Construcción del plan de pruebas con OWASP orientado a aplicaciones web

Breve descripción:

Para la construcción de un sistema de información por procesamiento electrónico de datos orientado a la web es muy importante implementar los mecanismos que brindan la seguridad de la aplicación, el presente componente tiene como objetivo elaborar un plan de pruebas de seguridad utilizando como metodología la Guía de referencia de pruebas de OWASP que pretende satisfacer los requerimientos de seguridad que se presentan.

Octubre 2023

Tabla de contenido

Introducción

Le damos la bienvenida al componente formativo denominado Construcción del plan de pruebas con OWASP orientado a aplicaciones web, el cual hace parte del programa de formación técnico en “Seguridad en aplicaciones web”, para lo cual se invita a observar el siguiente video:

Video 1. Construcción del plan de pruebas con OWASP orientado a aplicaciones web

Enlace de reproducción del video

Síntesis del video: Construcción del plan de pruebas con OWASP orientado a aplicaciones web

Dentro el desarrollo de “software”, existe un conjunto de pruebas que son muy importantes para la aplicación.

En este caso, vamos a tratar el tema de las pruebas de seguridad, y para ello se hablará del esquema o guía de referencia, denominado OWASP, la cual da a conocer un conjunto de guías necesarias para llevar a cabo este tipo de pruebas.

En este componente, nos vamos a centrar en la realización de un plan de actividades, definir una estrategia y forma de revisar y prevenir los diferentes ataques.

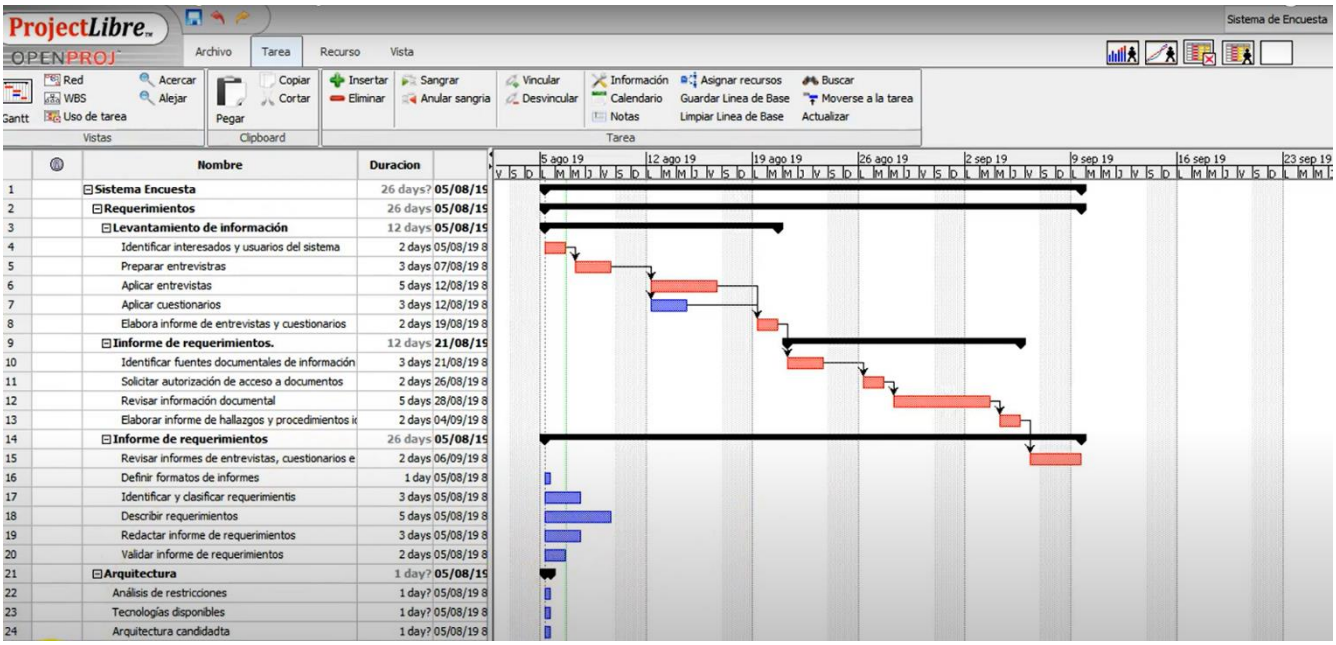
1. Plan de actividades

Es un documento que recoge un conjunto de tareas necesarias para la consecución de una acción u objetivo concreto. Antes de llevar a cabo la tarea de planificar un proyecto, es conveniente hacer un plan de actividades, en el cual se contemplará lo necesario que lleve a organizar las pruebas de seguridad de una aplicación web con base en la guía OWASP ("Open Web Application Security Project").

Haga clic aquí para conocer la Guía OWASP.

Se recomienda para la elaboración del plan de actividades, la creación de un diagrama de Gantt el cual se compone de dos grandes partes de la siguiente manera:

Figura 1. Diagrama de Gantt generado con Open Project



En su parte izquierda se describe la lista de tareas a llevar a cabo.

En la parte de derecha hay un cronograma con barras que representa el trabajo a realizar.

Así mismo, dentro del diagrama es importante definir:

- Duración normalmente dada en días incluso en horas.
- Las fechas de inicio y de fin de cada tarea.
- Las actividades predecesoras y antecesoras.
- Actividades indentadas.
- Recurso asociado por actividad.

La importancia de un diagrama de Gantt radica en la posibilidad de contar constantemente con una visualización global del proyecto, incluso de planes complejos, mostrando el impacto que un cambio puede generar en una sola actividad en todo el proyecto permitiendo generar retroalimentación a los roles interesados.

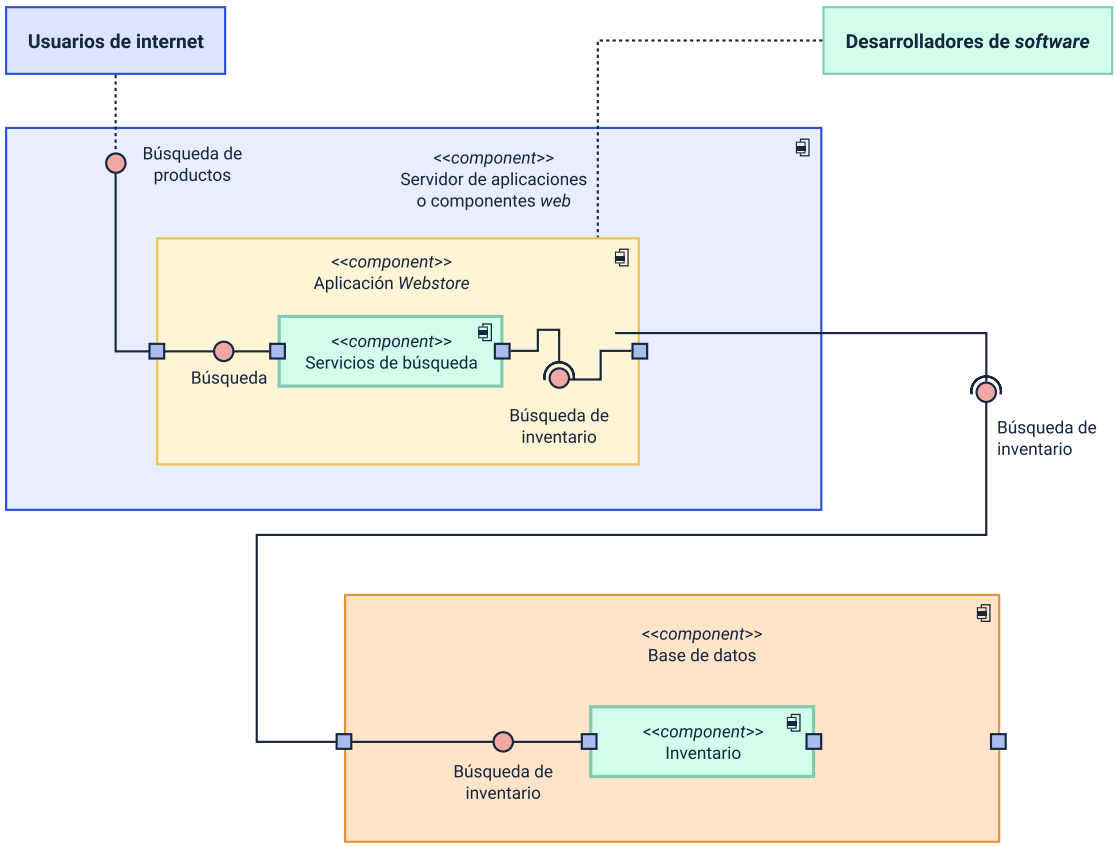
Para diseñar un diagrama de Gantt se recomienda la utilización de herramientas de software que permiten manejar de una manera ágil el seguimiento y verificación de su cumplimiento, incluso enmarcado dentro de un proyecto que puede considerarse más grande de marcando una hoja de ruta; algunas herramientas recomendadas son:

- Instagantt
- Open Project
- Teamgantt
- ExcelGanttpro

1.1. Objetivos

En este punto se deben plantear los objetivos generales a cumplir en el proyecto teniendo en cuenta que la guía OWASP comprende un gran número de componentes a los cuales se puede aplicar un proceso de análisis de vulnerabilidades como lo son por ejemplo, los componentes tipo API (interfaz de programación de aplicaciones) que normalmente se construyen bajo el concepto de "`web services`", de validación de datos de autorización y autenticación, de verificación y utilización de marcos en diferentes organizaciones, hasta llegar a la verificación de infraestructura, redes con sus configuraciones y talento humano.

Figura 2. Objetivos del plan de pruebas



Los objetivos se pueden plantear de la siguiente manera:

- Análisis del componente de aplicaciones web que corresponden a herramientas de “software” que funcionan vía internet y se convierten en soluciones que están “expuestas” al público de manera masiva.
- Establecer hacia quién va dirigido dicho análisis, entendiéndose que esas pruebas según la guía pueden estar orientadas a “testers” de “software” que son personas que trabajan en el marco de metodologías y pretenden dar cumplimiento a estándares de calidad unidos a una metodología en el marco del desarrollo de “software”.

Estos objetivos están orientados a:

Especialistas de seguridad que, en su gran mayoría, se encuentran en la capa de administración de la infraestructura que soporta el funcionamiento del componente web en un ecosistema tecnológico y también a desarrolladores de “software” que son las personas encargadas de escribir las sentencias de código fuente que conforman el componente.

1.2. Limitaciones y facilidades

Cuando se aborda la creación de un plan de actividades es muy importante recopilar la mayor cantidad posible de información de una organización. Esta debe estar orientada hacia la consecución de un contexto general de la situación actual de la empresa que va a utilizar o que está utilizando la aplicación web y que redunda en un análisis organizacional global que permita ubicar de una manera ágil los posibles recursos o la carencia de ellos.

Análisis organizacional: todas las empresas, sin importar su tamaño, tienen ciertas limitaciones que pueden representar un obstáculo para alcanzar los objetivos planteados.

Estas limitaciones son dificultades que pueden presentarse durante la ejecución del plan de trabajo y que se pueden superar, en muchas ocasiones, gracias a las facilidades con las que se cuenta.

Es importante listar e identificar esas limitantes y facilidades en los siguientes temas:

- Documentación existente y facilidad de acceso.
- Tecnologías.
- Infraestructura.
- Redes.
- Facilidad de acceso.
- Componentes.
- Identificación del personal encargado con sus respectivos roles y responsabilidades.
- Grado de conocimiento en la organización acerca del proceso de seguridad informática.

La importancia de este análisis radica en las posibilidades de recomendar a futuro, o durante la ejecución del plan de pruebas, mecanismos, herramientas de "software", prácticas, personal necesario, estrategias, tecnologías de carácter libre o

incluso de pago que ayuden en la consecución de los objetivos y que eliminen la existencia de brechas que antes no se han tenido en cuenta.

A continuación, se representa lo anteriormente mencionado.

a. **Facilidades:**

- Tecnológicas.
- Recursos logísticos.
- Talento humano.
- Recursos económicos.
- Capacitación.

b. **Análisis organizacional global:**

- Organigrama.
- Documentación.
- Misión.
- Visión.

c. **Limitaciones:**

- Aplicación de estándares (OWASP).
- Ecosistemas tecnológicos.
- Ambiente de desarrollo.
- Ambiente de pruebas.

- Ambiente de producción.

1.3. Fijar las metas y objetivos particulares

Para este apartado es importante definir que los objetivos serán guiados por la metodología OWASP y las metas a cumplir en este proceso son:

- Aplicar el estándar OWASP y el manual de pruebas de "pentesting" propuesto para determinar el plan de pruebas que se aplicarán sobre el sistema con el enfoque de análisis de seguridad en las aplicaciones web.
- Determinar las pruebas de la metodología OWASP de seguridad en aplicaciones web para los 10 casos de vulnerabilidades más conocidos y definidos en el punto de la estrategia de este proceso.
- Determinar los artefactos de pruebas a utilizar de acuerdo con la metodología OWASP de seguridad en aplicaciones web.

El término "pentesting" se refiere a pruebas de carácter complejo que involucran un proceso de identificación de la infraestructura, objetivos, vulnerabilidades y cómo podrían estas explotarse para causar el daño por parte de un atacante, y con los resultados obtener una adecuada protección.

1.4. Definir equipo de trabajo

Los mejores equipos de trabajo son los que se estructuran correctamente, saben comunicarse e interaccionan entre sí; la motivación, participación, organización, compromiso, confianza, objetivos comunes y resolución de problemas son las siete

características del trabajo en equipo fundamentales. De ahí la importancia de tomarse el tiempo necesario para listar a las personas indicadas a quienes se les van a asignar las actividades en el plan de trabajo para interactuar con ellas.

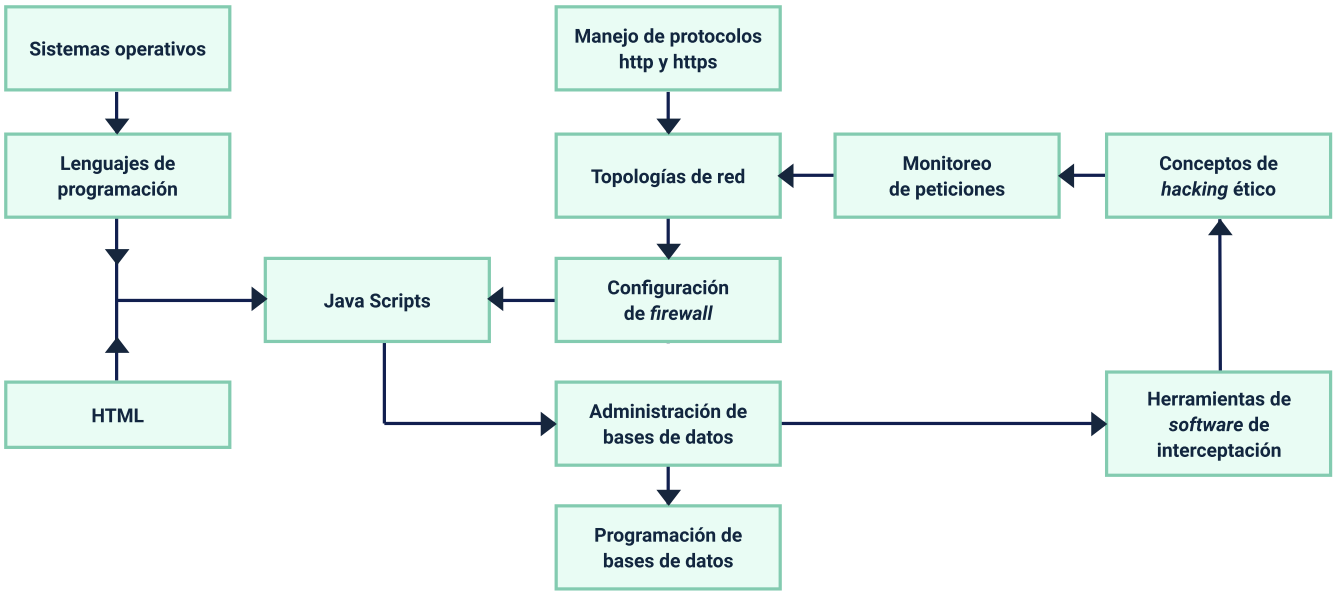
No obstante que este tipo de pruebas están orientadas a desarrolladores de "software", es importante tener en cuenta otros tipos de roles que pueden interactuar dentro de un equipo de trabajo como los administradores de red y plataformas, puesto que estos son los encargados de otorgar permisos en usuarios, definir tipologías de red, manejo de configuración de reglas de "firewalls", acceso a servidores de aplicaciones sistemas de autenticación entre otros, sobre todo para la parte en la cual consiste en contrastar los resultados obtenidos de las pruebas contra lo que realmente existe en el ecosistema tecnológico analizado en el que funciona una aplicación web.

1.5. Definir responsabilidades en el equipo

Los roles y responsabilidades se definen a nivel de la organización y del proyecto y el trabajo será conocer estos dos ítems del equipo en diferentes momentos de desarrollo de las actividades. Para ello, contar con una estructura de equipo sólida en el nivel organizacional es fundamental y es lo primero que se debe apropiar para apoyar en la resolución de posibles brechas en esta área.

El principal rol que es el del desarrollador, debe contar como mínimo con los siguientes conocimientos:

Figura 3. Conocimientos del desarrollador gráfico



Los conocimientos mínimos que debe tener el desarrollador son:

- Sistemas operativos.
- Lenguajes de programación.
- HTML.
- Java “Scripts”.
- Manejo de protocolos http y https.
- Topologías de red.
- Configuración de “firewall”.
- Administración de bases de datos.
- Programación de bases de datos.
- Monitoreo de peticiones.

- Conceptos de “haking” ético.
- Herramientas de “software” de interceptación.

1.6. Crear una estrategia

El procedimiento mediante el cual se determina para este caso es el análisis de vulnerabilidades de una aplicación web, y este corresponde a la verificación punto por punto de los 10 casos de ataques más comunes reportados por los informes que cada cierto tiempo destaca OWASP en dichos ambientes web, los cuales son:

- **Inyección (“Injection”).** Este ataque se produce cuando datos no confiables son enviados a un intérprete (ya sea del sistema operativo, de una base de datos o cualquier intérprete) como parte de un comando o consulta.

  Los datos hostiles introducidos por el atacante pueden engañar al intérprete haciendo que se ejecuten comandos no intencionados o accediendo a información sobre la que no se está autorizado. Uno de los ejemplos más significativos de este tipo de ataque es SQL “Injection”.

- **Pérdida de autenticación y gestión de sesiones.** Hay ciertos datos que deben estar cifrados, como credenciales de acceso, datos bancarios, información confidencial de la empresa, etc., ya que aparte de que la ley lo exija, lo que un ciberdelincuente pueda hacer con los datos puede ser catastrófico para la empresa.

- **Secuencia de comandos en sitios cruzados (XSS).** Este tipo de ataque ocurre cuando se obtienen datos no confiables y se envían directamente al navegador web. Esto provoca que se puedan ejecutar comandos no deseados en el navegador del usuario.

  Estos comandos pueden obtener desde las credenciales de acceso del usuario como instalar ciertos programas maliciosos.

- **Referencia directa insegura a objetos.** Este tipo de ataque ocurre cuando no se controlan los accesos a recursos sobre los que un usuario no debería tener acceso. En las aplicaciones, la mayoría de las veces, existen distintos usuarios y cada usuario tiene una serie de recursos sobre los que tiene acceso y otros sobre los que no debería tener acceso.

- **Configuración de seguridad incorrecta.** Este tipo de ataque ocurre cuando se han realizado malas configuraciones en los servidores de las aplicaciones, en las bases de datos o en la configuración del propio sistema operativo.

  Es importante tener todo el "software" actualizado con la última versión disponible y que todas las librerías o "frameworks" que use la aplicación también estén actualizadas.

- **Exposición de datos sensibles.** Este tipo de ataque ocurre cuando se puede acceder de forma fácil a datos de carácter sensible almacenados en la aplicación.

Cuando, por ejemplo, se almacenan las credenciales de los usuarios sin codificar o si la comunicación del servidor no es segura a la hora de realizar un pago con una tarjeta de crédito.

- **Ausencia de control de acceso a funciones.** Este tipo de ataque ocurre cuando se accede a funciones del servidor sobre las que un usuario no debería tener permiso.

  Por ejemplo: el servicio de listado de usuario solo debería estar disponible por la aplicación, pero cualquier usuario puede también acceder a esta información.

- **Falsificación de peticiones en sitios cruzados.** Este tipo de ataque ocurre cuando se realizan peticiones HTTP falsificadas del ordenador de la víctima a una aplicación web vulnerable.

- **Utilización de componentes con vulnerabilidades conocidas.** Este tipo de ataque ocurre cuando se emplean librerías o "frameworks" que contienen vulnerabilidades.

  Es por esto que es importante actualizar estos componentes o revisar el historial de revisiones para comprobar las mejoras de seguridad implementadas.

- **Redirecciones y reenvíos validados.** Este tipo de ataque ocurre cuando, al redireccionar al usuario a otra página, no se comprueba que el destino sea válido.

Es por esto que se puede redirigir a un usuario a una página que contenga contenido malicioso para robar las credenciales de dicho usuario.

1.7. Establecer los plazos

Mediante un cronograma se deben establecer los tiempos para cada actividad con fecha de inicio y fecha de finalización y cada una debe ser organizada en orden lógico de aplicación.

En el siguiente video se conocerá más detalladamente qué se requiere establecer en un cronograma:

Video 2. Establecer los plazos para cada actividad

Enlace de reproducción del video

Síntesis del video: Establecer los plazos para cada actividad
Como se viene hablando de cronograma, y que este representa, dentro de un diagrama de Gantt, el campo de la duración específica del tiempo estimado de cada tarea, es muy importante tener en cuenta que el cronograma debe ser ordenado en la asignación de tareas, al igual que contar con un seguimiento periódico para que sea efectivo, lo ideal es apoyarse con las herramientas de "software" tipo Excel, Open Project, entre otros. Cada cronograma debe contener unos criterios sencillos que permitan de manera asertiva, organizar el trabajo. Pero también se debe recordar, que todos los cronogramas cuentan con un nivel de complejidad y linealidad cronológica, que hacen parte de sus características principales. A continuación, detallaremos cada uno de ellos. La complejidad se refiere a aquellos proyectos demasiado grandes que cuentan con un equipo numeroso y contiene gran número de tareas y recursos. En el diagrama de Gantt es necesario diagramar todas las actividades, pero al contar con proyectos de esta magnitud, se hace muy difícil realizar esta tarea, lo cual aumenta el grado de complejidad. Diagramar decenas o cientos de actividades dentro de un diagrama de Gantt, llega a causar confusión y puede costar mucha dificultad intentar leer la información. Por ello es fundamental dar valor a las diferentes tareas y sus subtareas indentadas,

previo a la realización del diagrama, al igual que con la asignación de algún responsable que les haga el seguimiento a los detalles de cada tarea.

Para que un cronograma llegue a ser eficaz bajo el diagrama de Gantt, se debe planear el conjunto de actividades del proyecto de manera lineal, desde un comienzo hasta su final. Esto se genera cuando se conocen con anticipación, los posibles resultados esperados y las acciones necesarias que permitirán alcanzarlo.

Los anteriores conceptos son bastante evidentes cuando se trata de proyectos conocidos, pero en proyectos de desarrollo de "software" resulta mucho más complicado cuando se trata de aplicar el ciclo de vida por medio de la gestión de desarrolladores web involucrados a un proyecto.

1.8. Determinar los recursos necesarios

Centrado en la estrategia definida en el punto anterior para lograr la determinación de los recursos necesarios, se deben observar las condiciones dentro y fuera de la organización, a saber:

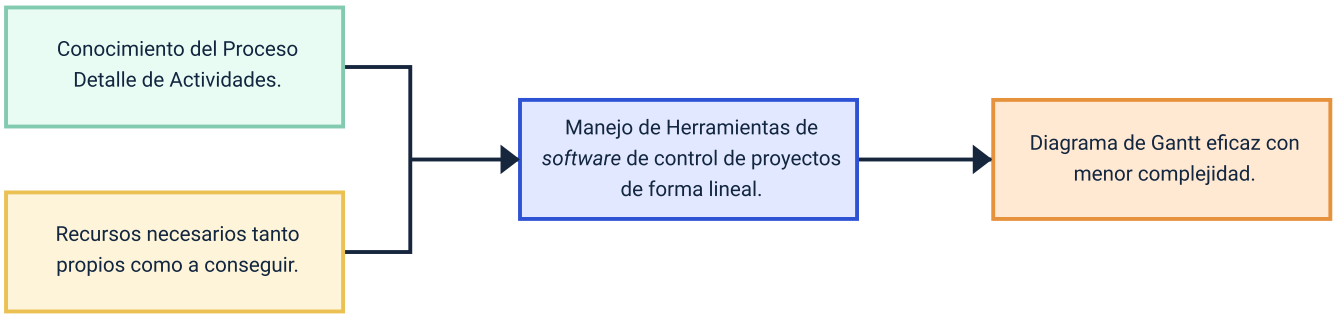
- Examina la base de recursos disponibles (tecnológicos, organizacionales y de documentación).
- Determina cómo se gestionan y relacionan esos recursos dentro de la organización. Esto servirá luego para tomar decisiones e integración de esos recursos.
- Comprueba qué recursos faltan para la puesta en marcha del proyecto.

- Evalúa qué estrategias se asumirán para terminar de dotar a la organización con los recursos necesarios. Esto siembra las bases para los planes de adquisición, procesos de innovación y desarrollo.
- Analizar si los recursos exigidos por el proyecto, no desestabilizan ni ponen en riesgo a la organización.

Se debe considerar que la determinación de los recursos necesarios para un proyecto debe contemplar cierto margen de flexibilidad y evolucionar a la par de la intervención, el comportamiento del entorno y las decisiones del cliente.

Dicho de otro modo, lo que no parece indispensable en un principio puede serlo más adelante.

Figura 4. Determinación de los recursos necesarios



Los recursos son:

- Conocimiento del proceso y detalle de actividades.
- Recursos necesarios tanto propios como a conseguir.
- Manejo de herramientas de “software” de control de proyectos de forma lineal.

- Diagrama de Gantt eficaz con menor complejidad.

1.9. Medir los resultados

Con el ánimo de evaluar el éxito o fracaso del proyecto se proponen, como mínimo, medir los siguientes indicadores:

- **Alcance.** Son los resultados que se esperan del proyecto y el conjunto de labores que se deben realizar para llegar a esos resultados planteados.
- **Calendario.** Corresponde a la línea de tiempo del proyecto. Abarca no solo la fecha de cierre final, sino también los hitos importantes y los plazos de las tareas a lo largo del proceso.
- **Presupuesto**. Es el valor del plan de pruebas, allí se deben tener en cuenta las siguientes preguntas: ¿cuánto es lo estimado para completar el alcance del plan? y ¿cuáles son los fondos que se están invirtiendo en el plan?
- **Logro de los objetivos del negocio.** Es la forma en que el proyecto funcionó comparando con los casos del negocio. Se debe dar respuesta a las siguientes preguntas: ¿se consiguieron los beneficios esperados (tanto tangibles como intangibles)? y ¿se obtuvo la ganancia esperada de acuerdo con la inversión (ROI - tasa de retorno de la inversión)?
- **La satisfacción del cliente.** Se refiere al grado en que los resultados del proyecto cumplen o superan las expectativas; esto abarca la calidad de las entregas, la experiencia general del cliente, el servicio al cliente y la

comunicación entre las partes interesadas internas y externas a lo largo del ciclo de vida del proyecto.

2. Artefactos de pruebas

Son los mecanismos que se centran en recoger toda la información como sea posible sobre la aplicación web objetivo y conocer la manera cómo opera. La recopilación de información es un paso fundamental y necesario en una prueba de intrusión.

Esta prueba corresponde al reconocimiento pasivo de información que se debe realizar sin intervenir los servidores o que estos generen "logs" de seguridad. Esta tarea normalmente se puede llevar a cabo de muchas formas. Utilizando herramientas de acceso público motores de búsqueda, "scanner", enviando peticiones HTTP simples, o peticiones especialmente diseñadas, es posible forzar a la aplicación a filtrar información al exterior con mensajes de error devueltos, o revelar las versiones y tecnología en uso por la aplicación web.

2.1. Metodología

La metodología a utilizar es OWASP, la que comprende las siguientes fases enmarcadas dentro del ciclo de vida del desarrollo de "software", para lo que se recomienda reconocer las pruebas dentro del siguiente proceso, en especial los pasos 4 y 5.

Proceso:

a. **Paso 1. Antes de comenzar el desarrollo.**

- Paso 1A – Revisión de políticas y estándares.

- Paso 1B – Desarrollo de un criterio de medidas y métricas (aseguramiento de trazabilidad).

b. Paso 2. Antes de comenzar el desarrollo.

- Paso 2A – Revisión de los requerimientos de seguridad.
- Paso 2B – Disenso de revisión de arquitectura.
- Paso 2C – Creación y revisión de modelos UML.
- Paso 2D – Creación y revisión de modelos de amenazas.

c. Paso 3. Durante el desarrollo.

- Paso 3A – Saltos de código.
- Paso 3B – Revisión de código.

d. Paso 4. Durante el despliegue.

- Paso 4A – Teseo de penetración sobre la aplicación.
- Paso 4B – Teseo sobre la administración y configuración.

e. Paso 5. Operación y mantenimiento.

- Paso 5A – Revisión operacional.
- Paso 5B – Conducción de chequeos periódicos.
- Paso 5C – Verificación del control de cambio.

2.2. Tipos de artefactos

Como ya se ha mencionado, los artefactos tecnológicos son aquellos dispositivos concebidos y creados para ayudar a resolver necesidades o facilitar ciertas tareas, empleando para su construcción y funcionamiento las virtudes de la técnica y la ciencia.

De acuerdo con la estrategia definida y según la metodología seleccionada, se describen los tipos de artefactos a construir para esta prueba, así:

Video 3. Tipos de artefactos

Enlace de reproducción del video

Síntesis del video: Tipos de artefactos

Esta fase del proceso de recopilación de información consiste en navegar y capturar los recursos relacionados con la aplicación web, como la configuración robots.txt.; esta fase es de uso frecuente por los motores de búsqueda para categorizar los enlaces internos.

Una vez los robots han completado el proceso de "crawling", inicia la indexación del contenido de las páginas web, basándose en el contenido y los atributos más relevantes.

Este componente es importante porque permite conocer cómo funciona la aplicación, como es la interacción de los usuarios y el navegador. Las peticiones "GET" y "POST" son un componente importante dentro de las páginas web.

Las peticiones "GET" son usadas cuando es información sensible, cuando se hace una petición "POST" se complementa usándola por "GET"; es común que dentro de las páginas web se envía información sin que el usuario se dé cuenta.

Saber el tipo exacto y versión del servidor web ayuda considerablemente el proceso de análisis empleado en el sitio web, permite determinar vulnerabilidades conocidas en dichas versiones y los "exploits" apropiados a usar durante la prueba, puede cambiar el curso de las pruebas.

En la actualidad, es común que a una dirección IP se le asignen múltiples dominios, al hacer el análisis de los puertos se encuentra que algunos servicios, que, a pesar de estar instalados en el servidor, no son visibles para los usuarios de la página web o para quien quiera hacer un análisis de los servicios instalados.

2.3. Diseño de artefactos

Con base en la metodología y los tipos de artefactos a utilizar se construye un modelo de implementación de estos:

a. **Spiders, "robots" y "crawlers" (OWASP-IG-001).** Los "crawlers/robots/spiders" web se encargan de inspeccionar los sitios web. Su comportamiento está estrictamente controlado por el protocolo de exclusión de "robots" del fichero robots.txt almacenado en la raíz del sitio web.

b. **Reconocimiento mediante motores de búsqueda (OWASP-IG-002).** Utilizar varios navegadores y realizar las siguientes búsquedas de su sitio web con el mismo comando.

c. **Identificación de puntos de entrada de la aplicación (OWASP-IG-003).** Se utiliza el complemento HTTPS "headers" disponible para Mozilla Firefox, al momento de verificar las solicitudes al sitio web www.sena.edu.co se debe buscar si utilizan los dos tipos de peticiones; "GET" y "POST".

d. **Pruebas para encontrar firmas de aplicaciones web (OWASP-IG-004).** En ambientes Linux se usa la herramienta netcat para leer datos de la red y con el siguiente comando se comprueba el campo Server en la cabecera de respuesta HTTP.

   nc 172.17.8.279 80 es decir comando nc más ip del servidor y puerto.

e. **Descubrimiento de aplicaciones (OWASP-IG-005).** El objetivo es descubrir los servicios que presta un servidor web por medio del siguiente comando en ambiente Linux:

nmap -Pn -sT -sV -p0-65535 172.157.25.189

Para descubrir nombres de dominios asociados a mi paginaweb.com se usa el comando en ambiente Linux:

host -t ns mipaginaweb.com

f. Análisis de códigos de error (OWASP-IG-006)

- HTTP 404 "Not Found" = página web no encontrada.
- HTTP 403 "Forbidden" = intento de acceso a recurso limitado.
- HTTP 301 "Moved Permanently" = recurso redireccionado a un sitio permanente.

g. Pruebas de gestión de configuración de infraestructura

- Pruebas de SSL/TLS (OWASP-CM-001). Para detectar el posible soporte a cifrados débiles en SSL/TLS, comando

  nmap -sv - -reason -PN -n -top-port 165.125.142.123

- Auditoría cifrado SSL débiles mediante OpenSSL en primer tener openssl instalado

  openssl s_client -connect mipaginaweb.com

- Pruebas del receptor de escucha de la BD (OWASP-CM-002)
- Pruebas de gestión de configuración de la infraestructura (OWASP-CM-003). Para establecer la información relevante de las aplicaciones y medidas de seguridad que están siendo

implementadas dentro de una máquina de prueba se ejecuta el siguiente comando en un entorno Linux:

HEAD mipaginaweb.com para ftp comando ftp y para un balanceador de carga el comando lbd mipaginaweb.com

- Pruebas de gestión de configuración de la aplicación (OWASP-CM-004).

  Gestión de extensiones de archivo (OWASP-CM-005).

  Archivos antiguos, copias de seguridad y sin referencias (OWASP-CM-006).

  Interfaces de administración de la infraestructura y de la aplicación (OWASP-CM-007).

  Métodos HTTP y XST (OWASP-CM-008).

h. Comprobación del sistema de autenticación

- Transmisión de credenciales a través de un canal cifrado (OWASP-AT001).
- Enumeración de usuarios (OWASP-AT-002).
- Pruebas de diccionario sobre cuentas de usuario o cuentas predeterminadas (OWASP-AT-003).
- Pruebas de fuerza bruta (OWASP-AT-004).
- Saltarse el sistema de autenticación (OWASP-AT-005).

- Comprobar sistemas de recordatorio/restauración de contraseñas vulnerables (OWASP-AT-006).
- Pruebas de gestión del caché de navegación y de salida de sesión (OWASP-AT-007).
- Pruebas de "captcha" (OWASP-AT-008).
- Pruebas para autenticación de factores múltiples (OWASP-AT-009).
- Probar por situaciones adversas (OWASP-AT-010).
- Pruebas para el esquema de gestión de sesiones (OWASP-Sm-001).

i. **Pruebas de validación de datos.** Pruebas de XSS. "Cross Site Scripting" e Inyección SQL.

3. Entorno de pruebas

Un ambiente de pruebas es un entorno recomendado que pretende emular, ojalá al 100 %, el ecosistema tecnológico en el que funciona una aplicación en este caso una aplicación web que ya se encuentra en uso por parte de una organización.

Es decir, ya se encuentra funcionando en un ambiente productivo, de ahí la necesidad de identificar claramente de ese entorno productivo los siguientes ítems para replicarlos en el entorno de pruebas: sistemas operativos, servidores de aplicaciones, sistemas de bases de datos, mecanismos de interoperabilidad con otras aplicaciones, entornos de red, documentación correspondiente a manuales técnicos y de usuario, marcos de trabajo ("frameworks") y demás "software" necesario para ejecutar las pruebas; para cada uno de estos es imprescindible la identificación de sus versiones y realizar el seguimiento a posibles cambios y sus dependencias.

El entorno de pruebas recomendado para este tipo de plan que se va a realizar del lado servidor se basa en sistemas operativos de la familia Unix (Linux en diferentes versiones CentOS, Red Hat, Ubuntu, FreeBSD o macOS) sin desconocer la necesidad de realizarlos en ambientes Windows, entre otros sistemas; el objetivo es ejecutar las pruebas sin temor a equivocarse para luego realizarlas en ambientes de producción.

A continuación, se muestra un ejemplo de implementación de ambiente de pruebas recomendado que consta de cuatro pasos.

- **Paso 1: identificar servicios remotos.** Consiste en identificar los servicios remotos para lo cual se puede utilizar la herramienta nmap que permite el escaneo de redes en donde se puede identificar qué servicios se están utilizando en un dispositivo remoto, así como la identificación de equipos

activos, sistemas operativos en el equipo remoto, existencia de filtros o "firewalls", entre otros.

Por línea de comando en Linux su utilización es así más la ip objetivo en donde la respuesta será un listado de los puertos abiertos o cerrados en esa dirección ip.

Algunos parámetros del comando son:

[-iL] Sirve para indicar una lista de direcciones ips o redes a escanear. > nmap -iL hosts.txt

[-sP] Sirve para ver cuántos equipos con sus Ips se pueden escanear.

[-p] Lista los puertos que se desean escanear.> nmap -iL hosts.txt -p 22,25,80,445

[-sV] Sirve para tratar de determinar la versión del servicio en el destino remoto.

[-O] Informa el sistema operativo en el objetivo.

- **Paso 2: buscar vulnerabilidades.** Una vez encontrados los servicios remotos con esta información, se pueden utilizar las herramientas apropiadas para hallar los huecos de seguridad de esos servicios.

  Entonces, con los datos obtenidos del paso anterior existe una herramienta que se llama Nessus para iniciarse en esta actividad; esta consiste en una aplicación de carácter gratuita que también cuenta con la versión de pago, que por su base de datos y su facilidad de uso es muy apropiada en este

aspecto; aunque se puede utilizar por línea de comandos, se recomienda utilizar su interfaz gráfica que es bastante intuitiva.

- **Paso 3: explotar vulnerabilidades.** Una vez encontradas las vulnerabilidades se recomienda el uso de la herramienta Metasploit para determinar cómo se pueden explotar esas vulnerabilidades. Así, en primer lugar, se tiene que verificar si realmente las vulnerabilidades identificadas permiten a un atacante causar algún daño para luego intentar conocer cuál sería ese daño.

  Metasploit cuenta con una base de datos de vulnerabilidades y de "exploits" que podrían aprovecharlas. En otras palabras, en lugar de revisar si hay una vulnerabilidad en un equipo remoto, directamente se intenta la ejecución de un "exploit" y se simulan las consecuencias posteriores, en caso de que este se ejecutará con éxito.

  Finalmente, Metasploit se ejecuta por medio de línea de comandos, msfconsole, y se recomienda usar su interfaz gráfica para tener una mayor comprensión.

- **Paso 4: probar en entornos legales.** Este paso se hace necesario para las tres herramientas anteriores y definir un sistema objetivo en el que se harán las pruebas. Se recomienda iniciar en estas labores de pruebas en un entorno real, en ningún caso en sistemas públicos de internet. Para aprender a usar estas herramientas, se debe utilizar el entorno de pruebas, construyendo escenarios de investigación en donde cada persona pueda

tener acercamientos sin riesgos de afectar a ningún entorno en producción.

Las herramientas recomendadas aquí son “Damn Vulnerable Linuxy” (DVL) y “Damn Vulnerable Web Application” (DVWA).

3.1. Herramientas para pruebas

Para complementar las herramientas de un entorno de pruebas asociadas a la metodología utilizada, a continuación, se mencionan algunas de las herramientas más conocidas para ejecutar este tipo de pruebas:

- Navegadores web Mozilla, Chrome, etc.
- Motores de búsqueda Bing, Google.
- Complemento HTTPS “headers” para Mozilla.
- Servidores de base de datos y sus respectivos clientes, servidores de aplicaciones y balanceadores de carga de acuerdo con el análisis y el ambiente que se vaya a verificar.

Además, también se pueden utilizar las siguientes:

- **OpenSSL.** Verifica certificados digitales y demás métodos de encriptamiento de información.
- **Zed Attack Proxy.** Herramienta de “pentesting” que ayuda a encontrar vulnerabilidades en nuestras aplicaciones.

- **OWTF.** Herramienta de "pentesting" que es, quizás, un poco más sencilla que la herramienta anterior.
- **Dependency Check.** Herramienta que permite analizar todas las dependencias de nuestra aplicación y comprobar si existen vulnerabilidades dentro de ellas.
- **SSL advanced forensic tool.** Herramienta que permite mostrar información sobre SSL y los certificados.

3.2. Herramientas de gestión

Para realizar labores de control de operaciones, registro, administración, documentación y seguimiento a casos se describen las siguientes posibilidades:

- **TestLink.** Sistema de gestión de pruebas de "software" basado en la web.
- **Zephyr. "**Software" de gestión de ciclo de pruebas de "software", disponible como versión empresarial y como un Add-On de Atalassian Jira.
- **Bugzilla.** Sistema de seguimiento de defectos ("Bug Tracking System").
- **Gemini.** Solución automatización de flujo de trabajo, comunicaciones y reportes en una gran variedad de escenarios de gestión de tecnología de información (TI).
- **Selenium WebDriver.** Es uno de los principales exponentes en la automatización de "software testing" para aplicaciones web hoy en día.

3.3. Herramientas de operación

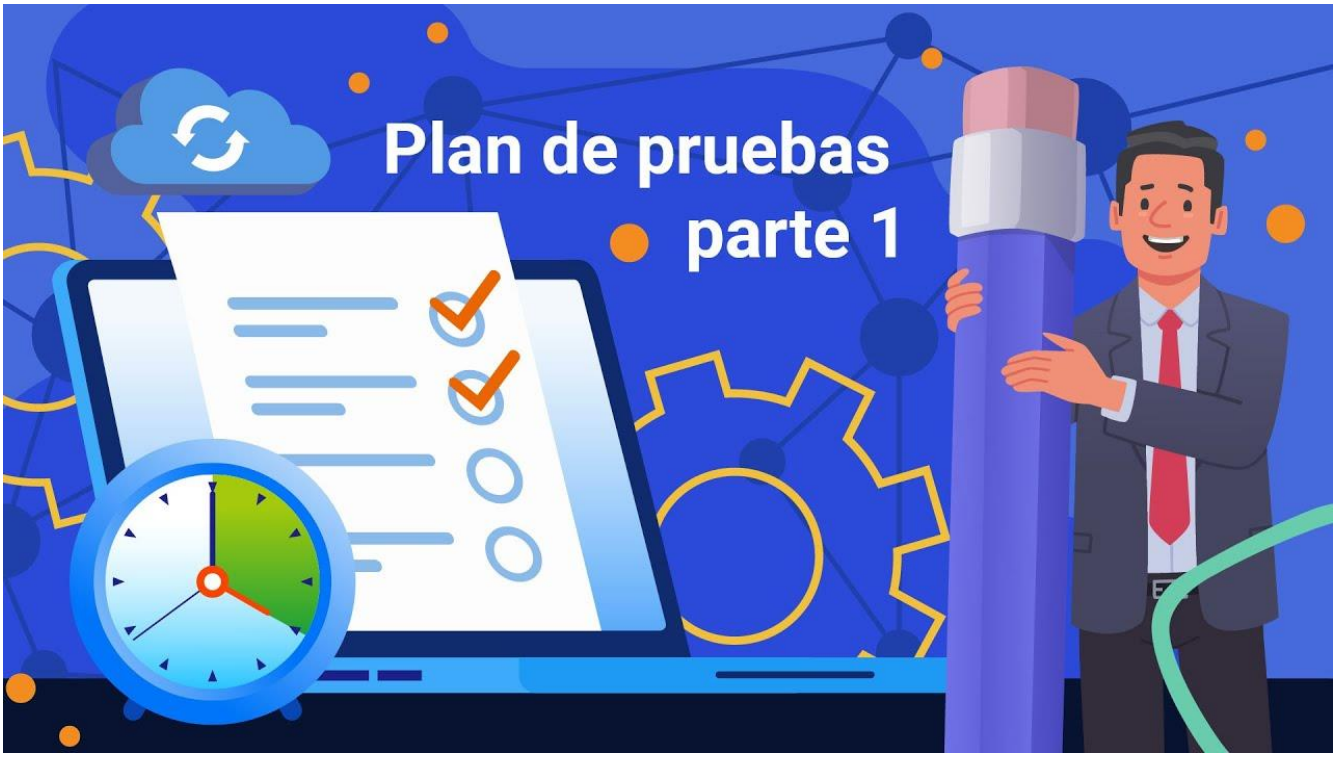
Una vez se cuente con un ambiente de producción, se pretende implementar el seguimiento y monitoreo a dicho ambiente para lo cual se recomienda el uso de estas herramientas:

- **Netdata.** Es un "software" de monitorización gratuito y "open source", que permite supervisar todo tipo de KPI en tiempo real.
- **Zabbix.** Es una de las herramientas de "software" para monitorización de servidores más conocidas.
- **Monit.** Es un "software" de monitorización que se emplea en sistemas Linux o basados en UNIX.
- **Ntopng.** Se ha erigido como uno de los "software" de monitorización de redes más completos.
- **New Relic.** Es una herramienta que permite realizar una monitorización integral del rendimiento de aplicaciones o de usuarios reales.

Hemos llegado a la finalización de este componente, pero antes es importante que observar las videoclases que se presentarán a continuación y que servirán como apoyo al ejercicio práctico que debe realizar en la elaboración de un plan de pruebas.

El siguiente video presenta un tutorial del proceso de elaboración de un plan de pruebas con base en el contenido estándar OWASP.

Video 4. Plan de pruebas parte 1



Enlace de reproducción del video

Síntesis del video: Plan de pruebas parte 1
El video presenta la primera parte de un tutorial sobre el proceso de elaboración de un plan de pruebas con base en el contenido estándar OWASP.

Para dar continuidad al ejercicio que se viene desarrollando, lo invitamos a ver el siguiente video.

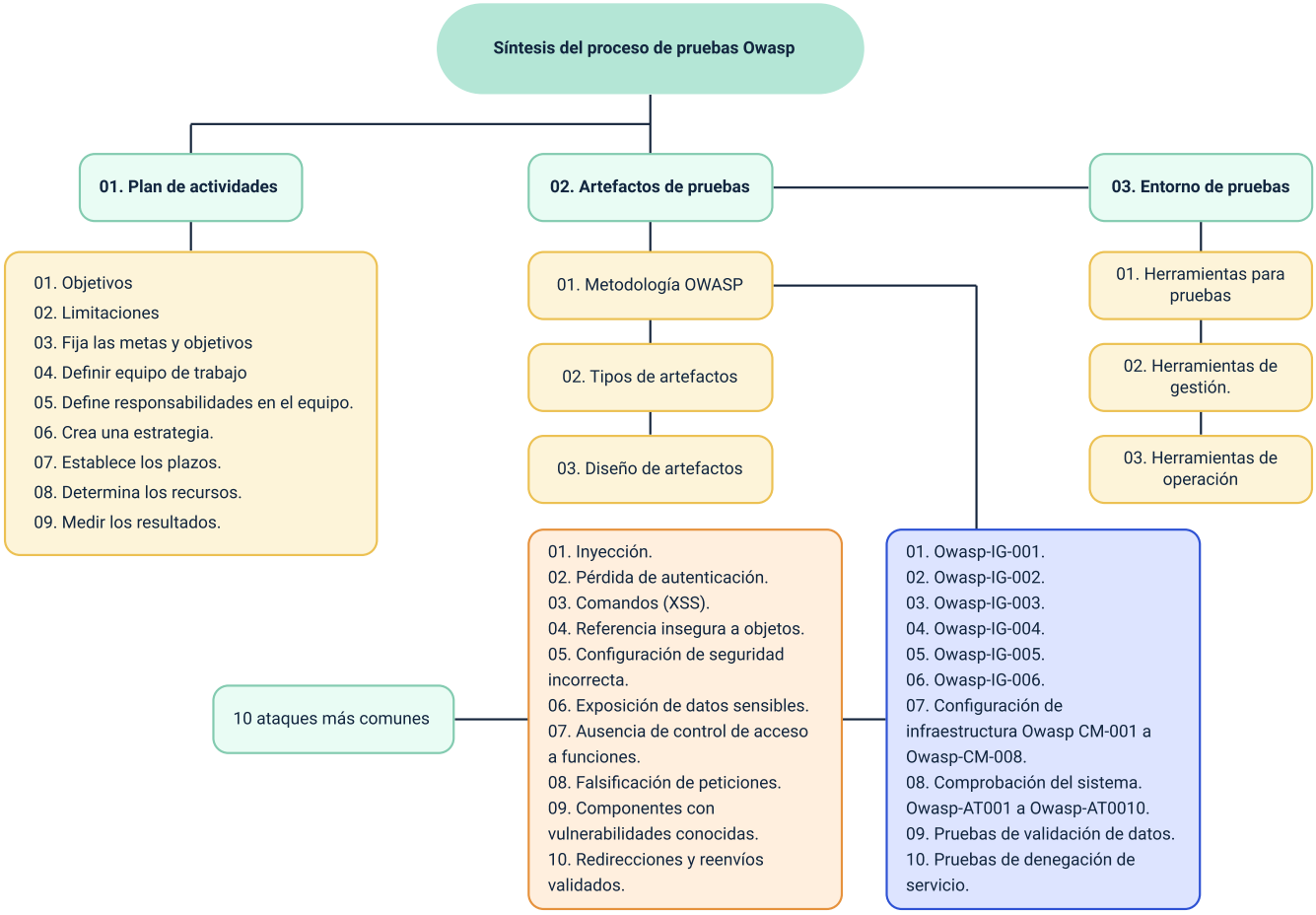
Video 5. Plan de pruebas parte 2

Enlace de reproducción del video

Síntesis del video: Plan de pruebas parte 2
Parte 2 del tutorial sobre el proceso de elaboración de un plan de pruebas con base en el contenido estándar OWASP.

Síntesis

En el siguiente recurso se expone un resumen de este componente formativo:



El esquema presenta la síntesis de la temática estudiada en el componente formativo, comenzando por el proceso de pruebas OWASP, compuesto por:

a. Plan de actividades:

- Objetivos.
- Limitaciones.
- Fija las metas y objetivos.

- Definir equipo de trabajo.
- Define responsabilidades en el equipo.
- Crea una estrategia.
- Establece los plazos.
- Determina los recursos.
- Medir los resultados.

b. Artefactos de pruebas:

- Metodología OWASP.
- Tipos de artefactos.
- Diseño de artefactos.

c. Entorno de pruebas:

- Herramientas para pruebas.
- Herramientas de gestión.
- Herramientas de operación.

Material complementario

Tema	Referencia	Tipo de material	Enlace del recurso
Plan de Objetivos	Caballero, Q., A. E. (2019). Webinar gratuito: guía de pruebas de OWASP.	Video	https://www.youtube.com/watch?v=kXfZqQY0rcg&ab_channel=AlonsoCaballero

Glosario

“Checklist**”:** lista de chequeo que sirve para registrar un proceso de auditoría.

“Exploits**”:** parte de un “software” o una secuencia de comandos que se aprovecha de un error o vulnerabilidad.

“Netcat**”:** herramienta de línea de comandos que sirve para escribir y leer datos en la red.

OWASP: “Open Web Application Security Project**”.**

“Pentesting**”:** proceso que imita posibles ataques a una red informática e intenta robar datos.

Referencias bibliográficas

Amor. F. (2020). Introducción a OWASP. https://www.adictosaltrabajo.com/2016/03/07/introduccion-a-OWASP/

Asana, (2020). 4 técnicas eficaces para definir roles y responsabilidades. https://asana.com/es/resources/roles-and-responsibilities

Diaz, M., y Marulanda, M. F. (2018). Proyecto de aplicación de OWASP. https://repository.unad.edu.co/bitstream/handle/10596/20479/1060648494.pdf?sequence=3&isAllowed=y#page=57&zoom=100,148,204

Junta de Andalucía. (2019). OWASP Testing Project. Junta de Andalucía. https://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/553

Créditos

Nombre	Cargo	Centro de Formación y Regional
Claudia Patricia Aristizábal	Responsable del Ecosistema	Dirección General
Rafael Neftalí Lizcano Reyes	Responsable de Línea de Producción	Centro Industrial del Diseño y la Manufactura - Regional Santander
Carlos Hernán Muñoz	Experto Temático	Centro de teleinformática y producción industrial - Regional Cauca
Paula Andrea Taborda Ortiz	Diseñadora Instruccional	Centro de la Industria, la Empresa y Los Servicios CIES - Regional Norte de Santander
Carolina Coca Salazar	Asesora Metodológica	Centro de Diseño y Metrología - Regional Distrito Capital
José Gabriel Ortiz Abella	Corrector de estilo	Centro de Diseño y Metrología - Regional Distrito Capital
Juan Daniel Polanco Muñoz	Diseñador de Contenidos Digitales	Centro Industrial del Diseño y la Manufactura - Regional Santander
Edward Leonardo Pico Cabra	Desarrollador Fullstack	Centro Industrial del Diseño y la Manufactura - Regional Santander
María Natalia Maldonado Delgado	Diseño web	Centro Industrial del Diseño y la Manufactura - Regional Santander
Zuleidy María Ruíz Torres	Revisión de guion audiovisual	Centro de Comercio y Servicios - Regional Tolima
María Carolina Tamayo López	Locución	Centro Industrial del Diseño y la Manufactura - Regional Santander
Yicelly Estefania Mesa Silva	Ilustración	Centro Industrial del Diseño y la Manufactura - Regional Santander
Lady Adriana Ariza Luque	Validación Ilustración	Centro Industrial del Diseño y la Manufactura - Regional Santander

Nombre	Cargo	Centro de Formación y Regional
Wilson Andrés Arenales Cáceres	Validación Ilustración	Centro Industrial del Diseño y la Manufactura - Regional Santander
John Jairo Arciniegas González	Producción audiovisual	Centro Industrial del Diseño y la Manufactura - Regional Santander
Oleg Litvin	Producción audiovisual	Centro Industrial del Diseño y la Manufactura - Regional Santander
Laura Gisselle Murcia Pardo	Producción audiovisual	Centro Industrial del Diseño y la Manufactura - Regional Santander
Gilberto Junior Rodríguez Rodríguez	Validación audiovisual	Centro Industrial del Diseño y la Manufactura - Regional Santander
Diego Fernando Velasco Güiza	Desarrollo front-end	Centro Industrial del Diseño y la Manufactura - Regional Santander
Yenny Patricia Ulloa Villamizar	Validación de contenido	Centro Industrial del Diseño y la Manufactura - Regional Santander
Zuleidy María Ruiz Torres	Validador de Recursos Educativos Digitales	Centro Industrial del Diseño y la Manufactura - Regional Santander
Luis Gabriel Urueta Álvarez	Validador de Recursos Educativos Digitales	Centro Industrial del Diseño y la Manufactura - Regional Santander
Daniel Ricardo Mutis Gómez	Evaluador para contenidos inclusivos y accesibles	Centro Industrial del Diseño y la Manufactura - Regional Santander