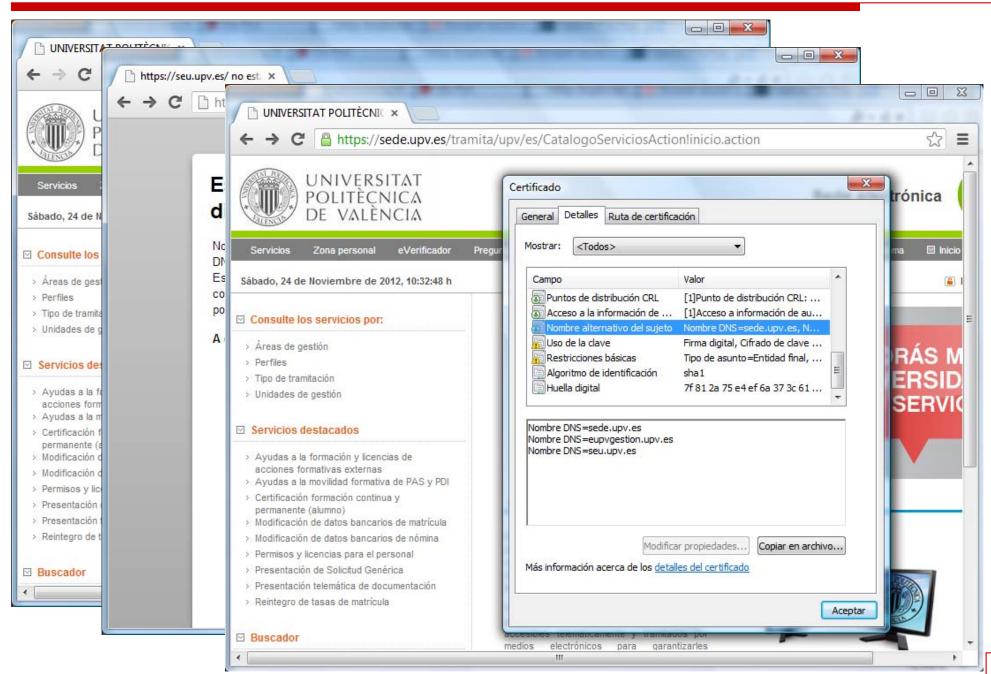
Ejemplo 4: mareando al usuario (1/3)



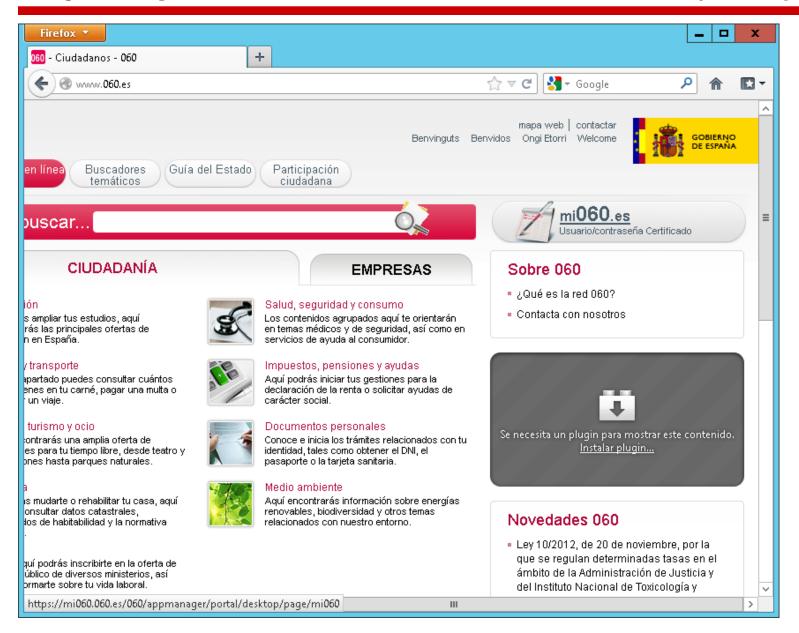
Ejemplo 4: mareando al usuario (2/3)



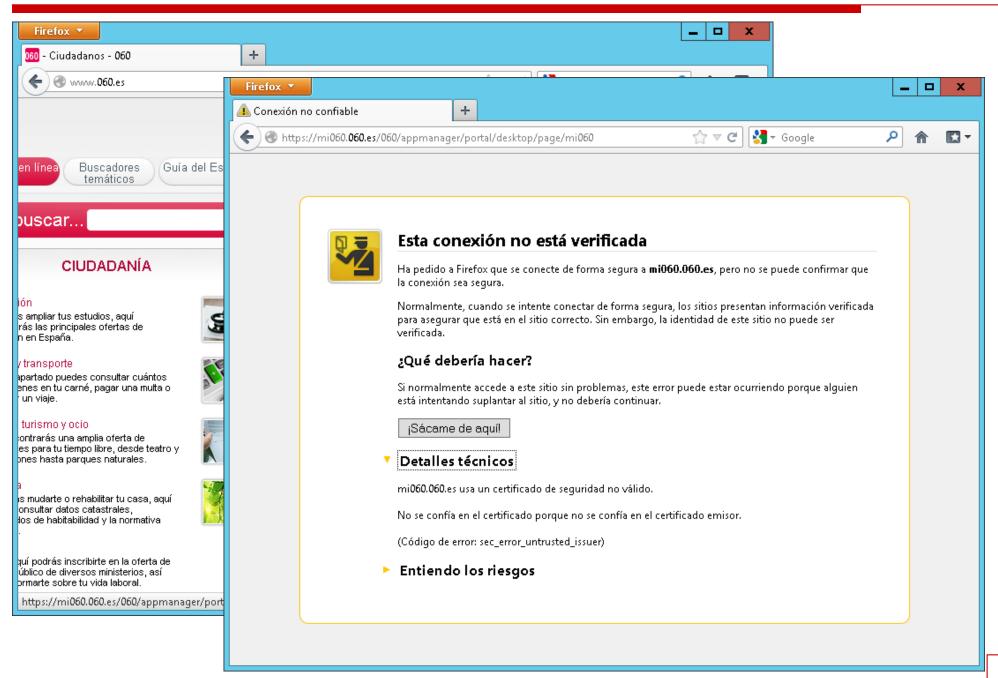
Ejemplo 4: mareando al usuario (3/3)



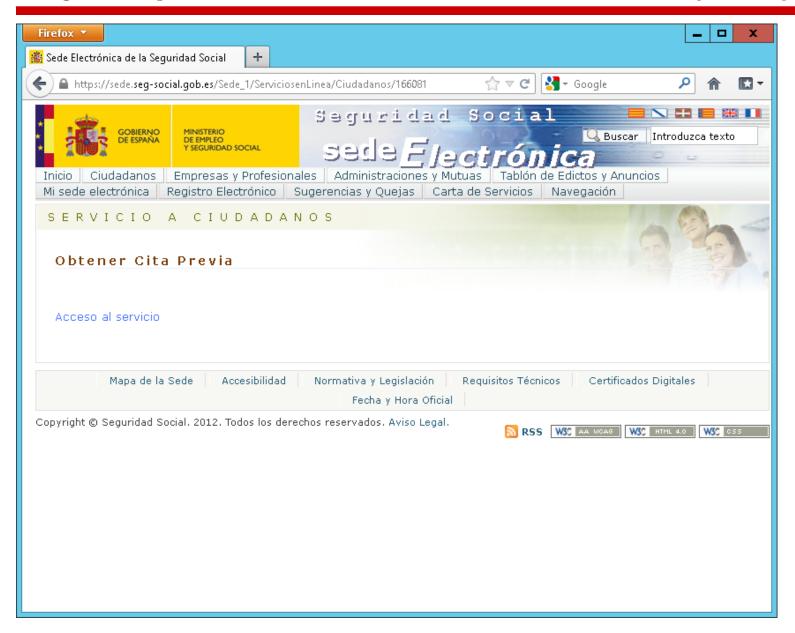
Ejemplo 5: certificado raíz (1/2)



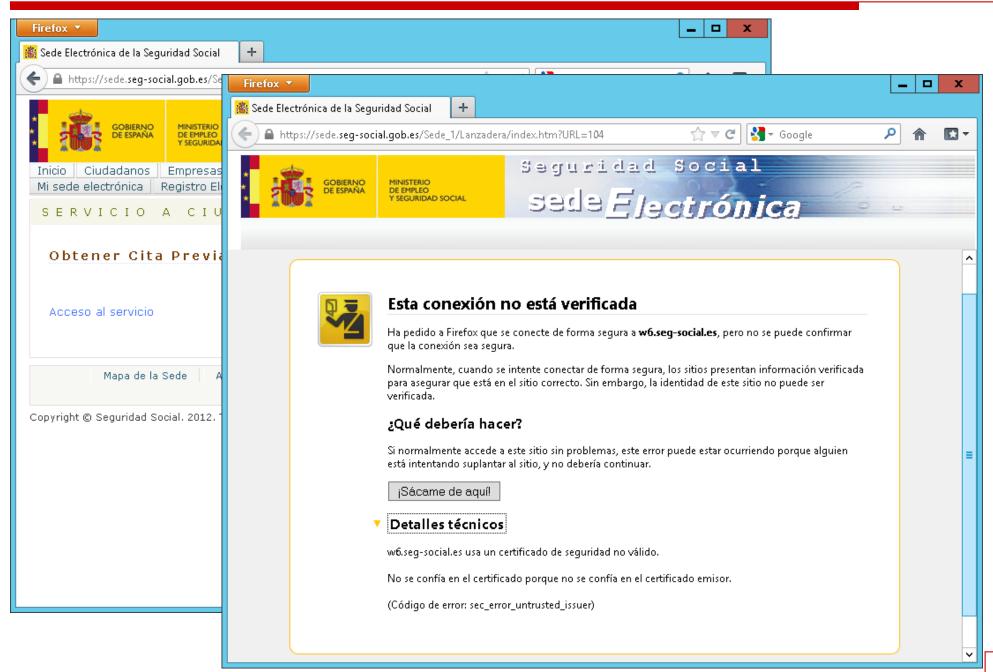
Ejemplo 5: certificado raíz (2/2)



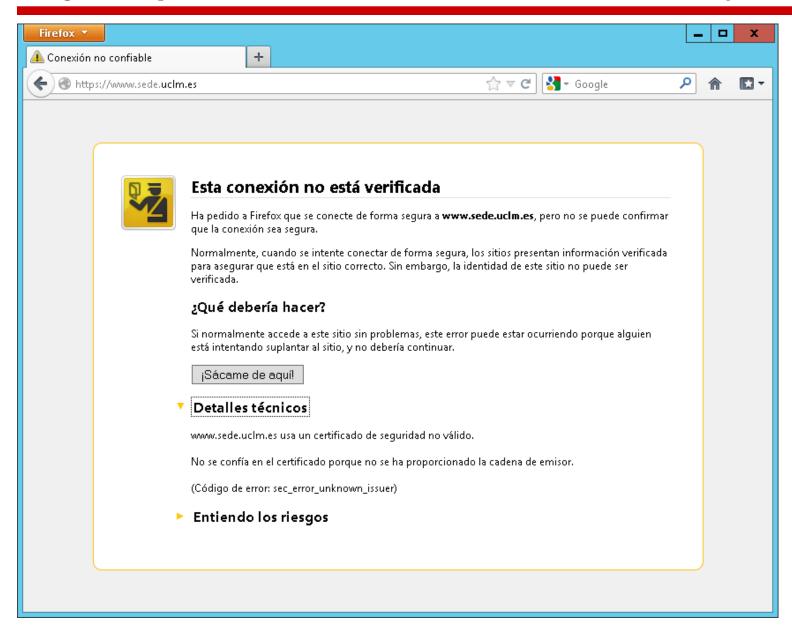
Ejemplo 6: certificado raíz (1/2)



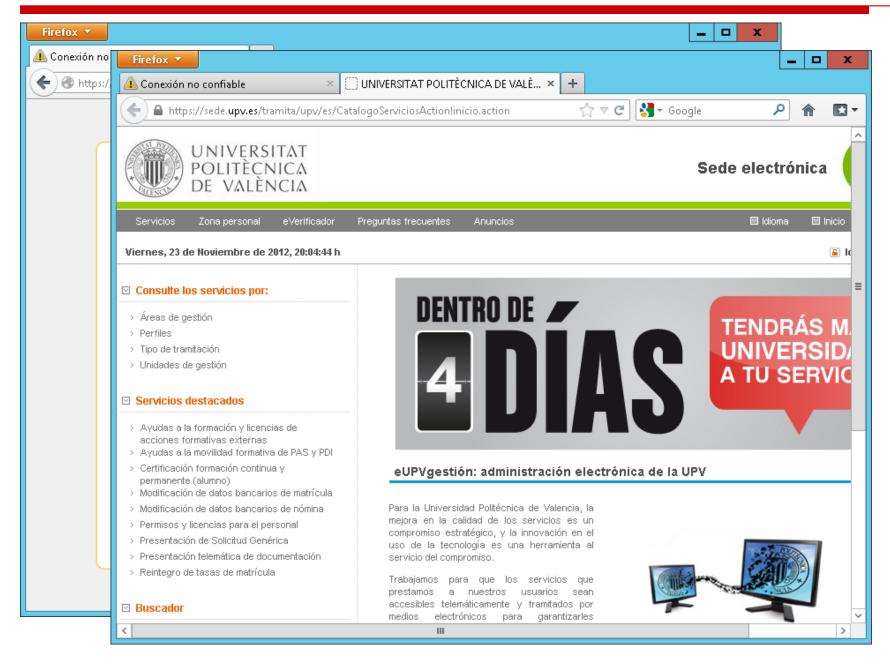
Ejemplo 6: certificado raíz (2/2)



Ejemplo 7: cadena errónea (1/3)



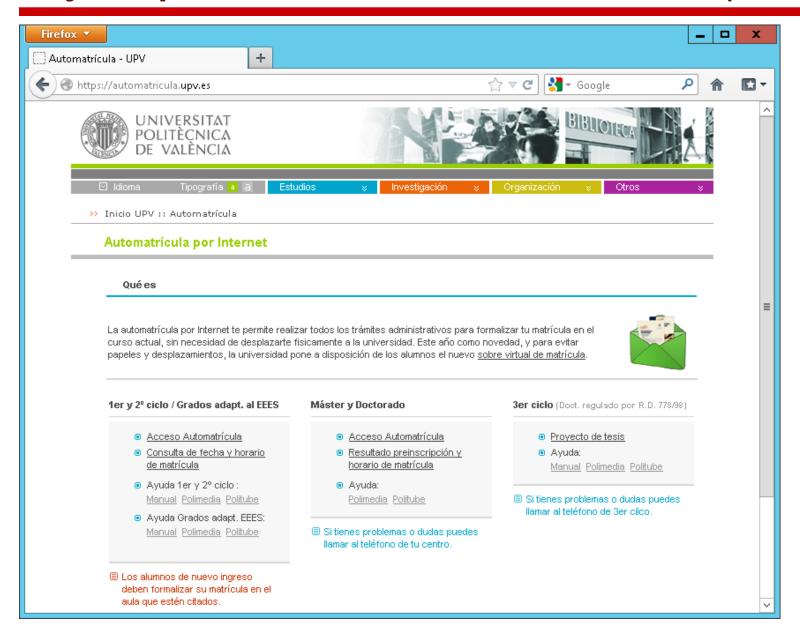
Ejemplo 7: cadena errónea (2/3)



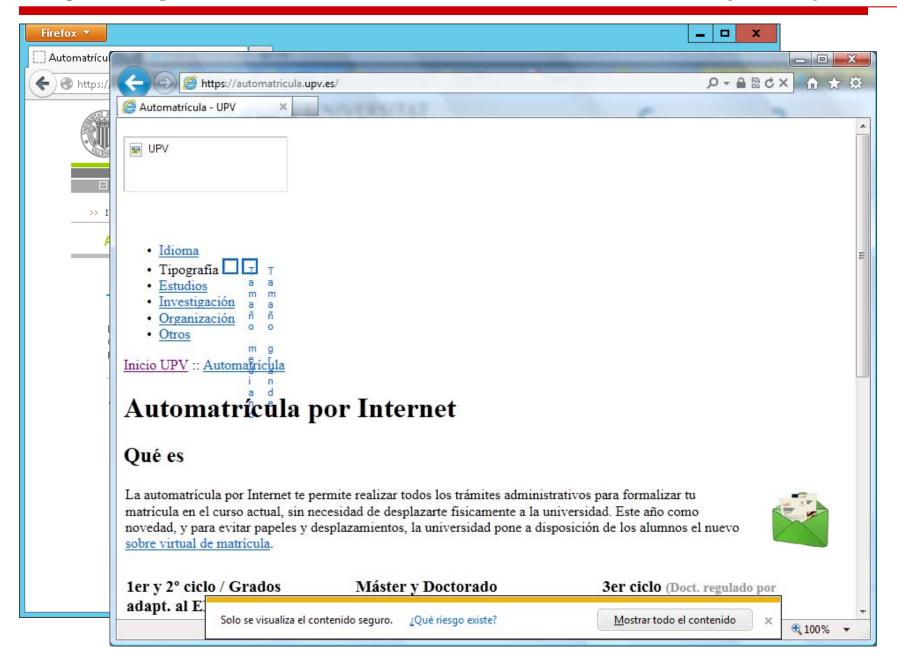
Ejemplo 7: cadena errónea (3/3)



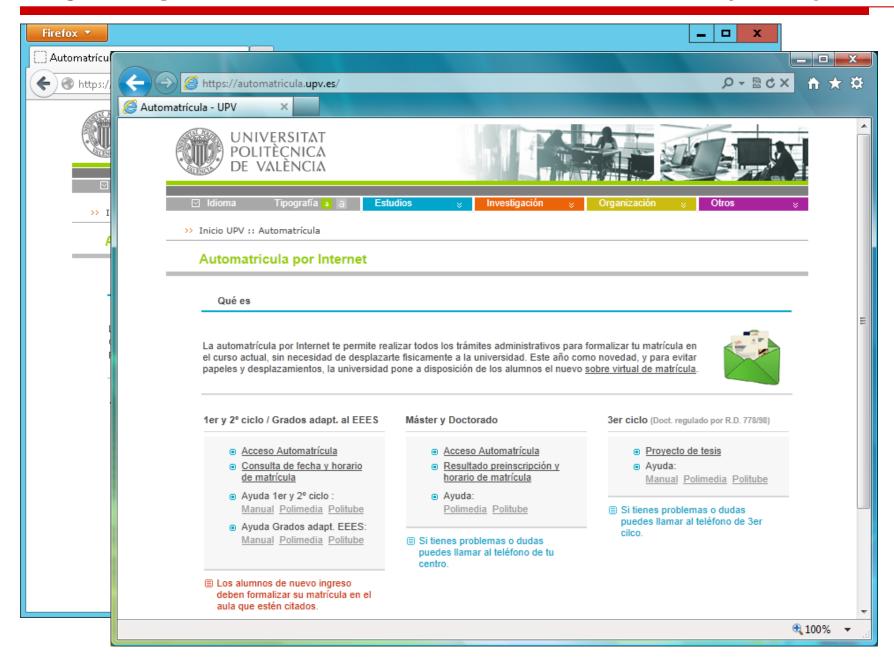
Ejemplo 8: contenido mixto (1/5)



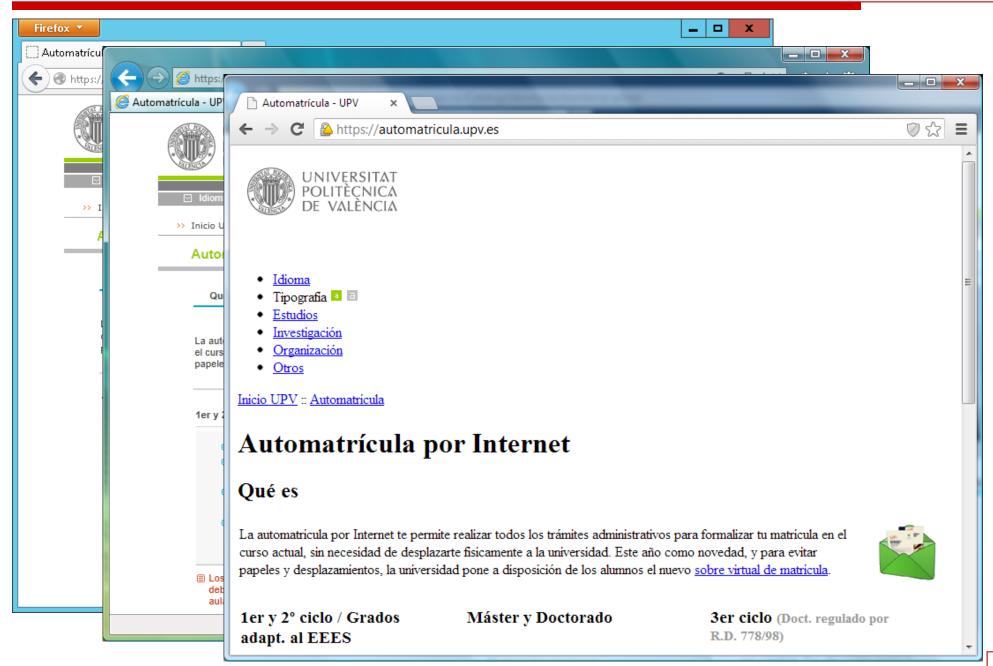
Ejemplo 8: contenido mixto (2/5)



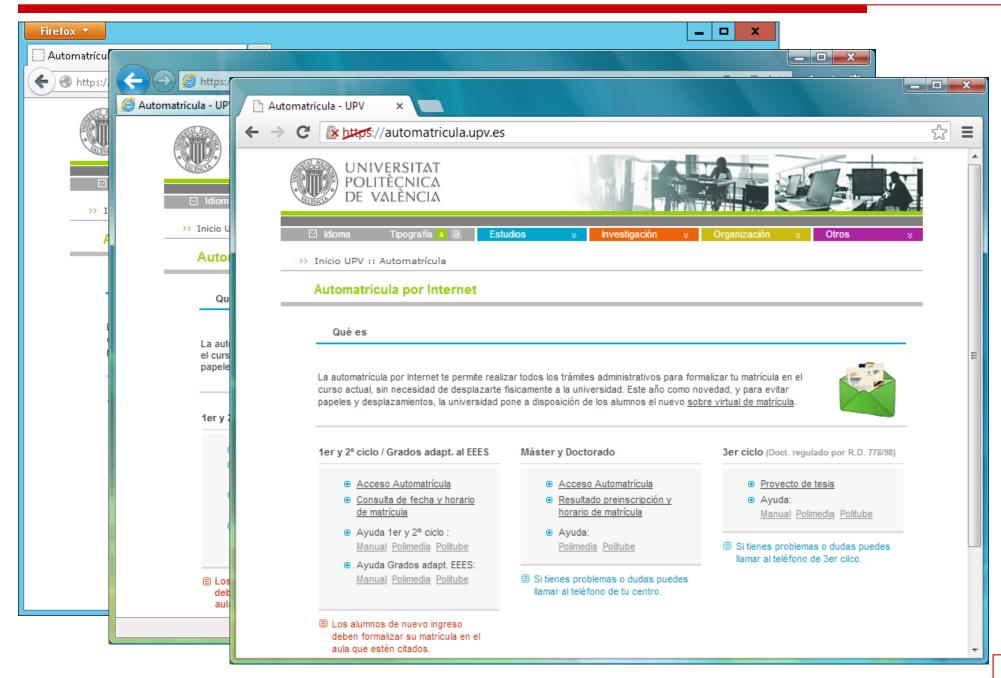
Ejemplo 8: contenido mixto (3/5)



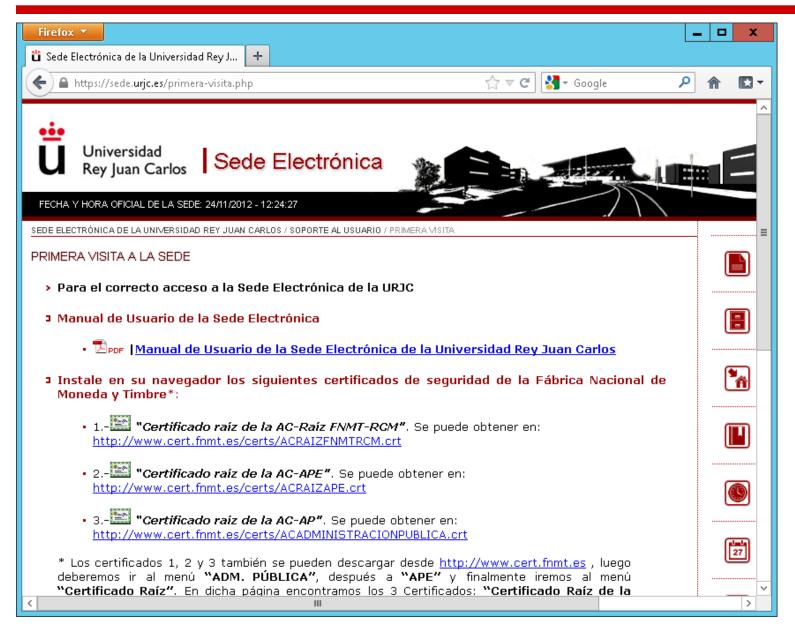
Ejemplo 8: contenido mixto (4/5)



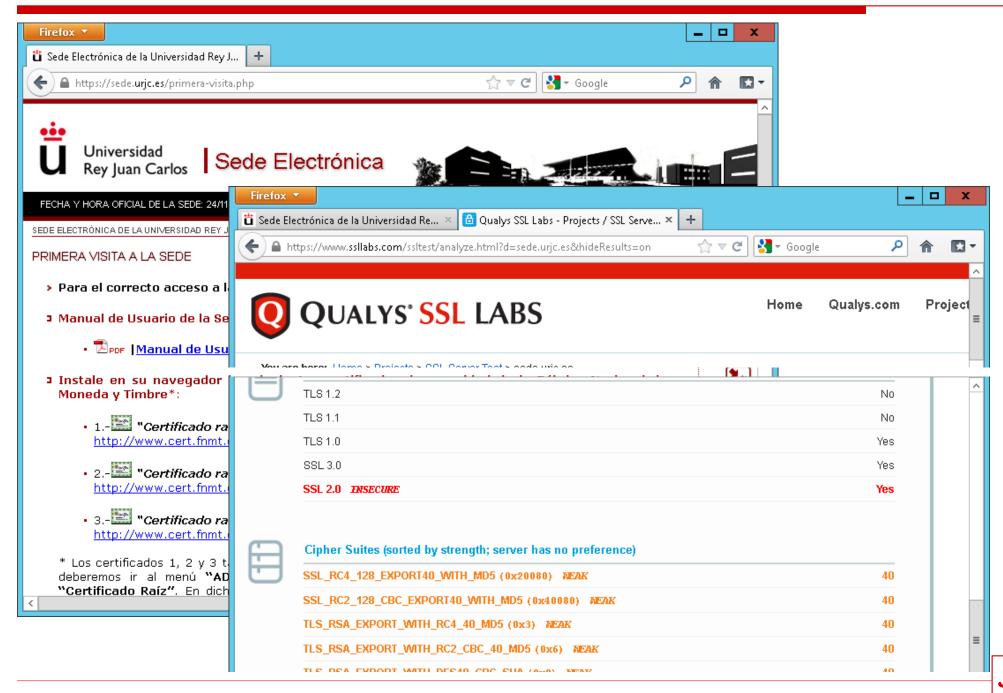
Ejemplo 8: contenido mixto (5/5)



Ejemplo 9: otros problemas (1/2)



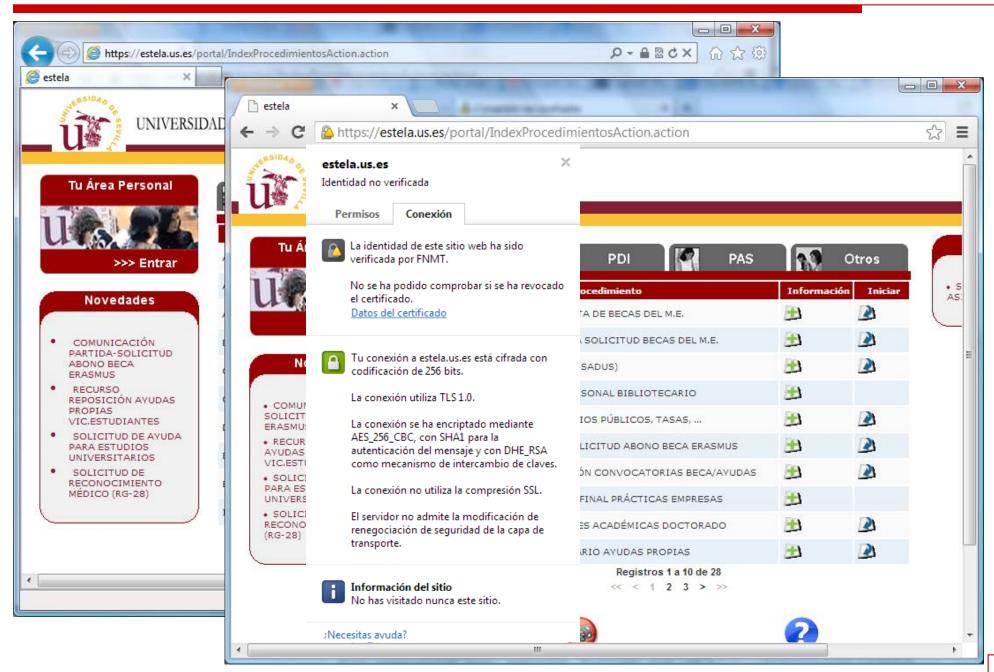
Ejemplo 9: otros problemas (2/2)



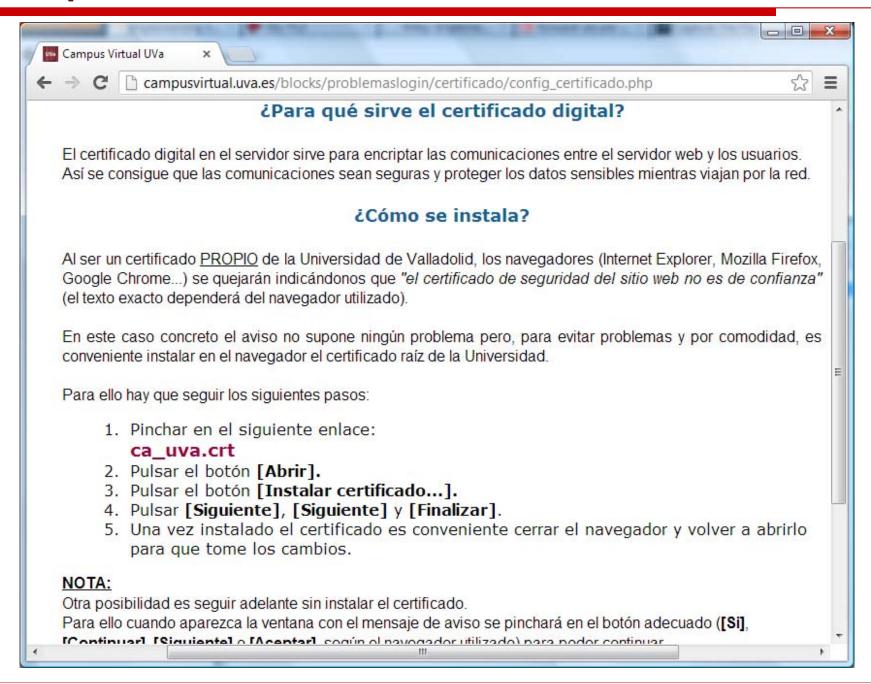
Ejemplo 10: otros problemas (1/2)



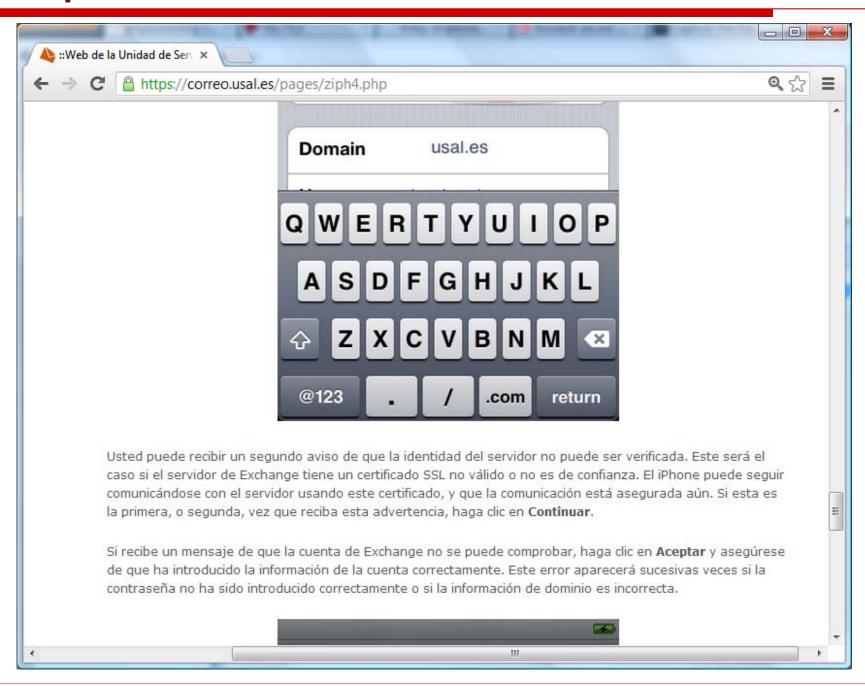
Ejemplo 10: otros problemas (2/2)



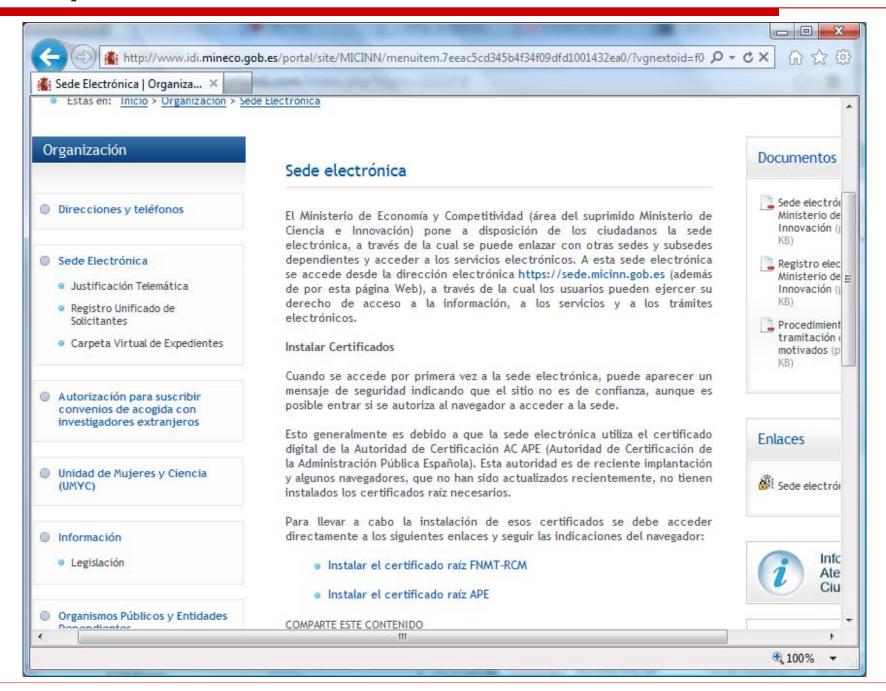
Ejemplo 11: instrucciones de uso



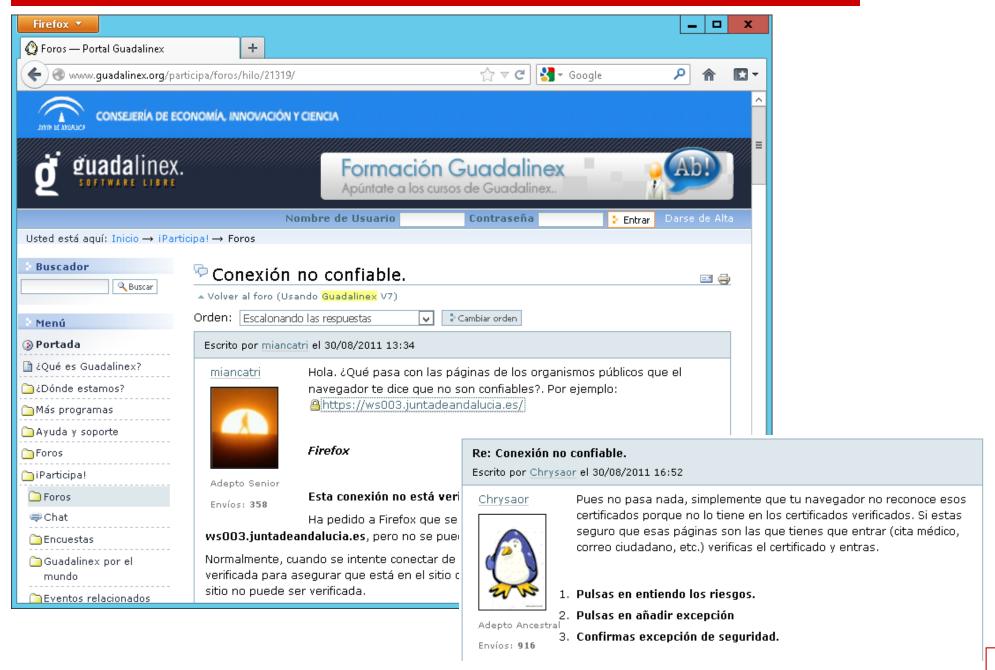
Ejemplo 12: instrucciones de uso



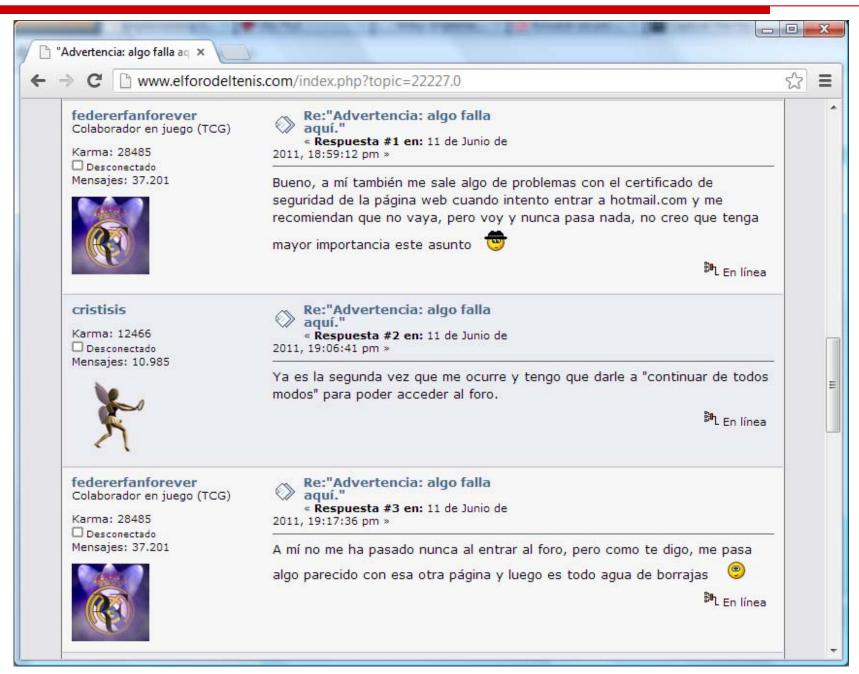
Ejemplo 13: instrucciones de uso



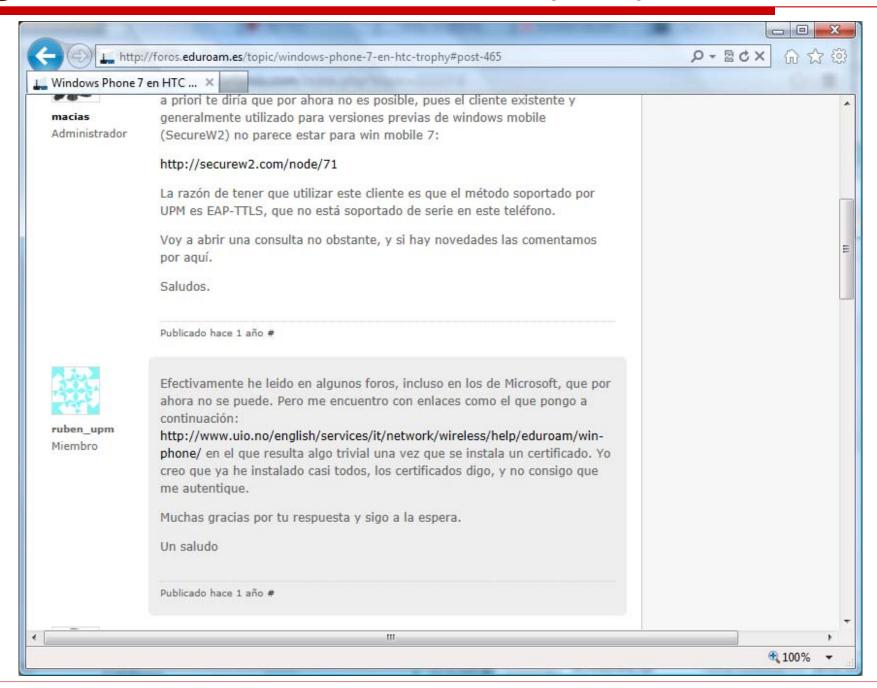
Algunas consecuencias (1/3)



Algunas consecuencias (2/3)



Algunas consecuencias (3/3)



La situación ideal

- El uso de SSL/TLS debería garantizar, en nuestras conexiones:
 - la confidencialidad
 - la integridad
 - la autenticidad
 - la disponibilidad
- SSL tiene su origen en la navegación web (nace como una extensión de HTTP), pero hoy en día soporta cualquier tipo de aplicación
 - □ SMPT, POP, IMAP, FTP, NNTP, etc.
- Tampoco hay que olvidar las conexiones entre servidores (sin interacción de usuario)

SSL / TLS

- ¿Qué protocolo(s) deberíamos implementar?
 - **★** SSLv2 (1995): Secure Sockets Layer
 - http://www.mozilla.org/projects/security/pki/nss/ssl/draft02.html
 - Propuesta original de Netscape. Hoy en día es inseguro
 - ✓ SSLv3 (1996)
 - http://www.mozilla.org/projects/security/pki/nss/ssl/draft302.txt
 - Actualmente en uso. Aceptado como base por el IETF
 - ✓ TLS 1.0 (1999): Transport Layer Security
 - http://www.ietf.org/rfc/rfc2246.txt
 - Prácticamente similar a SSLv3. El más utilizado hoy en día
 - ✓TLS 1.1 (2006): RFC 4346
 - Mejora la versión anterior y tiene un gran base de clientes
 - ✓TLS 1.2 (2008): RFC 5246
 - Versión actual (teóricamente). Soportado por IE en W7+

Confidencialidad

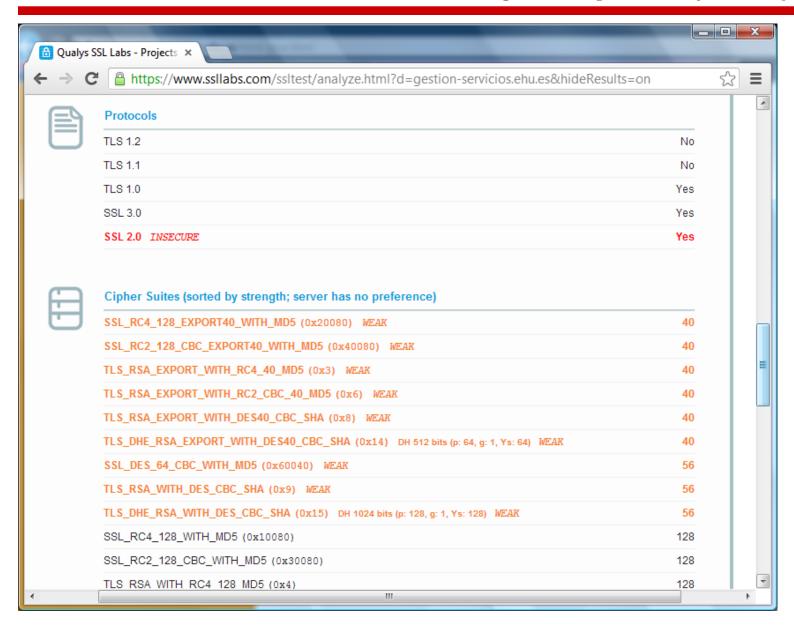
- Garantizar la confidencialidad es sencillo: sólo hay que cifrar los datos
- El cifrado simétrico es el apropiado para cifrar / descifrar grandes cantidades de información
 - ambos extremos deben conocer la clave de cifrado
- Así pues, basta con elegir una algoritmo de cifrado y ponerse de acuerdo en la clave a utilizar
 - □ ya veremos el problema de comunicar la clave...
- Los algoritmos pueden funcionar en modo bloque o en modo flujo de datos, teniendo disponibles:
 - □ bloque: DES, 3DES, AES, Camellia, SEED, ARIA
 - □ flujo: RC4

Confidencialidad: a tener en cuenta

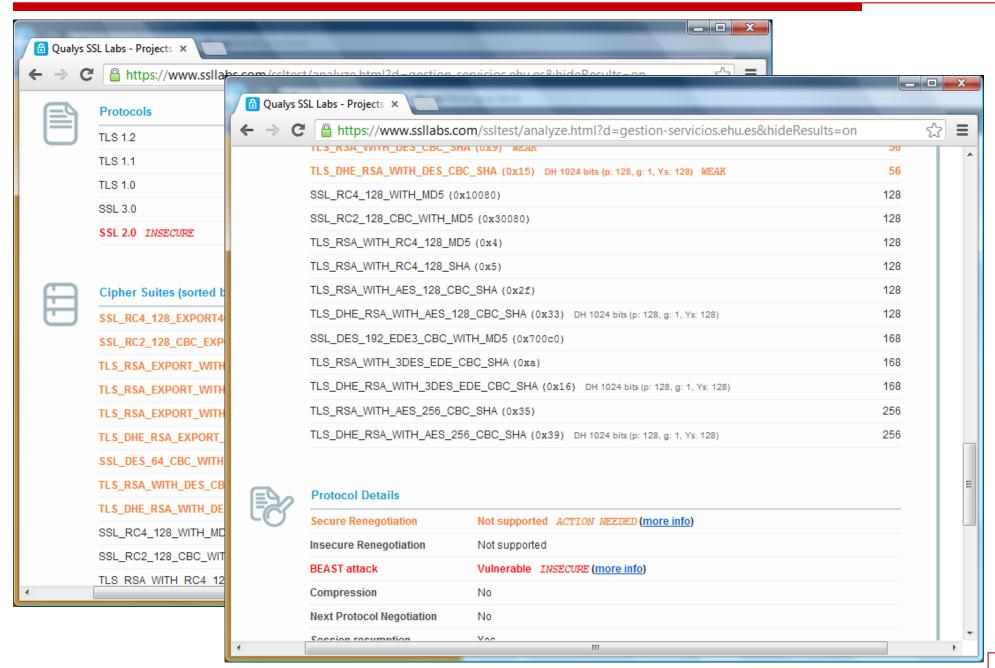
- La seguridad del cifrado simétrico se basa en la clave: en su tamaño y en que se mantenga secreta
- Toda clave cuyo tamaño sea inferior a 128 bits se considera débil
 - en otro caso, los ataques por fuerza bruta son efectivos
 - esto descarta DES y "3DES mal implementado"
- Los algoritmos por bloques pueden trabajar en distintos modos: ECB, CBC, OFB, ...
 - ECB no es apropiado -> no se utiliza
 - □ CBC es vulnerable en SSLv3 / TLS 1.0 (no en TLS 1.1+)■ BEAST (CVE-2011-3389)
 - "Browser Exploit Against SSL/TLS"

 Juliano Rizzo, Thai Duong; ekoparty 2011

Confidencialidad: ejemplo (1/2)



Confidencialidad: ejemplo (2/2)

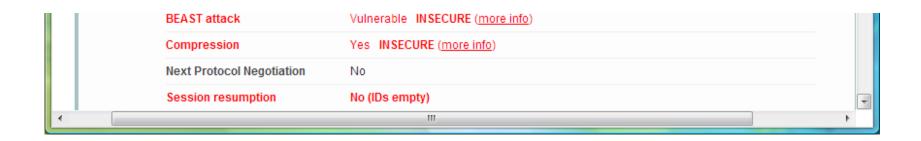


Confidencialidad: compresión

- En la ekoparty 2012, Juliano Rizzo y Thai Duong, nos sorprenden con la puesta en marcha de otro ataque (conocido pero etiquetado como teórico):
 - □ CRIME (Compression Ratio Info-leak Made Easy)
- El problema se presenta si utilizamos compresión (si se utiliza es antes del cifrado, claro)
 - en ese caso corremos el riesgo de fuga de información
- Google estaba trabajando seriamente en la mejora del rendimiento de HTTP y la compresión era un mecanismo fundamental
 - a raíz de este ataque se han visto obligados a eliminar la compresión en SSL

Compresión: ejemplo





Integridad

- Puede parecer que habiendo cifrado no es necesario implementar ningún mecanismo de integridad
- Pero existen ataques (de truncamiento, por ejemplo) que exigen garantizar la integridad de los mensajes
- Como es habitual, la integridad se verifica mediante un resumen (hash) del mensaje
- Para ello se utiliza la función MD5 o alguna variante de SHA (SHA-1, SHA-256, SHA-384, SHA-512)
- Estas funciones se utilizan con el mecanismo HMAC,
 que utiliza claves secretas al calcular el resumen
- Para cada registro SSL se genera su HMAC antes de ser cifrado

47

Disponibilidad

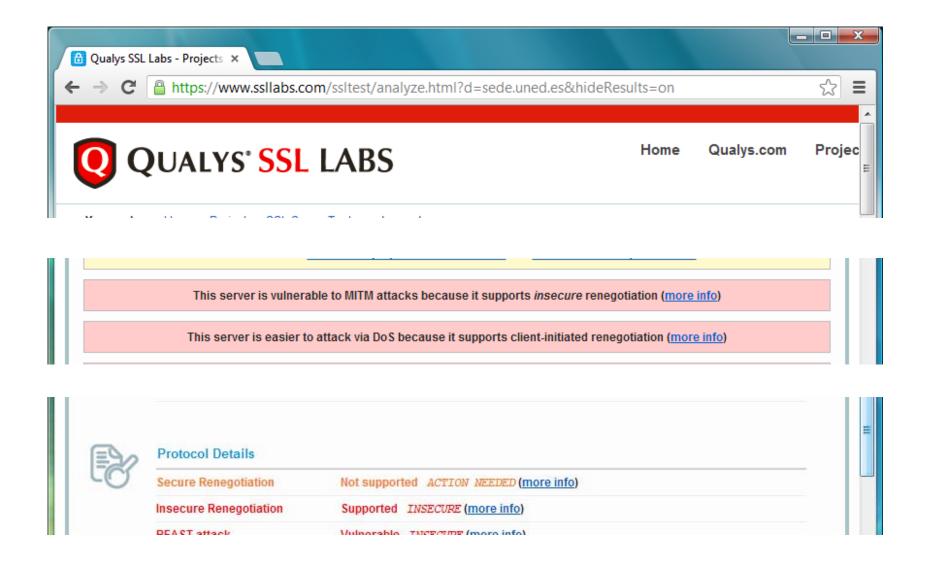
- Está muy extendida la idea de que el uso de SSL consume muchos recursos
- Pero Google (como empresa, no como buscador) no está de acuerdo:
 - http://www.imperialviolet.org/2010/06/25/ overclocking-ssl.html
 - http://www.imperialviolet.org/2011/02/06/ stillinexpensive.html
- Y ya que tenemos recursos suficientes para utilizar SSL en todas las conexiones:
 - Always On SSL
 Online Trust Alliance
 https://otalliance.org/resources/AOSSL/

SSL

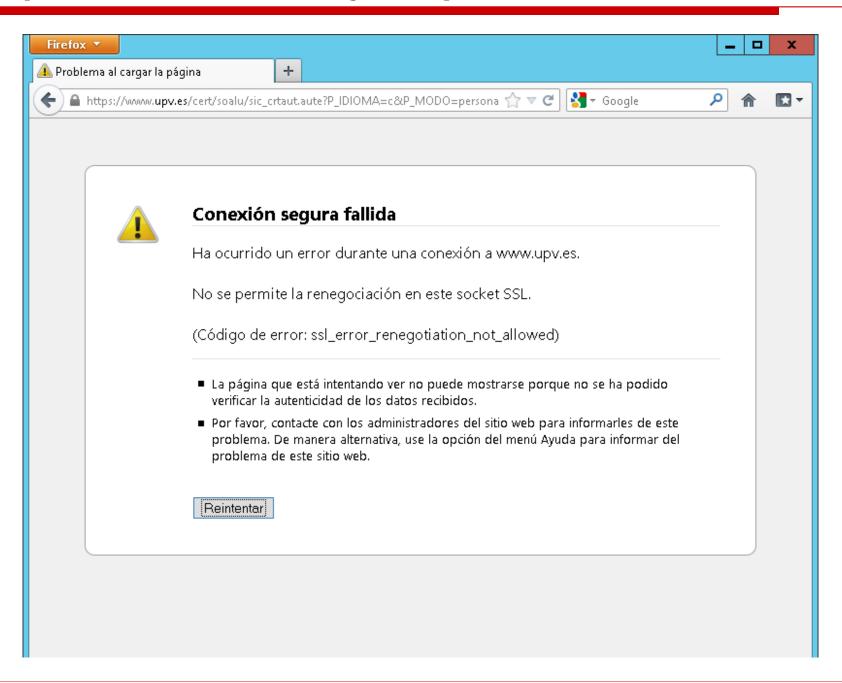
Disponibilidad: denegación de servicio

- Si nuestro servidor permite la renegociación SSL iniciada por el cliente, somos vulnerables a ataques de DoS (Denial Of Service)
- Actualmente se considera que la renegociación sólo debería iniciarla el servidor
- Además, en 2009 se descubrió un fallo de diseño que permitía aprovechar la renegociación para inyectar tráfico en una sesión SSL
 - https://community.qualys.com/blogs/securitylabs/2009/11 /05/ssl-and-tls-authentication-gap-vulnerability-discovered
- Así pues, nuestros servidores deberían implementar el RFC 5746, que soluciona el fallo descrito

Disponibilidad: ejemplo 1



Disponibilidad: ejemplo 2



Autenticidad

- Para comprobar la autenticidad del servidor se utilizan certificados digitales: asociación entre una clave pública y los datos de identificación avalada por un tercero de confianza
 - también se pueden utilizar certificados digitales para autenticar al cliente, aunque no es muy habitual todavía
- En general se utiliza RSA para gestionar y utilizar el par de claves (una pública y una privada)
 - basado en la complejidad de factorizar en números primos
- Pero veremos aparecer las curvas elípticas en breve
- Los certificados digitales son el origen de la mayoría de problemas que salen a la superficie en SSL

Certificados digitales

- A la hora de obtener un certificado digital tenemos que tomar unas cuantas decisiones:
 - el tamaño de las claves
 - la identificación asociada a la clave pública
 - □ la entidad raíz a la que solicitamos el certificado
 - el período de validez
 - el tipo de certificado
- Un "mal" certificado puede ponernos las cosas muy difíciles para generar confianza en los usuarios
 - □ por decirlo de manera suave...
- Un "buen" certificado no es suficiente para que SSL funcione bien (automáticamente)

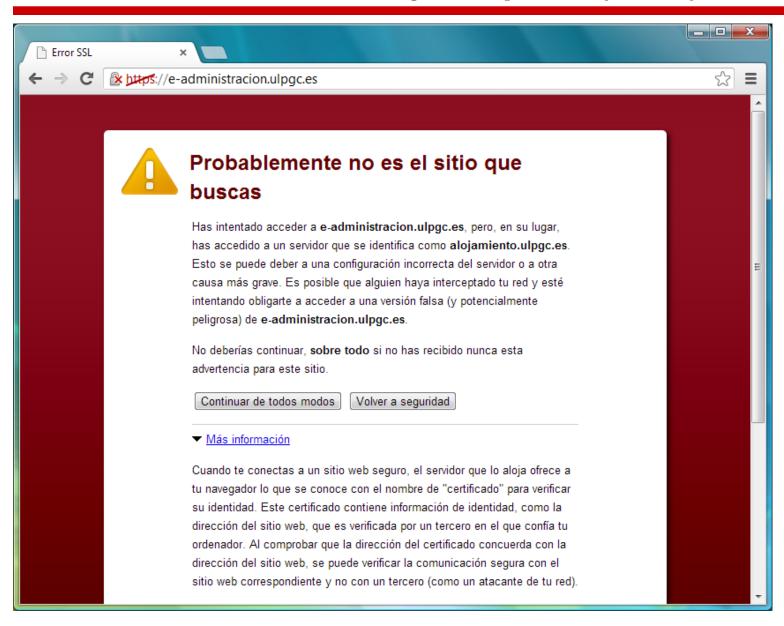
Certificados digitales: claves

- Las claves de cifrado asimétrico tienen un tamaño mucho mayor que las claves de cifrado simétrico
- El mínimo exigible para RSA es de 1024 bits
- Aunque el mínimo recomendable es de 2048 bits
 - romper una clave asimétrica de 2048 bits es equivalente a romper una clave simétrica de 112 bits
- Cuanto más tiempo utilicemos las mismas claves, más tiempo daremos para averiguarlas
- Se recomienda utilizar claves asimétricas sólo durante un año y generar nuevas con cada renovación del certificado
 - □ se desaconseja completamente utilizarlas más de 3 años

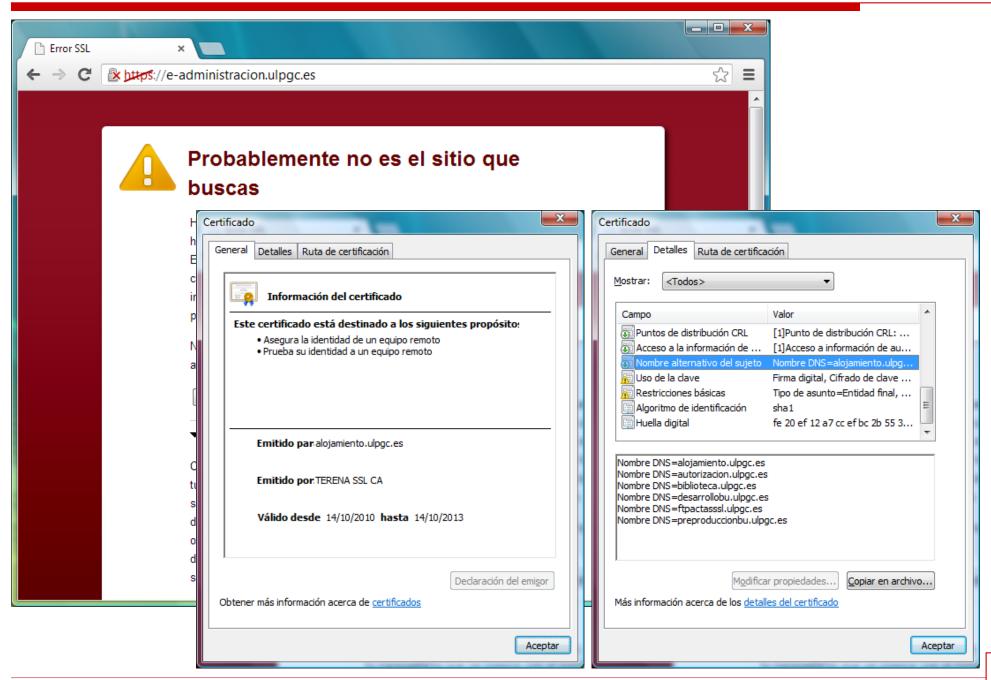
Certificados digitales: identificación

- Cuando un usuario se conecta a un servidor utiliza un nombre DNS para localizarlo
- En nuestro certificado tendremos que asociar la clave pública a todos los nombres que pueden utilizar los usuarios para llegar a nuestro servidor
- Podría ser que sólo tuviésemos un nombre posible
 - aparecería como CN dentro del atributo Subject
- Que tuviésemos varios nombres alternativos
 - aparecerían en el atributo SubjectAlternativeName (y, en dicho caso, el atributo Subject se ignora)
- Que utilicemos un comodín (como prefijo) para agrupar todos los nombres de un subdominio

Identificación: ejemplo (1/2)



Identificación: ejemplo (2/2)



Certificados digitales: entidad raíz

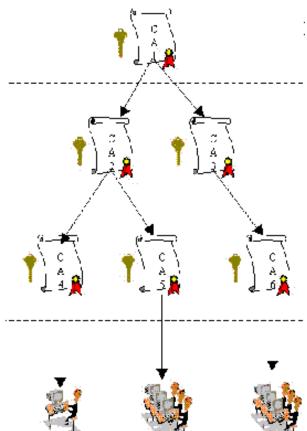
- La asociación entre clave pública e identificación la firma una entidad certificadora (CA)
- Para que los clientes confíen en el certificado tienen que confiar, previamente, en dicha entidad
- En un entorno controlado podríamos utilizar nuestra propia CA y distribuirla a todos nuestros clientes
 ¿existe dicho entorno?
- La opción más adecuada es utilizar una de las entidades raíz que estén distribuidas ampliamente
 en la mayoría de navegadores y dispositivos
- RedIRIS nos proporciona gratuitamente certificados firmados por la CA de Comodo (actualmente)

58

Certificados digitales: jerarquía

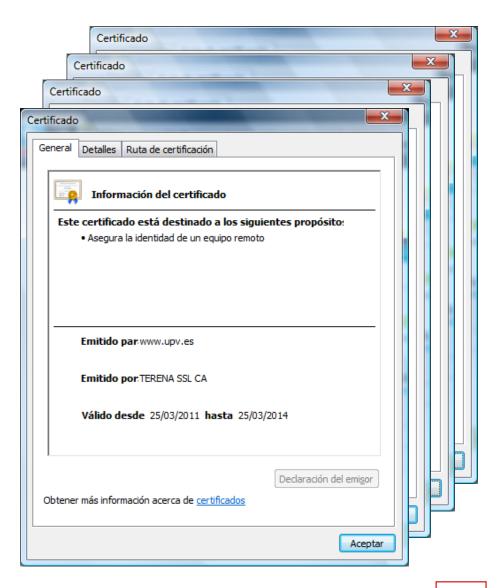
 Las autoridades de certificación utilizan una jerarquía de entidades, de manera que la entidad raíz sólo firma a otras entidades certificadoras

- Los clientes sólo necesitan confiar en las entidades raíz, nunca es necesario que instalen entidades intermedias
- No entregar los certificados de las entidades intermedias es uno de los errores más frecuentes
 - además, no se puede verificar "probando" que funciona



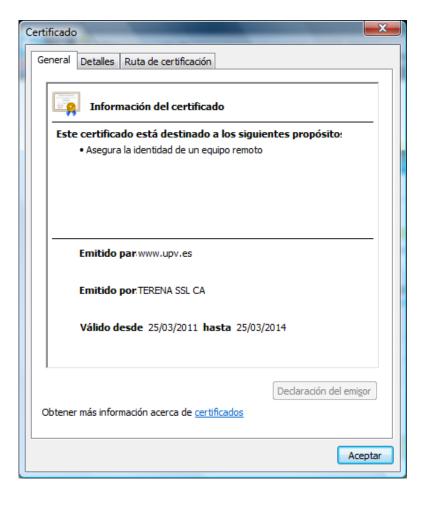
Jerarquía de certificados (1/6)

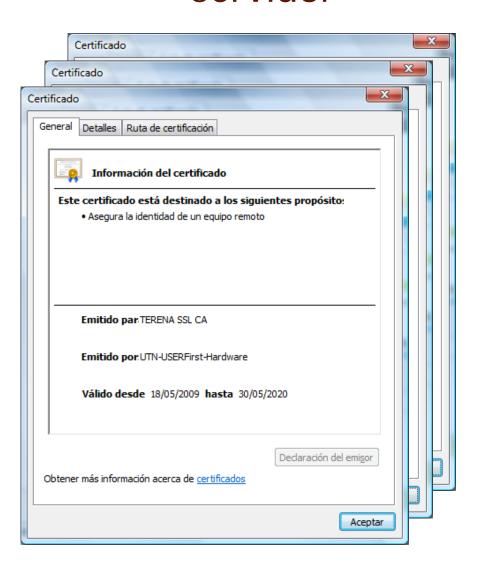
cliente



Jerarquía de certificados (2/6)

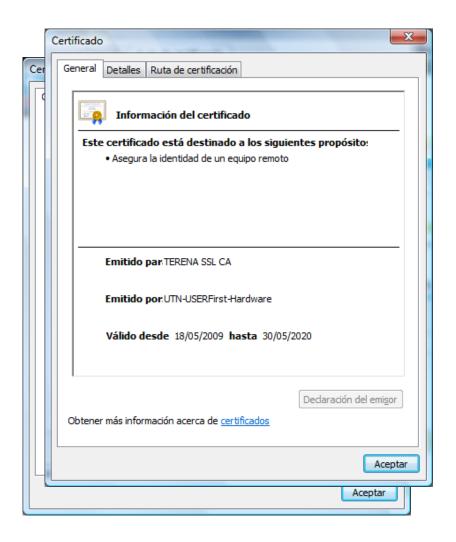
cliente

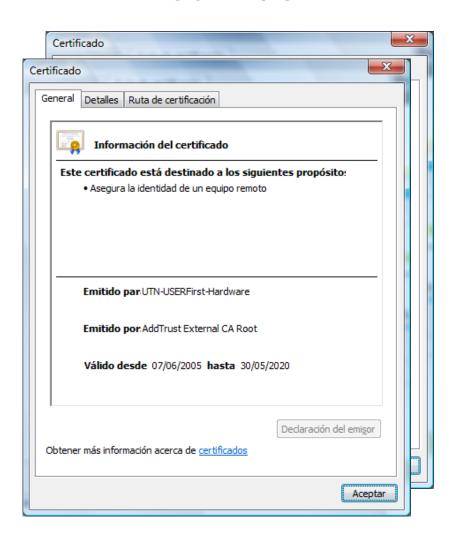




Jerarquía de certificados (3/6)

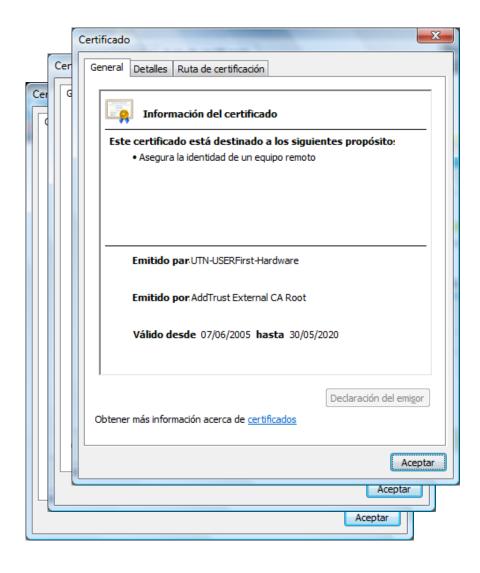
cliente

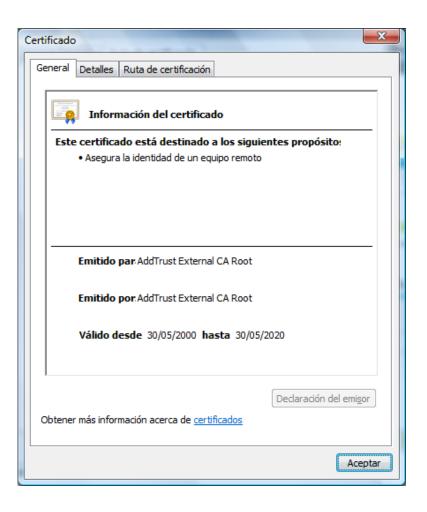




Jerarquía de certificados (4/6)

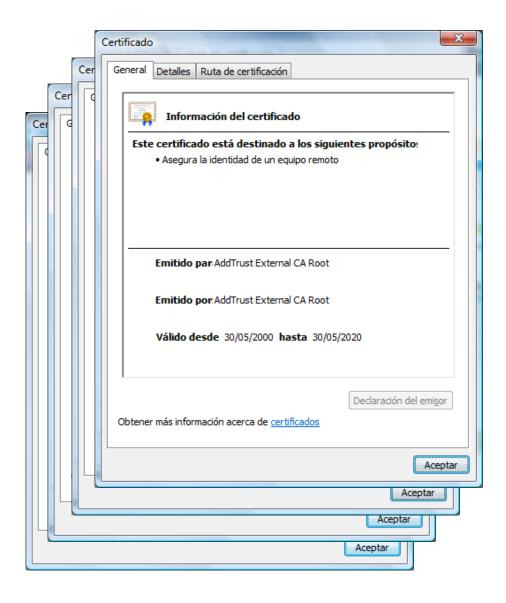
cliente





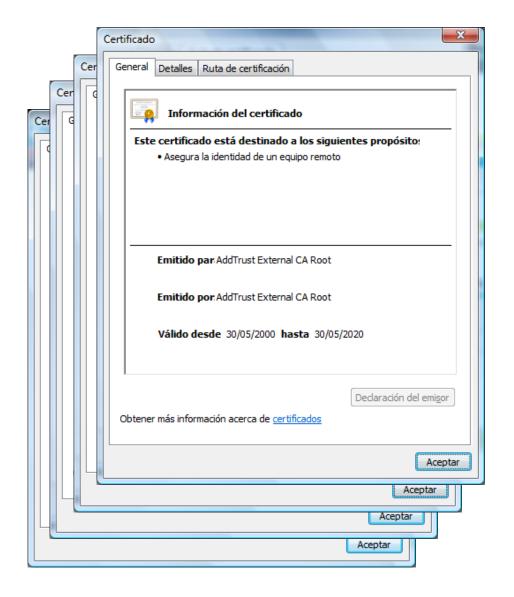
Jerarquía de certificados (5/6)

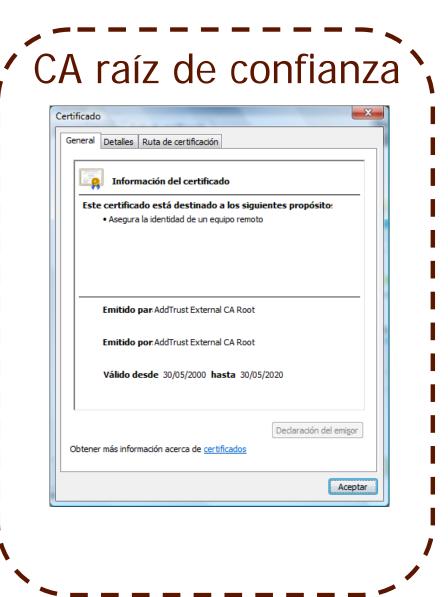
cliente



Jerarquía de certificados (6/6)

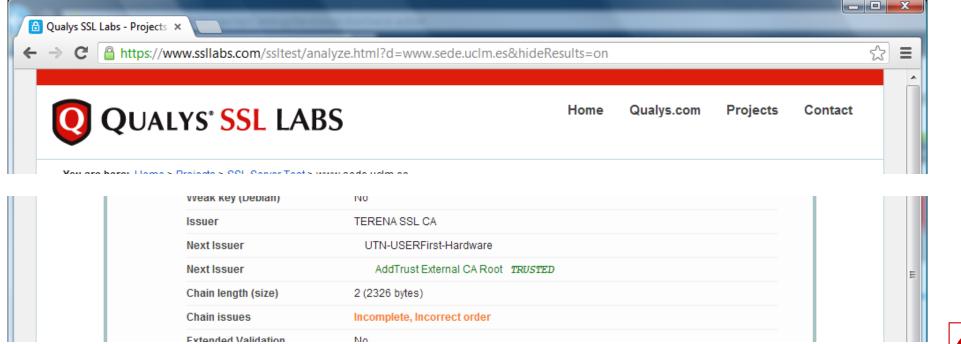
cliente





Cadena errónea: ejemplo

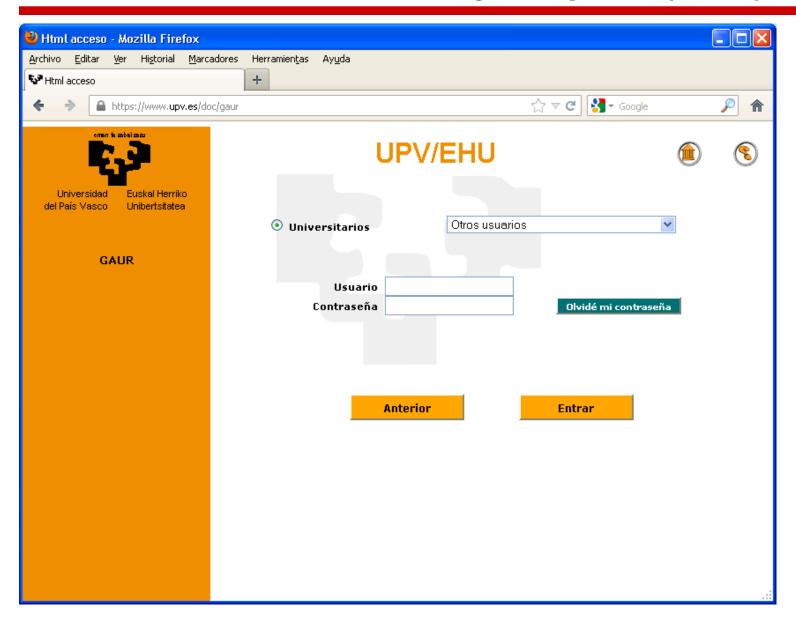
```
>openssl s_client -connect www.sede.uclm.es:443
...
Certificate chain
0 s:/C=ES/ST=Castilla-La Mancha/L=Castilla-La Mancha/O=Universidad de Castilla-La Mancha - Q1368009E/OU=uclm.es/CN=www.sede.uclm.es
i:/C=NL/O=TERENA/CN=TERENA SSL CA
1 s:/C=US/ST=UT/L=Salt Lake City/O=The USERTRUST Network/OU=http://www.usertrus
t.com/CN=UTN-USERFirst-Hardware
i:/C=SE/O=AddTrust AB/OU=AddTrust External TTP Network/CN=AddTrust External C
A Root
...
```



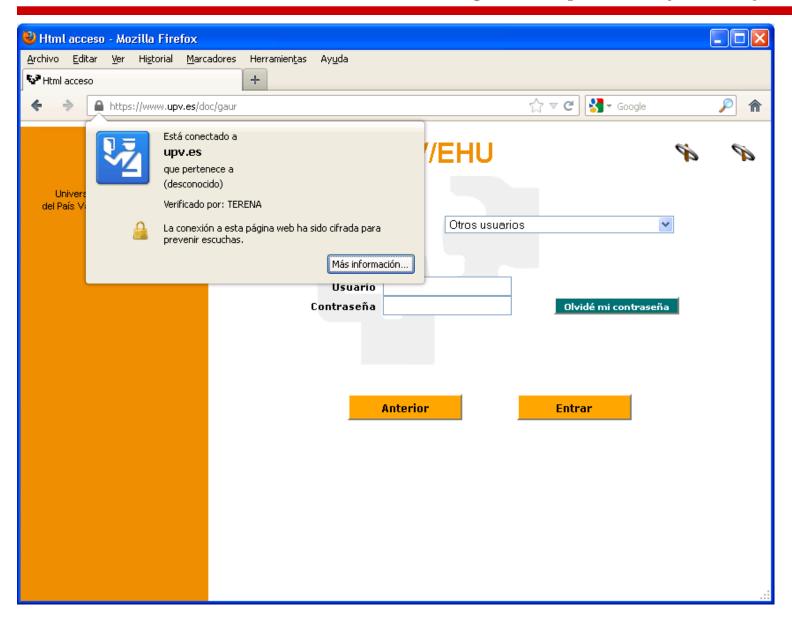
Certificados digitales: tipo

- Los certificados digitales establecen el uso que se puede dar a las claves y los datos identificativos que se muestran sobre el propietario
- También hay distintas opciones en cuanto a las comprobaciones que hace una CA sobre los datos
- Esto da lugar a distintos tipos de certificados:
 - □ clase 1 / domain validation
 - clase 2 / organization verified
 - extended validation (EV)
 - de sede electrónica
- Para el usuario sólo hay 2 tipos: el "verde" (EV) y todos los demás

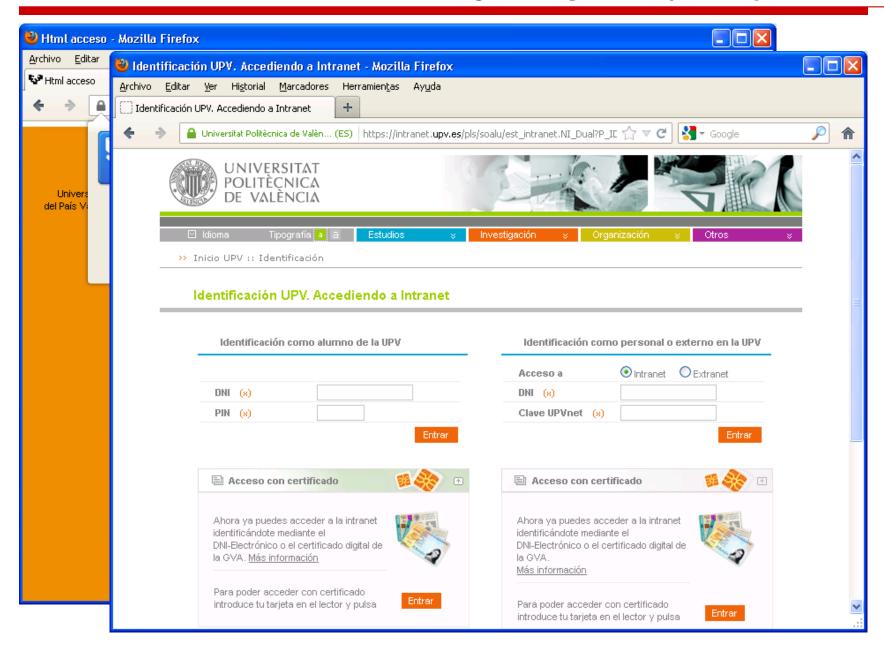
Certificados EV: ejemplo (1/4)



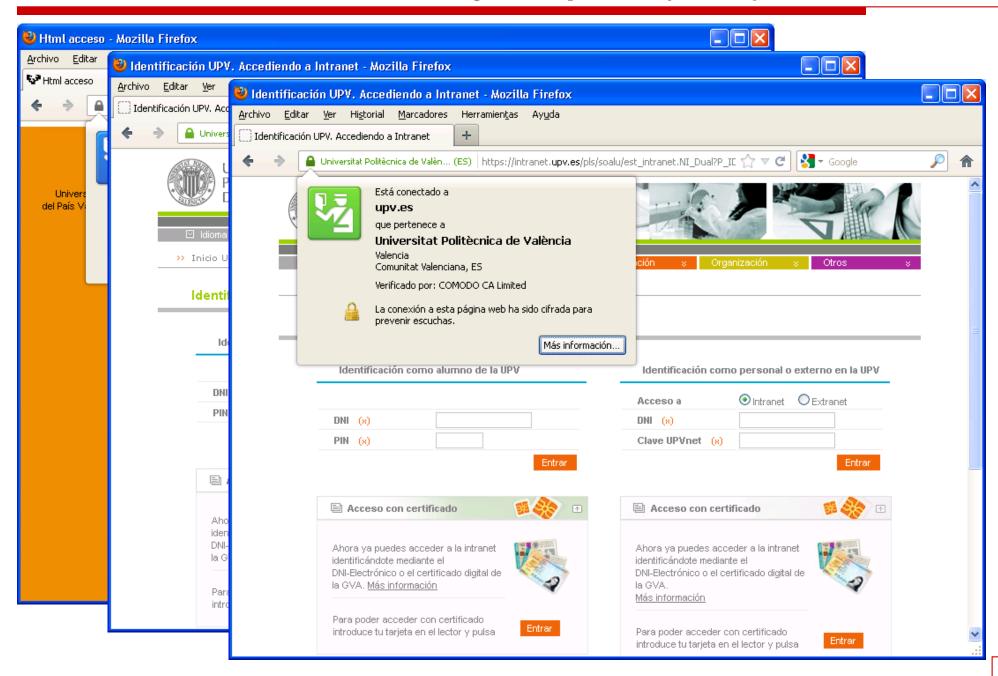
Certificados EV: ejemplo (2/4)



Certificados EV: ejemplo (3/4)



Certificados EV: ejemplo (4/4)



Certificados: validación

- Para validar un certificado hay que comprobar que:
 - el nombre que utilizamos coincide con uno de los nombres incluidos en el certificado
 - el certificado se encuentra vigente
 - el certificado no ha sido revocado
- Éstas comprobaciones se tienen que hacer para el certificado del servidor y para los certificados de todas las entidades intermedias
- La revocación se puede consultar a través de listas
 CRL o utilizando un servicio OCSP
- Hablando de la FNMT: ¿qué pasaría si te roban la clave privada de tu certificado?

Certificados: OCSP

- Comprobar la revocación es fundamental para confiar en un certificado
- OCSP mejora el rendimiento de las CRL, pero ha tenido y sigue teniendo algunos problemas
 - por ejemplo, de privacidad: cada vez que visitamos un sitio web se lo "decimos" a la CA de su certificado
- La extensión de TLS Status Request (conocida como OCSP stapling: grapado OCSP) es perfecta:
 - el cliente solicita al servidor (en el establecimiento del túnel SSL) que le informe del estado de revocación de su certificado
 - es el servidor, por tanto, el encargado de hacer las consultas OCSP

Intercambiando claves simétricas

- Habíamos dejado pendiente un gran problema: ¿cómo nos ponemos de acuerdo para usar una clave secreta (y que siga siendo secreta)?
- El uso de certificados digitales para la autenticación resuelve este problema:
 - la clave simétrica se cifra de manera asimétrica
- Grosso modo: el cliente cifra, con la clave pública del servidor, la clave simétrica y se la envía a éste
- Y si alguien guarda la comunicación cifrada y logra descubrir (o robar) la clave privada del servidor?
- Secreto permanente (perfect forward secrecy): Diffie-Hellman (EDH) para intercambio de claves

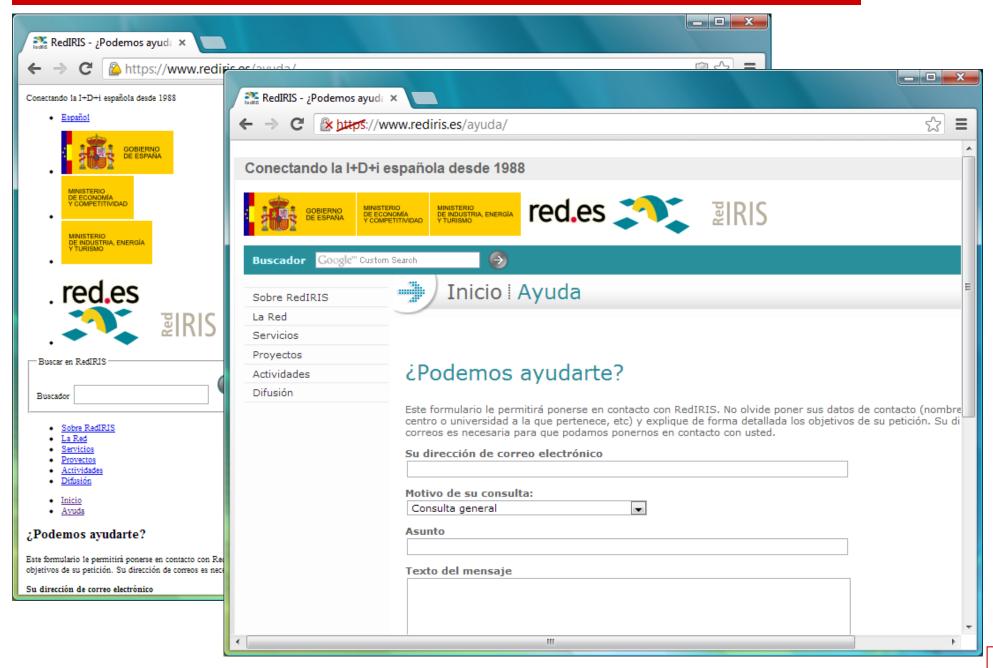
Conjuntos de cifrado

- Para establecer un túnel SSL, cliente y servidor utilizan descriptores (*cipher suites*) para ponerse de acuerdo en los siguientes elementos:
 - algoritmo de intercambio de clave
 - algoritmo de cifrado
 - algoritmo de resumen (hash)
- Ejemplos:
 - □ TLS_RSA_WITH_AES_128_CBC_SHA256
 - □ TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA
- Puedes consultar el listado completo en:
 - http://www.iana.org/assignments/tls-parameters/ tls-parameters.xml#tls-parameters-3

Sitios web: un caso especial

- Las características de la web hacen que SSL no sea suficiente para garantizar las comunicaciones:
 - si consiguen robar nuestras cookies podrán suplantarnos (aunque tengamos perfectamente configurado SSL)
- Además de desarrollar la web con la seguridad en mente, hay que tener cuidado de:
 - marcar las cookies de sesión con los atributos Secure (sólo se transmitirán por HTTPS) y HTTPonly (no serán accesibles desde JavaScript)
 - no utilizar contenido mixto (elementos servidos por HTTPS y por HTTP en la misma página)
 - es un error tremendamente frecuente
- Es imprescindible, también, el uso de HSTS

Contenido mixto: ejemplo



Configurando SSL/TLS GRACIAS POR Hacia la seguridad real...

TU ATENGLON Miguel Macias Enguidanos

miguel.macias@upv.es



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

XXXIV Grupos de Trabajo Bilbao, 27/11/2012

Por cierto...



- Lamentablemente no mencionan S/MIME (o, para no asustar a nadie: firmado y cifrado digitales)
- ¿Sabías que RedIRIS nos ofrece SCP?