GOBIERNO DE ESPAÑA
MINISTERIO DE HACIENDA Y ADMINISTRACIONES PÚBLICAS

SEGURIDAD







Guía de Comunicación Digital
para la Administración General del Estado

REQUISITOS ANTES DE TENER EL SITIO WEB

6. SEGURIDAD

Este quinto fascículo de la Guía de Comunicación Digital de la Administración General del Estado[1] recoge los aspectos a tener en cuenta a la hora de dotar a nuestro sitio web de medidas de seguridad adecuadas.

La necesidad de control de la seguridad en los sitios web nace de la evolución de los contenidos de los mismos, cada día se ofrecen más servicios web y posibilidades de realizar muchos trámites a través de internet.

Las medidas de seguridad de un Sitio Web deben tomarse tanto a nivel físico, almacenamiento del sitio *("hosting")* y del servidor como a nivel lógico durante el desarrollo del sitio.

Por lo que respecta a la seguridad física, se seguirán las especificaciones propias para sitios web que se recojan en los planes de seguridad generales de los sistemas de información.

En cuanto a la seguridad lógica, al desarrollar el sitio hay que identificar y analizar los requisitos de seguridad asociados a los sitios web con el objetivo de reducir las posibles amenazas de seguridad asociadas a ellos durante su diseño antes de su paso a producción. Será por tanto necesario tener herramientas de análisis de tráfico, herramientas de escáner de puertos, herramientas de detección de intrusiones, procedimientos de actuación definidos en caso de denegación de servicio, etc. Todas estas herramientas estarán actualizadas a las últimas versiones para que solventen las últimas vulnerabilidades descubiertas para dotar a los sitios web de la Administración de las medidas de seguridad adecuadas para que los ciudadanos los utilicen con confianza.

El grado de vulnerabilidad que puede presentar un Sitio Web depende directamente de las funcionalidades que el mismo ofrezca. Si el sitio está creado usando simplemente HTML (y en nuestras carpetas sólo tenemos archivos *.htm, .html, .css, .jpg y .gif*) entonces el peligro será mínimo. Por el contrario, si el Sitio Web está creado usando un sistema de *Server Side Scripting* (como lo son *PHP, ASP, JSP*, etc.) entonces existe la posibilidad de que aparezcan potenciales fallos de seguridad, sobre todo si en uno o más lugares del sitio hay formularios que permitan al usuario enviarnos datos (un formulario de contacto o un formulario de suscripción a un boletín).

[1] Los fascículos de la Guía de Comunicación digital de la AGE son: Aspectos Generales, Imagen Institucional, Multilingüismo, Accesibilidad, Seguridad, Aspectos de Comunicación, Tecnología Web 2.0 y Mejora y Mantenimiento.

Por último, si además de usar un lenguaje como *PHP o ASP* nuestro sitio usa bases de datos (como *MySQL, Oracle, SQL-Server*, etc.), entonces las posibilidades de ataques se multiplican, de igual manera que si utilizamos scripts y programas estándar dentro del sitio (como ser scripts de administración de contenidos, foros, galerías de fotos, programas de intercambios de enlaces, etc.).

En todo caso deberán siempre observarse las directrices y políticas de seguridad generales que existan en el Departamento y en general lo previsto en:

- la Ley 11/2007 de acceso electrónico de los ciudadanos a los servicios públicos
- el Real Decreto 1671/2009, de 6 de noviembre, por el que se desarrolla parcialmente la Ley 11/2007, de 22 de junio, de acceso electrónico de los ciudadanos a los servicios públicos.
- y en especial el Real Decreto 3/2010, de 8 de enero, (http://www.boe.es/aeboe/consultas/bases_datos/act.php?id=BOE-A-2010-1330) por el que se regula el Esquema Nacional de Seguridad en el ámbito de la Administración Electrónica.
- Título VIII del RD 1720/2007 "Medidas de seguridad en el tratamiento de datos de carácter personal".

La forma más eficaz de garantizar la seguridad de nuestros sitios web es la prevención, en este sentido, debemos dedicar todos los recursos a nuestro alcance para implantar medidas y acciones encaminadas a la prevención de vulnerabilidades y como consecuencia a evitar posibles ataques. Se deberán seguir las políticas generales de seguridad de la organización, mantener actualizado el software y herramientas de seguridad (antivirus, anti espías…), auditar permanentemente nuestras instalaciones, Respetar las normas establecidas en la organización en cuanto a permisos y contraseñas, realizar las copias de seguridad pertinentes, utilizar código seguro en la programación basándose en la metodología OWASP, realizar las transferencias de datos cifrados y utilizar archivos de configuración distribuida.

Por supuesto las medidas preventivas deben aplicarse desde las fases iníciales al diseñar y desarrollar nuestros entornos y aplicaciones web. Para ello recomendamos que para saber más sobre requisitos y verificaciones de seguridad durante el diseño y desarrollo de los entornos y aplicaciones web y sobre auditorías y análisis de vulnerabilidades posteriormente tras su puesta en producción se consulte el marco exhaustivo que sobre este ámbito se presenta en la Guía CCN-STIC 812 - Seguridad en Entornos y Aplicaciones Web del Centro Criptológico Nacional, que además ofrece una lista de comprobación para evaluar entornos y aplicaciones web de terceros antes de su adquisición.

No obstante las medidas preventivas, podría ocurrir que se produzca un ataque en nuestro entorno o aplicación web para lo cual en primer lugar deberemos identificar el tipo de ataque que hemos sufrido y a continuación aplicar medidas y realizar actuaciones para corregir los efectos producidos por el ataque.

Para ampliar información sobre aspectos de detección de ataques, actuación para minimizar los posibles daños una vez sufrido, podemos consultar la Guía CCN-STIC 403 – Guía de Gestión de Incidentes publicada por el Centro Criptológico Nacional en el portal del CCN-CERT.

Las recomendaciones contenidas en esta Guía van dirigidas al equipo encargado del desarrollo del sitio y deberán ir acompañadas por las medidas de seguridad de los responsables del "*hosting*" en el afán común de mantener el sitio con los niveles de seguridad necesarios.

Los objetivos que generalmente se persiguen con los ataques más frecuentes pueden agruparse en:

- Obtener información confidencial tanto del servidor como de la red a la que pueda dar acceso.
- Comprometer los equipos de los usuarios que visitan el sitio web atacado, creando una *botnet* o red de ordenadores infectados.
- Obtener direcciones de correo para el envío de *spam*.
- Abusar del ancho de banda contratado por los usuarios.
- Alojar páginas de *phishing* suplantando a otras entidades.
- Uso de la capacidad de procesamiento de los sistemas comprometidos.
- Uso del espacio web para alojar diversos contenidos con fines fraudulentos o maliciosos.
- Dañar la imagen del Organismo mediante acciones de *hacktivismo*

A continuación se aportan recomendaciones para evitar estos ataques o resolver sus consecuencias si, desgraciadamente, ya se han producido.

6.1. Inyección de código

Los fallos de inyección de código, tales como *SQL, OS, y LDAP*, ocurren cuando datos no confiables son enviados al intérprete del servidor como parte de un comando o consulta.

Los datos hostiles del atacante pueden engañar al intérprete al ejecutar comandos no intencionados o acceder a datos no autorizados.

Recomendado

- Para evitar este tipo de ataques prima la regla máxima sobre seguridad: Nunca confiar en los datos recibidos por el usuario.
- La modificación de parámetros de entrada por URL puede mostrar errores pero no descriptivos acerca de datos técnicos, es decir, se debe notificar al usuario de los errores producidos pero mediante mensajes personalizados. En ningún caso un mensaje de error del software o de la base de datos sobre la que se apoya el portal debe trascender hasta la interfaz del usuario.
- Los parámetros de entrada por URL serán correctamente filtrados y al modificarlos con texto en formato SQL no permitirá alterar la funcionalidad original.
- La modificación de parámetros de formularios puede mostrar errores pero no descriptivos acerca de datos técnicos, es decir, se debe notificar al usuario de los errores producidos pero mediante mensajes personalizados. En ningún caso un mensaje de error del software o de la base de datos sobre la que se apoya el portal debe trascender hasta la interfaz del usuario.
- Los parámetros de entrada en formularios serán correctamente filtrados y al modificarlos con texto en formato SQL no permitirá alterar la funcionalidad original.
- Cualquier parámetro recibido, ya sea por método GET o POST (u otro si se usan más comandos HTTP) debe ser filtrado para, o bien eliminar caracteres especiales o rechazar completamente su contenido si se detecta un contenido potencialmente peligroso.
- A menos que el contenido lo requiera explícitamente, se deberán filtrar todos los caracteres especiales que puedan ser tratados de forma diferente en lenguajes JavaScript, SQL o cualquiera que se esté utilizando para el funcionamiento del portal.
- Si los datos recibidos pueden ser devueltos al usuario en formato HTML se deberán filtrar los caracteres “<” y “>” o si es posible eliminar completamente o filtrar todas las etiquetas HTML que contenga el texto a excepción de las de formato de mensaje.
- Se recomienda igualmente aplicar las características propias del motor de los lenguajes aplicados al portal, activando el escape de caracteres. Para ello, se deben utilizar las citadas funciones.
- Si los datos recibidos pueden ser almacenados, aunque sea de forma temporal en una base de datos se deberán filtrar los caracteres de comillas simples y dobles y posiblemente alguno más en función del gestor de base de datos utilizado. Es recomendable utilizar la documentación del gestor de base de datos que siempre suelen acompañar una información al respecto del juego de caracteres soportado.

- Se recomienda el uso de cortafuegos de aplicaciones web (WAF).

6.2. Secuencia de Comandos en sitios Cruzados (Cross Site Scripting *XSS*)

Las vulnerabilidades XSS ocurren cada vez que una aplicación toma datos no confiables y los envía al navegador web sin una validación y codificación apropiada.

XSS permite a los atacantes ejecutar secuencia de comandos en el navegador de la victima los cuales pueden secuestrar las sesiones de usuario, destruir sitios web, o dirigir al usuario hacia un sitio malicioso.

R Recomendado

- Los parámetros de entrada por URL serán correctamente filtrados y al insertar texto en formato script (como JavaScript) no se mostrará de vuelta en la página.
- Los parámetros de entrada en formularios serán correctamente filtrados y al insertar texto en formato script (como JavaScript) no se muestra de vuelta en la página.
- Al igual que con los fallos de tipo inyección, los problemas derivados de XSS se pueden evitar con un correcto filtrado de todos los datos de entrada procedentes de los usuarios.
- Para evitar potenciales problemas con el JavaScript (u otros lenguajes de script), se deberán filtrar los caracteres “<” y “>” o si es posible eliminar completamente o filtrar todas las etiquetas HTML que contenga el texto a excepción de las de formato de mensaje. También es posible mostrar los caracteres problemáticos con una codificación equivalente en entidades mediante su correspondiente código como &#xxxx; o & en lugar de &.
- Se recomienda el uso de cortafuegos de aplicaciones web (WAF), que además sea capaz de generar una respuesta activa frente a posibles ataques. Dicha respuesta, debe incluir en todo caso la posibilidad de excluir las direcciones IP´s correspondientes a la conexión originaria del ataque.

6.3. Pérdida de autenticación y Gestión de Sesiones

Las funciones de la aplicación relacionadas a autenticación y gestión de sesiones son frecuentemente implementadas de manera incorrecta, permitiendo a los atacantes comprometer contraseñas, llaves, "*token*" de sesiones, o explotar otras fallas de implementación para asumir la identidad de otros usuarios.

R Recomendado

- No habrá referencias a identificadores de sesión en la URL.
- En caso de que un usuario pueda iniciar sesión, por defecto su validez debe ser como máximo hasta el cierre del navegador (o la duración que haya especificado el usuario).
- En caso de que un usuario pueda iniciar sesión, el usuario tiene la opción de cerrarla en cualquier momento.
- En caso de que un usuario pueda iniciar sesión, debe realizarse siempre mediante conexiones seguras y cifradas (protocolo HTTPS).
- En caso de que un usuario pueda iniciar sesión, dos sesiones independientes deben tener identificadores distintos y no relacionados entre sí.
- Las sesiones deben ser tratadas con el máximo cuidado posible. El secuestro de sesiones mediante el filtrado de las comunicaciones es un ataque bastante común de consecuencias muy graves ya que permite la suplantación completa de un usuario con los mismos roles de seguridad y permisos de acceso que tuviese el usuario legítimo.
- Su creación, a través de la autenticación de credenciales procedentes del usuario debe realizarse siempre utilizando un canal seguro para evitar la interceptación de las comunicaciones, como por ejemplo HTTPS y el uso de certificados de confianza.
- A partir de ese momento, toda la identificación del usuario frente al portal debe reducirse a la mínima expresión, siendo recomendable un único identificador aleatorio alfanumérico almacenado en una cookie de forma que sea transparente para el usuario.
- Dicha cookie debe cumplir que:
  - Sea lo suficientemente larga y aleatoria para que sea imposible averiguar el algoritmo de creación y secuestrar sesiones. Normalmente su generación está delegada según la tecnología que se esté usando (*PHP, Java, ASP…)* pero se suele poder reforzar la modificación de la semilla para la generación de números pseudoaleatorios.

- Su caducidad debe ser a nivel de sesión a menos que se habilite una opción para especificarla. En ese último caso los valores recomendados no deberían superar las 3 horas de inactividad del usuario excepto en casos justificados.
- La finalización de la sesión debe destruir cualquier cookie que se haya generado en el inicio de la misma, independientemente de la caducidad que se haya marcado.
- Debe estar marcada mediante los atributos “secure” y “httpOnly” para que los navegadores modernos lo interpreten correctamente y añada opciones de seguridad extra para que las cookies sensibles tengan el mínimo riesgo.

6.4. Referencia Directa Insegura a Objetos

Una referencia directa a objetos ocurre cuando un desarrollador expone una referencia a un objeto de implementación interno del servidor, tal como un fichero, directorio, o base de datos.

Sin un chequeo de control de acceso u otra protección, los atacantes pueden manipular estas referencias para acceder a datos no autorizados.

Recomendado

- Si hay valores personales de usuario por URL, su cambio no debe permitir visualizar datos de otros usuarios.
- Si hay valores personales de usuario por "cookies", su cambio no debe permitir visualizar datos de otros usuarios.
- A la hora de realizar el diseño de un portal es recomendable aplicar siempre la directriz de denegación por defecto. El acceso a cualquier recurso estará prohibido a menos que se autorice explícitamente a un usuario que pueda visualizarlo. De este modo se oculta mucha información a un posible atacante y es más robusto frente a errores humanos que pudiesen desvelar datos sensibles.
- También es recomendable minimizar el tránsito de datos entre el usuario y el servidor para evitar puntos de entrada a posibles fallos, evitando enviar más datos de los necesarios en las peticiones (ya sean GET o POST) o almacenar datos adicionales en las cookies relacionadas con el dominio del portal.

6.5. Falsificación de Peticiones en sitios Cruzados (Cross-site Request Forgery *CSRF*).

Un ataque *CSRF* obliga al navegador de una víctima autenticada a enviar una petición *HTTP* falsificada, incluyendo la sesión del usuario y cualquier otra información de autenticación incluida automáticamente a una aplicación web vulnerable.

Esto permite al atacante forzar en el navegador de la victima a generar pedidos que la aplicación vulnerable piensa que son peticiones legítimas provenientes de la víctima.

Recomendado

- No se crearán peticiones por URL que realicen una acción funcional completa.
- No se crearán peticiones por formulario que realicen una acción funcional completa.
- Esta clase de ataques de invocación de peticiones fraudulentas es muy común y sencillo de explotar y por desgracia rara vez los portales se encuentran protegidos contra él.
- Es necesario poder distinguir entre una petición legítima del usuario y una fraudulenta de un atacante realizada en nombre del usuario. La medida de prevención más simple, aunque no muy efectiva, es la comprobación del origen de la petición a través de las cabeceras HTTP que envían los navegadores. Lamentablemente dicho origen también es fácilmente suplantable.
- La solución ideal consiste en crear “tokens” de petición de modo que cualquier acción, ya sea en enlace o en solicitud contra el portal, incluya un "token" generado dinámicamente y que el portal al recibirla sea capaz de comprobar su validez y autenticidad.
- Existen múltiples maneras de implementar una solución como ésta, pero la más sencilla es mantener en un sistema de almacenamiento como una base de datos, una caché de "tokens" generados aleatoriamente en tiempo de ejecución asociados a cada petición que se muestre al usuario, de forma que pueda comprobarlo fácilmente una vez recibidos. Evidentemente los token serían de un único uso y desechables.
- El escenario ideal sería que no se pudiera invocar ninguna acción desde un entorno ajeno al portal, pero las de consulta no serían peligrosas a menos que estén en combinación con otro fallo de seguridad de inyección o XSS.

6.6. Defectuosa Configuración de Seguridad

Una buena seguridad requiere tener definida e implementada una configuración segura para la aplicación, marcos de trabajo, servidor de aplicación, servidor web, base de datos y plataforma.

Todas estas configuraciones deben ser definidas, implementadas y mantenidas, ya que por defecto son consideradas como no seguras. Esto incluye mantener todo el software actualizado incluidas las librerías de código utilizadas por la aplicación.

Para el bastionado de las distintas arquitecturas implementadas, el Centro Criptológico Nacional, publica y revisa periódicamente la Serie de Guías STIC que pueden ser descargadas del portal del CCN-CERT (https://www.ccn-cert.cni.es)

R Recomendado

- Sólo serán accesibles los servicios relevantes en los puertos del servidor para ofrecer la funcionalidad deseada.
- Sólo serán accesibles los servicios relevantes por URL en el servidor para ofrecer la funcionalidad deseada.
- Tan importante es el desarrollo como el software sobre el que se apoya, tanto en librerías que ofrecen funcionalidades adicionales como con los servidores a los que accede. Es recomendable leer la documentación de todos aquellos programas adicionales que se utilicen para realizar una configuración de seguridad efectiva ya que la mayoría ofrecen por defecto unos valores pobres. Por ejemplo, el servidor de base de datos MySQL no tiene asociada una contraseña para su usuario administrador.
- Del mismo modo, para aquellas librerías o módulos de los que haga uso la aplicación deberán estar correctamente actualizados. Es común que se encuentren vulnerabilidades en ellas y sean corregidas con diligencia por los responsables de las mismas. Es responsabilidad del equipo de desarrollo hacer uso de las últimas versiones de dichas librerías y adaptar la aplicación a cada nueva versión correctiva que aparezca.
- Se recuerda que deben aplicarse, al menos, las reglas bastionado de un servidor, a saber: mínimos privilegios posibles, mínimo punto de exposición, y defensa en profundidad.
- Igualmente, dichas medidas de seguridad deben estar relacionadas entre sí, actuando de forma global para evitar que se soapen o interrumpan entre ellas. Ejemplo: No es de utilidad el uso de un IDS/IPS que no pueda analizar el trafico cifrado con SSL, o un IDS/IPS que sea incapaz de interactuar con el firewall para bloquear una posible intrusión bloqueando la conexión malintencionada.

6.7. Almacenamiento Criptográfico Inseguro

Muchas aplicaciones web no protegen adecuadamente los datos sensibles, tales como tarjetas de crédito, NSSs y credenciales de autenticación con mecanismos de cifrado o "*hashing*".

Los atacantes pueden modificar o robar tales datos protegidos inadecuadamente para conducir robos de identidad, fraudes de tarjeta de crédito u otros crímenes.

Recomendado

- En caso de permitir inicio de sesión a los usuarios, la recuperación de la contraseña no expone ningún dato personal como dirección de correo y sólo permite modificarla sin poder conocer su valor original.
- Cualquier información sensible que se encuentre almacenada en el sistema deberá estar cifrada. Si se trata únicamente de datos de comprobación, es decir, que no van a ser transmitidos a ninguna otra entidad o sistema como puede ser una contraseña de acceso, deberá ser cifrada con un algoritmo hash preferiblemente SHA1. Si se trata de datos sensibles pero necesarios para interactuar con otras aplicaciones o servicios como puede ser un número de tarjeta de crédito, deberán estar correctamente cifrados utilizando algún sistema de criptografía simétrica convenientemente implementado para que la clave secreta nunca salga de la aplicación.

6.8. Falla de Restricción de Acceso a *URL*

Muchas aplicaciones web verifican los privilegios de acceso a *URL*s antes de generar enlaces o botones protegidos. Sin embargo, las aplicaciones necesitan realizar controles similares cada vez que estas páginas son accedidas o los atacantes podrán falsificar *URL*s para acceder a estas páginas igualmente.

Recomendado

- Los enlaces a las páginas de administración se encuentran protegidos en lugar de sólo ocultos.
- Aunque es recomendable ocultar de la vista enlaces protegidos para evitar tentaciones, es insuficiente limitarse únicamente a esa protección. Cualquier página que sea accesible a través de una dirección URL debe realizar una comprobación de privilegios en su inicio y en caso de no cumplir con lo exigido, mostrar un error.

6.9. Protección Insuficiente en la Capa de Transporte

Las aplicaciones frecuentemente fallan al autenticar, cifrar y proteger la confidencialidad e integridad de tráfico de red sensible. Cuando esto ocurre, es debido a la utilización de algoritmos débiles, certificados expirados, inválidos o sencillamente no utilizados correctamente.

R Recomendado

- Todas las comunicaciones que contengan datos sensibles para el usuario deberán hacer uso de cifrados como SSL (HTTPS).
- Todas las cookies de sesión tienen el atributo "secure" activado de forma que el navegador nunca las transmita en claro.
- El certificado de seguridad del servidor será legítimo, firmado por una autoridad de certificación reconocida, con validez vigente y que cubra todos los nombres de dominio utilizados por la aplicación.
- La interceptación de las comunicaciones es un riesgo muy real y más habitual de lo que se puede pensar. Debe presuponer en el desarrollo de una aplicación web que cualquier dato que el usuario envíe o que se envíe hacia el usuario puede ser observado por un posible atacante.
- El escenario ideal y la configuración por defecto debería ser cifrar todas las comunicaciones, ya no sólo para los datos personales procedentes del usuario, sino también porque la información consultada puede estar sujeta a restricciones de visibilidad.
- Se recomienda actualizar regularmente el software de cifrado SSL, para evitar de esta forma ser objeto de ataques basados en vulnerabilidades de software conocidas y/o errores en la implementación del sistema de cifrado (Ejemplo: vulnerabilidades conocidas de versiones antiguas de “openssl”).
- En caso de ser posible, se debe forzar siempre la conexión y la navegación HTTPS. De esta manera se podrán evitar ataques mediante el método “Man in the Middle” que fuercen conexiones HTTP (Ejemplo: “sslstrip”).

6.10. Redirecciones y Reenvíos no Validados

Las aplicaciones web frecuentemente redirigen y reenvían a los usuarios hacia otras páginas o sitios web y utilizan datos no confiables para determinar la página de destino.

Sin una validación apropiada, los atacantes pueden redirigir a las víctimas hacia sitios de "*phishing" o "malware*" o utilizar reenvíos para acceder a páginas no autorizadas.

Recomendado
❑ En caso de que la aplicación realice redirecciones por URL, éstas no deberán admitir cualquier nombre o destino a menos que su funcionalidad así lo requiera explícitamente.

6.11. Eliminación y limpieza de los metadatos de un documento

Los metadatos son un arma de doble filo, útil para clasificar y organizar los contenidos de la propia librería de documentos, y peligrosos por la cantidad de información sensible que transmiten sin que tengamos control sobre ella.

La mayoría de editores de documentos son capaces de leer y manipular los metadatos de los archivos, algo muy útil para clasificarlos. Sin embargo, los metadatos también suponen un riesgo para nuestra privacidad, especialmente por lo poco que pensamos en ellos.

Recomendado
❑ Es conveniente revisar y eliminar los metadatos de los documentos publicados en la web, hay varias herramientas automáticas públicas destinadas a recopilar metadatos en documentos disponibles en Internet. Existen igualmente multitud de herramientas para eliminar metadatos que contienen información sensible de los ficheros.

6.12. Canal de comunicación para la gestión de incidentes

De acuerdo con sus responsabilidades, el CCN-CERT ofrece un sistema de apertura y gestión de incidentes de seguridad, que permite a los usuarios comunicar los incidentes que se produzcan y realizar el seguimiento hasta su resolución.

Si se produce un incidente de seguridad, la vía para realizar dicha comunicación es a través del portal del CCN-CERT o por correo electrónico mediante la cuenta incidentes@ccn-cert.cni.es.

Si se desea denunciar o informar de un posible delito derivado de un incidente de seguridad puede realizarse a través del Grupo de Delitos Telemáticos de la Guardia Civil[2] o de la Brigada de Investigación Tecnológica de la Policía Nacional[3]. En el sitio web de ambos portales, pueden encontrarse consejos y buenas prácticas que pueden incorporarse a la web a modo de sugerencia para los usuarios.

[2] https://www.gdt.guardiacivil.es/webgdt
[3] http://www.policia.es/org_central/judicial/udef/bit_quienes_somos.html